

To our customers,

---

## Old Company Name in Catalogs and Other Documents

---

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

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

April 1<sup>st</sup>, 2010  
Renesas Electronics Corporation

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

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

## Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
  - “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
  - “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
  - “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

# **$\mu$ PD179F11x, 179F12x Microcontrollers**

**8-bit Single-Chip Microcontrollers**

---

**$\mu$ PD179F110**

**$\mu$ PD179F111**

**$\mu$ PD179F112**

**$\mu$ PD179F113**

**$\mu$ PD179F114**

**$\mu$ PD179F122**

**$\mu$ PD179F123**

**$\mu$ PD179F124**

[MEMO]

## 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  $V_{IL}$  (MAX) and  $V_{IH}$  (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  $V_{IL}$  (MAX) and  $V_{IH}$  (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  $V_{DD}$  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.

**Windows and Windows NT are registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.**

**PC/AT is a trademark of International Business Machines Corporation.**

**HP9000 series 700 and HP-UX are trademarks of Hewlett-Packard Company.**

**SPARCstation is a trademark of SPARC International, Inc.**

**Solaris and SunOS are trademarks of Sun Microsystems, Inc.**

**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.

• **The information in this document is current as of February, 2008. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**

- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC 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 NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".

The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

"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.

"Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).

"Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

(Note)

- (1) "NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.
- (2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

M8E 02.11-1

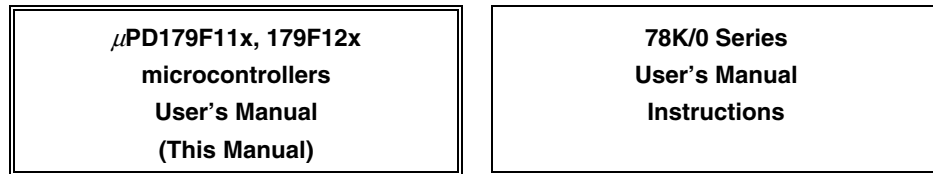
## INTRODUCTION

**Readers** This manual is intended for user engineers who wish to understand the functions of the  $\mu$ PD179F11x, 179F12x microcontrollers and design and develop application systems and programs for these devices.  
The target products are as follows.

$\mu$ PD179F11x, 179F12x microcontrollers:  $\mu$ PD179F110, 179F111, 179F112, 179F113, 179F114, 179F122, 179F123, 179F124

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

**Organization** The  $\mu$ PD179F11x, 179F12x microcontrollers manual is separated into two parts: this manual and the instructions edition (common to 78K0 microcontrollers).



- Pin functions
- Internal block functions
- Interrupts
- Other on-chip peripheral functions
- Electrical specifications
- 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**. The mark "<R>" shows major revised points. The revised points can be easily searched by copying an "<R>" in the PDF file and specifying it in the "Find what." field.
- 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 RA78K0, and is defined as an sfr variable using the #pragma sfr directive in the CC78K0.
- To know details of the 78K0 microcontroller instructions:
  - Refer to the separate document **78K/0 Series Instructions User's Manual (U12326E)**.



**Conventions**

Data significance: Higher digits on the left and lower digits on the right

Active low representations:  $\overline{\text{xxx}}$  (overscore over pin and signal name)

**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.
$\mu$ PD179F11x, 179F12x Microcontrollers User's Manual	This manual
78K/0 Series Instructions User's Manual	U12326E

**Documents Related to Development Tools (Software) (User's Manuals)**

Document Name	Document No.	
RA78K0 Ver. 3.80 Assembler Package	Operation	U17199E
	Language	U17198E
	Structured Assembly Language	U17197E
CC78K0 Ver. 3.70 C Compiler	Operation	U17201E
	Language	U17200E
ID78K0-QB Ver. 3.00 Integrated Debugger	Operation	U18492E
PM+ Ver. 6.00		U17178E

**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 (Hardware) (User's Manuals)**

Document Name	Document No.
QB-179F124 In-Circuit Emulator	U18586E
QB-MINI2 On-Chip Debug Emulator with Programming Function	U18371E

**Documents Related to Flash Memory Programming (User's Manuals)**

Document Name	Document No.
PG-FP4 Flash Memory Programmer	U15260E
PG-FP5 Flash Memory Programme	U18865E

**Other Documents**

Document Name	Document No.
SEMICONDUCTOR SELECTION GUIDE – Products and Packages –	X13769X
Semiconductor Device Mount Manual	<b>Note</b>
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 Device Mount Manual” website (<http://www.necel.com/pkg/en/mount/index.html>).

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

# CONTENTS

<b>CHAPTER 1 OUTLINE</b> .....	<b>15</b>
<b>1.1 Features</b> .....	<b>15</b>
<b>1.2 Applications</b> .....	<b>15</b>
<b>1.3 Ordering Information</b> .....	<b>16</b>
<b>1.4 Application circuit</b> .....	<b>16</b>
<b>1.5 Pin Configuration (Top View)</b> .....	<b>17</b>
<b>1.6 <math>\mu</math>PD179F11x, 179F12x Microcontrollers Lineup</b> .....	<b>19</b>
<b>1.7 Block Diagram</b> .....	<b>19</b>
<b>1.8 Outline of Functions</b> .....	<b>20</b>
<b>CHAPTER 2 PIN FUNCTIONS</b> .....	<b>22</b>
<b>2.1 Pin Function List</b> .....	<b>22</b>
<b>2.2 Description of Pin Functions</b> .....	<b>24</b>
2.2.1 P00 to P07 (port 0).....	24
2.2.2 P10 to P17 (port 1).....	25
2.2.3 P20 to P27 (port 2).....	25
2.2.4 P30 to P35 (port 3) (38-pin products only) .....	26
2.2.5 P120 to P123 (port 12).....	26
2.2.6 REGC .....	27
2.2.7 V <sub>DD</sub> .....	27
2.2.8 V <sub>SS</sub> .....	27
2.2.9 FLMD0.....	27
<b>2.3 Pin I/O Circuits and Recommended Connection of Unused Pins</b> .....	<b>27</b>
<b>CHAPTER 3 CPU ARCHITECTURE</b> .....	<b>30</b>
<b>3.1 Memory Space</b> .....	<b>30</b>
3.1.1 Internal program memory space .....	37
3.1.2 Internal data memory space .....	38
3.1.3 Special function register (SFR) area .....	38
3.1.4 Data memory addressing.....	38
<b>3.2 Processor Registers</b> .....	<b>44</b>
3.2.1 Control registers.....	44
3.2.2 General-purpose registers .....	48
3.2.3 Special function registers (SFRs).....	49
<b>3.3 Instruction Address Addressing</b> .....	<b>53</b>
3.3.1 Relative addressing .....	53
3.3.2 Immediate addressing.....	54
3.3.3 Table indirect addressing.....	55
3.3.4 Register addressing.....	55
<b>3.4 Operand Address Addressing</b> .....	<b>56</b>
3.4.1 Implied addressing.....	56
3.4.2 Register addressing.....	57
3.4.3 Direct addressing.....	58

3.4.4	Short direct addressing .....	59
3.4.5	Special function register (SFR) addressing.....	60
3.4.6	Register indirect addressing.....	61
3.4.7	Based addressing .....	62
3.4.8	Based indexed addressing.....	63
3.4.9	Stack addressing.....	64
<b>CHAPTER 4</b>	<b>PORT FUNCTIONS .....</b>	<b>65</b>
<b>4.1</b>	<b>Port Functions .....</b>	<b>65</b>
<b>4.2</b>	<b>Port Configuration .....</b>	<b>67</b>
4.2.1	Port 0 .....	68
4.2.2	Port 1 .....	73
4.2.3	Port 2 .....	74
4.2.4	Port 3 (38-pin products only).....	77
4.2.5	Port 12 .....	78
<b>4.3</b>	<b>Registers Controlling Port Function .....</b>	<b>81</b>
<b>4.4</b>	<b>Port Function Operations.....</b>	<b>85</b>
4.4.1	Writing to I/O port.....	85
4.4.2	Reading from I/O port.....	85
4.4.3	Operations on I/O port.....	85
<b>4.5</b>	<b>Settings of Port Mode Register, Output Latch, Pull-up Resistor Option Register, and Port Output Mode Register When Using Alternate Function.....</b>	<b>86</b>
<b>4.6</b>	<b>Cautions on 1-bit Manipulation Instruction for Port Register n (Pn).....</b>	<b>87</b>
<b>CHAPTER 5</b>	<b>CLOCK GENERATOR .....</b>	<b>88</b>
<b>5.1</b>	<b>Functions of Clock Generator.....</b>	<b>88</b>
<b>5.2</b>	<b>Configuration of Clock Generator .....</b>	<b>89</b>
<b>5.3</b>	<b>Registers Controlling Clock Generator .....</b>	<b>91</b>
<b>5.4</b>	<b>System Clock Oscillator .....</b>	<b>99</b>
5.4.1	X1 oscillator.....	99
5.4.2	Internal high-speed oscillator .....	100
5.4.3	Internal low-speed oscillator.....	100
5.4.4	Prescaler.....	101
<b>5.5</b>	<b>Clock Generator Operation .....</b>	<b>101</b>
<b>5.6</b>	<b>Controlling Clock .....</b>	<b>103</b>
5.6.1	Example of controlling high-speed system clock.....	103
5.6.2	Example of controlling internal high-speed oscillation clock.....	105
5.6.3	Example of controlling internal low-speed oscillation clock.....	107
5.6.4	Clocks supplied to CPU and peripheral hardware.....	108
5.6.5	CPU clock status transition diagram .....	109
5.6.6	Condition before changing CPU clock and processing after changing CPU clock.....	112
5.6.7	Time required for switchover of CPU clock and main system clock.....	112
5.6.8	Conditions before clock oscillation is stopped.....	113
5.6.9	Peripheral hardware and source clocks .....	114

<b>CHAPTER 6</b>	<b>16-BIT TIMER/EVENT COUNTER 00</b>	<b>115</b>
6.1	Functions of 16-bit Timer/Event Counter 00	115
6.2	Configuration of 16-bit Timer/Event Counter 00	116
6.3	Registers Controlling 16-bit Timer/Event Counter 00	121
6.4	Operation of 16-bit Timer/Event Counter 00	129
6.4.1	Interval timer operation	129
6.4.2	Square wave output operation	132
6.4.3	External event counter operation	135
6.4.4	Operation in clear & start mode entered by TI000 pin valid edge input	138
6.4.5	Free-running timer operation	151
6.4.6	PPG output operation	160
6.4.7	One-shot pulse output operation	163
6.4.8	Pulse width measurement operation	168
6.5	Special Use of TM00	176
6.5.1	Rewriting CR010 during TM00 operation	176
6.5.2	Setting LVS00 and LVR00	176
6.6	Cautions for 16-bit Timer/Event Counter 00	178
<b>CHAPTER 7</b>	<b>8-BIT TIMER/EVENT COUNTERS 50 AND 51</b>	<b>182</b>
7.1	Functions of 8-bit Timer/Event Counters 50 and 51	182
7.2	Configuration of 8-bit Timer/Event Counters 50 and 51	182
7.3	Registers Controlling 8-bit Timer/Event Counters 50 and 51	185
7.4	Operations of 8-bit Timer/Event Counters 50 and 51	190
7.4.1	Operation as interval timer	190
7.4.2	Operation as external event counter	192
7.4.3	Square-wave output operation	193
7.4.4	PWM output operation	194
7.5	Cautions for 8-bit Timer/Event Counters 50 and 51	198
<b>CHAPTER 8</b>	<b>8-BIT TIMERS H0 AND H1</b>	<b>199</b>
8.1	Functions of 8-bit Timers H0 and H1	199
8.2	Configuration of 8-bit Timers H0 and H1	199
8.3	Registers Controlling 8-bit Timers H0 and H1	203
8.4	Operation of 8-bit Timers H0 and H1	208
8.4.1	Operation as interval timer/square-wave output	208
8.4.2	Operation as PWM output	211
8.4.3	Carrier generator operation (8-bit timer H1 only)	217
<b>CHAPTER 9</b>	<b>WATCHDOG TIMER</b>	<b>224</b>
9.1	Functions of Watchdog Timer	224
9.2	Configuration of Watchdog Timer	225
9.3	Register Controlling Watchdog Timer	226
9.4	Operation of Watchdog Timer	227
9.4.1	Controlling operation of watchdog timer	227
9.4.2	Setting overflow time of watchdog timer	228
9.4.3	Setting window open period of watchdog timer	229

<b>CHAPTER 10 SERIAL INTERFACE UART6 .....</b>	<b>231</b>
<b>10.1 Functions of Serial Interface UART6.....</b>	<b>231</b>
<b>10.2 Configuration of Serial Interface UART6 .....</b>	<b>232</b>
<b>10.3 Registers Controlling Serial Interface UART6.....</b>	<b>235</b>
<b>10.4 Operation of Serial Interface UART6.....</b>	<b>243</b>
10.4.1 Operation stop mode.....	243
10.4.2 Asynchronous serial interface (UART) mode .....	244
10.4.3 Dedicated baud rate generator.....	255
10.4.4 Calculation of baud rate .....	257
<b>CHAPTER 11 INTERRUPT FUNCTIONS .....</b>	<b>262</b>
<b>11.1 Interrupt Function Types.....</b>	<b>262</b>
<b>11.2 Interrupt Sources and Configuration .....</b>	<b>262</b>
<b>11.3 Registers Controlling Interrupt Functions .....</b>	<b>266</b>
<b>11.4 Interrupt Servicing Operations .....</b>	<b>273</b>
11.4.1 Maskable interrupt acknowledgment.....	273
11.4.2 Software interrupt request acknowledgment .....	275
11.4.3 Multiple interrupt servicing .....	276
11.4.4 Interrupt request hold .....	279
<b>CHAPTER 12 KEY INTERRUPT FUNCTION .....</b>	<b>280</b>
<b>12.1 Functions of Key Interrupt .....</b>	<b>280</b>
<b>12.2 Configuration of Key Interrupt.....</b>	<b>280</b>
<b>12.3 Register Controlling Key Interrupt .....</b>	<b>281</b>
<b>CHAPTER 13 STANDBY FUNCTION .....</b>	<b>283</b>
<b>13.1 Standby Function and Configuration.....</b>	<b>283</b>
13.1.1 Standby function .....	283
13.1.2 Registers controlling standby function.....	283
<b>13.2 Standby Function Operation.....</b>	<b>286</b>
13.2.1 HALT mode.....	286
13.2.2 STOP mode .....	288
<b>CHAPTER 14 RESET FUNCTION.....</b>	<b>294</b>
<b>14.1 Register for Confirming Reset Source.....</b>	<b>300</b>
<b>CHAPTER 15 POWER-ON-CLEAR CIRCUIT.....</b>	<b>301</b>
<b>15.1 Functions of Power-on-Clear Circuit .....</b>	<b>301</b>
<b>15.2 Configuration of Power-on-Clear Circuit .....</b>	<b>302</b>
<b>15.3 Operation of Power-on-Clear Circuit.....</b>	<b>302</b>
<b>15.4 Cautions for Power-on-Clear Circuit.....</b>	<b>304</b>
<b>CHAPTER 16 LOW-VOLTAGE DETECTOR .....</b>	<b>306</b>
<b>16.1 Functions of Low-Voltage Detector.....</b>	<b>306</b>
<b>16.2 Configuration of Low-Voltage Detector .....</b>	<b>307</b>

<b>16.3</b>	<b>Registers Controlling Low-Voltage Detector .....</b>	<b>307</b>
<b>16.4</b>	<b>Operation of Low-Voltage Detector .....</b>	<b>310</b>
	16.4.1 When used as reset.....	311
	16.4.2 When used as interrupt.....	315
<b>16.5</b>	<b>Cautions for Low-Voltage Detector.....</b>	<b>319</b>
<b>16.6</b>	<b>RAM Data Retention Detector.....</b>	<b>322</b>
<b>CHAPTER 17 OPTION BYTE.....</b>		<b>323</b>
<b>17.1</b>	<b>Functions of Option Bytes .....</b>	<b>323</b>
<b>17.2</b>	<b>Format of Option Byte.....</b>	<b>324</b>
<b>CHAPTER 18 FLASH MEMORY.....</b>		<b>326</b>
<b>18.1</b>	<b>Internal Memory Size Switching Register .....</b>	<b>326</b>
<b>18.2</b>	<b>Writing with Flash Memory Programmer.....</b>	<b>327</b>
<b>18.3</b>	<b>Programming Environment.....</b>	<b>329</b>
<b>18.4</b>	<b>Communication Mode .....</b>	<b>329</b>
<b>18.5</b>	<b>Handling of Pins on Board.....</b>	<b>330</b>
	18.5.1 FLMD0 pin .....	330
	18.5.2 Serial interface pins .....	331
	18.5.3 $\overline{\text{RESET}}$ pin .....	332
	18.5.4 Port pins.....	333
	18.5.5 REGC pin.....	333
	18.5.6 Other signal pins.....	333
	18.5.7 Power supply .....	333
<b>18.6</b>	<b>Programming Method.....</b>	<b>334</b>
	18.6.1 Controlling flash memory .....	334
	18.6.2 Flash memory programming mode .....	334
	18.6.3 Selecting communication mode .....	335
	18.6.4 Communication commands .....	335
<b>18.7</b>	<b>Security Settings.....</b>	<b>336</b>
<b>18.8</b>	<b>Flash Memory Programming by Self-Programming.....</b>	<b>339</b>
<b>CHAPTER 19 ON-CHIP DEBUG FUNCTION.....</b>		<b>342</b>
<b>19.1</b>	<b>Connecting QB-MINI2 .....</b>	<b>342</b>
<b>19.2</b>	<b>Reserved Area Used by QB-MINI2 .....</b>	<b>344</b>
<b>CHAPTER 20 INSTRUCTION SET .....</b>		<b>345</b>
<b>20.1</b>	<b>Conventions Used in Operation List.....</b>	<b>345</b>
	20.1.1 Operand identifiers and specification methods .....	345
	20.1.2 Description of operation column .....	346
	20.1.3 Description of flag operation column.....	346
<b>20.2</b>	<b>Operation List.....</b>	<b>347</b>
<b>20.3</b>	<b>Instructions Listed by Addressing Type .....</b>	<b>355</b>
<b>CHAPTER 21 ELECTRICAL SPECIFICATIONS.....</b>		<b>358</b>

<b>CHAPTER 22 PACKAGE DRAWINGS .....</b>	<b>369</b>
<b>APPENDIX A DEVELOPMENT TOOLS.....</b>	<b>371</b>
<b>A.1 Software Package .....</b>	<b>374</b>
<b>A.2 Language Processing Software.....</b>	<b>374</b>
<b>A.3 Control Software .....</b>	<b>375</b>
<b>A.4 Flash Memory Writing Tools .....</b>	<b>376</b>
A.4.1 When using flash memory programmer PG-FP4, FL-PR4, PG-FP5 and FL-PR5 .....	376
A.4.2 When using on-chip debug emulator with programming function QB-MINI2 .....	376
<b>A.5 Debugging Tools (Hardware).....</b>	<b>377</b>
A.5.1 When using in-circuit emulator QB-179F124 .....	377
A.5.2 When using on-chip debug emulator with programming function QB-MINI2 .....	378
<b>A.6 Debugging Tools (Software) .....</b>	<b>378</b>
<b>APPENDIX B REVISION HISTORY .....</b>	<b>379</b>
<b>B.1 Major Revisions in This Edition.....</b>	<b>379</b>
<b>B.2 Revision History up to Previous Editions .....</b>	<b>381</b>



## CHAPTER 1 OUTLINE

### 1.1 Features

- High speed (1  $\mu$ s:  $V_{DD} = 1.8$  to  $3.6$  V, high-speed system clock: @  $f_{XH} = 2$  or  $4$  MHz operation, 0.5  $\mu$ s:  $V_{DD} = 2.0$  to  $3.6$  V, high-speed system clock: @  $f_{XH} = 4$  MHz operation)
- General-purpose register: 8 bits  $\times$  32 registers (8 bits  $\times$  8 registers  $\times$  4 banks)
- ROM, RAM capacities

Part Number \ Item	Program Memory (ROM)		Data Memory
			Internal High-Speed RAM <sup>Note</sup>
$\mu$ PD179F110	Flash memory <sup>Note</sup>	4 KB	512 bytes
$\mu$ PD179F111		8 KB	
$\mu$ PD179F112, 179F122		16 KB	768 bytes
$\mu$ PD179F113, 179F123		24 KB	1 KB
$\mu$ PD179F114, 179F124		32 KB	

**Note** The internal flash memory, and internal high-speed RAM capacities can be changed using the internal memory size switching register (IMS). For IMS, see **18.1 Internal Memory Size Switching Register**.

- On-chip single-power-supply flash memory
- Self-programming (with boot swap function)
- On-chip debug function
- 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)
- On-chip key interrupt function
- I/O ports:
  - 30-pin products ( $\mu$ PD179F11x microcontrollers) : 26 (N-ch open drain/CMOS: 24, P-ch open drain<sup>Note1</sup>/CMOS: 1)
  - 38-pin products ( $\mu$ PD179F12x microcontrollers) : 34 (N-ch open drain: 32, P-ch open drain<sup>Note1</sup>/CMOS: 1)
- Timer: 6 channels
  - 16-bit timer/event counter: 1 channel
  - 8-bit timer/event counter: 2 channels
  - 8-bit timer: 2 channels
  - Watchdog timer: 1 channel
- Serial interface: 1 channel
  - UART: 1 channel
- Power supply voltage:  $V_{DD} = 1.8$  to  $3.6$  V<sup>Note2</sup>
- Operating ambient temperature:  $T_A = -40$  to  $+85^\circ\text{C}$

**Notes 1.** Available for a remote control output

**2.** Use this product in a voltage range of 1.9 to 3.6 V because the detection voltage ( $V_{POC}$ ) of the power-on-clear (POC) circuit is  $1.8\text{ V} \pm 0.1\text{ V}$ .

### 1.2 Applications

- Preset remote control

### 1.3 Ordering Information

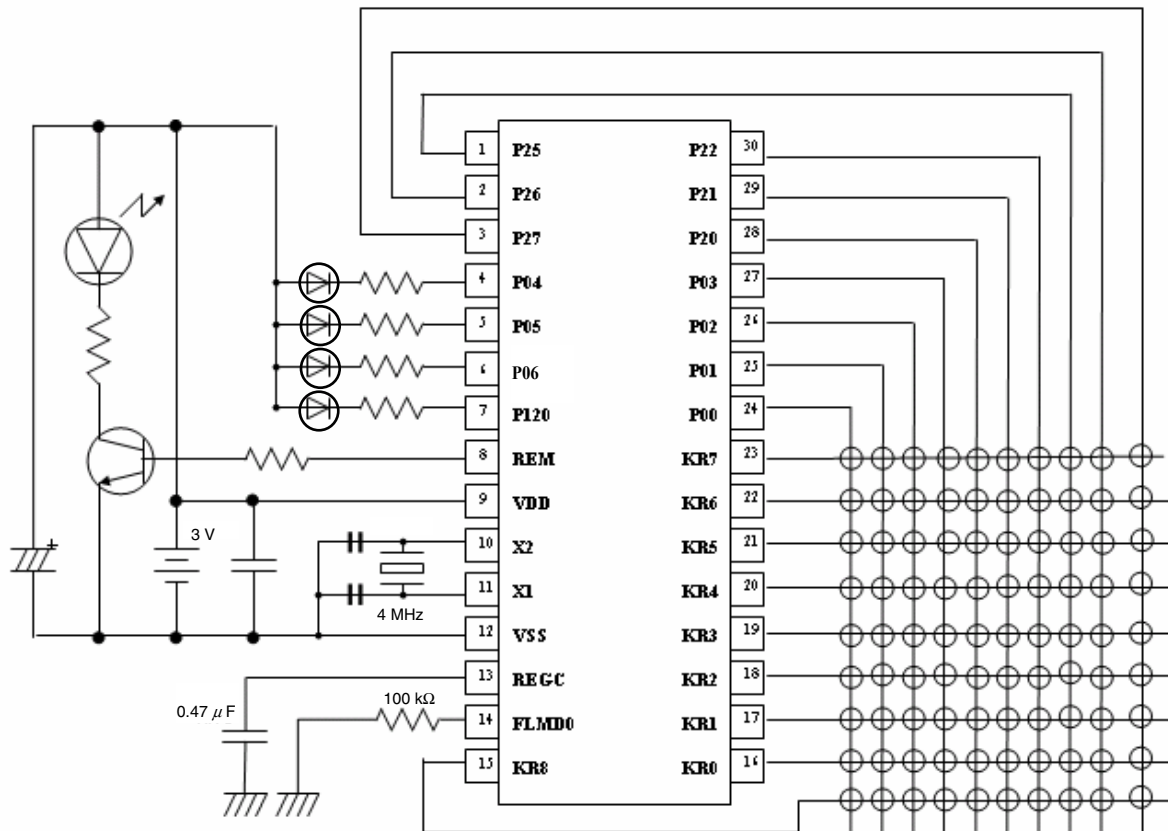
<R> • Flash memory version

Part Number	Package	Quality Grade
$\mu$ PD179F110MC-CAB-AX	30-pin plastic SSOP (7.62 mm (300))	Standard
$\mu$ PD179F111MC-CAB-AX	30-pin plastic SSOP (7.62 mm (300))	Standard
$\mu$ PD179F112MC-CAB-AX	30-pin plastic SSOP (7.62 mm (300))	Standard
$\mu$ PD179F113MC-CAB-AX	30-pin plastic SSOP (7.62 mm (300))	Standard
$\mu$ PD179F114MC-CAB-AX	30-pin plastic SSOP (7.62 mm (300))	Standard
$\mu$ PD179F122MC-GAA-AX	38-pin plastic SSOP (7.62 mm (300))	Standard
$\mu$ PD179F123MC-GAA-AX	38-pin plastic SSOP (7.62 mm (300))	Standard
$\mu$ PD179F124MC-GAA-AX	38-pin plastic SSOP (7.62 mm (300))	Standard

**Remark** Products with -AX at the end of the part number are lead-free products.

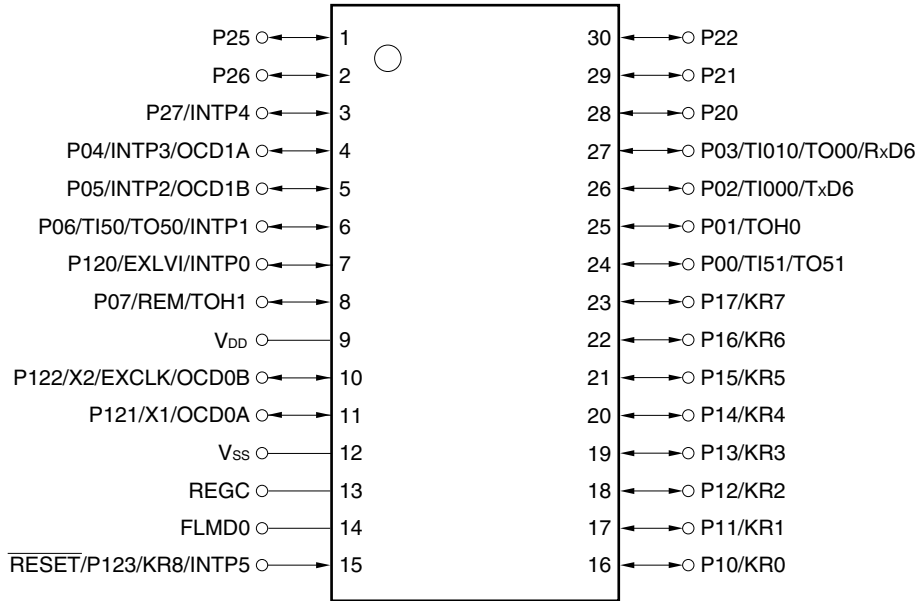
### 1.4 Application circuit

The following figure shows an example of the key matrix:  $9 \times 10 = 90$  keys. By using P04 to P06 and P120, the maximum of 126 keys can be configured (in the case of  $\mu$ PD179F11x microcontrollers).



### 1.5 Pin Configuration (Top View)

- 30-pin plastic SSOP (7.62 mm (300))

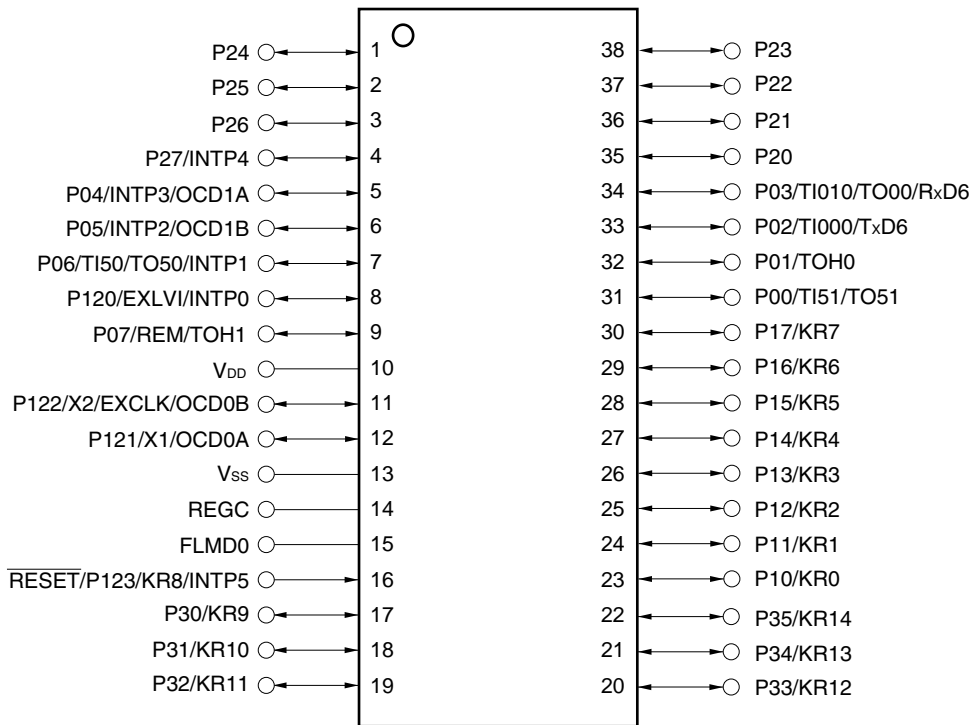


**Caution** Connect the REGC pin to V<sub>SS</sub> via a capacitor (0.47  $\mu$ F: recommended).

#### Pin Identification

EXCLK:	External Clock Input (Main System Clock)	TO00, TO50, TO51, TOH0, TOH1:	Timer Output
EXLVI:	External potential Input for Low-voltage detector	TxD6:	Transmit Data
FLMD0:	Flash Programming Mode	V <sub>DD</sub> :	Power Supply
INTP0 to INTP5:	External Interrupt Input	V <sub>SS</sub> :	Ground
KR0 to KR8:	Key Return	X1, X2:	Crystal Oscillator (Main System Clock)
OCD0A, OCD0B, OCD1A, OCD1B:	On Chip Debug Input/Output		
P00-P07:	Port 0		
P10-P17:	Port 1		
P20 to P22, P25 to P27:	Port 2		
P120 to P123:	Port 12		
REGC:	Regulator Capacitance		
REM:	Remote Control Output		
RESET:	Reset		
RxD6:	Receive Data		
TI000, TI010, TI50, TI51:	Timer Input		

- 38-pin plastic SSOP (7.62 mm (300))



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

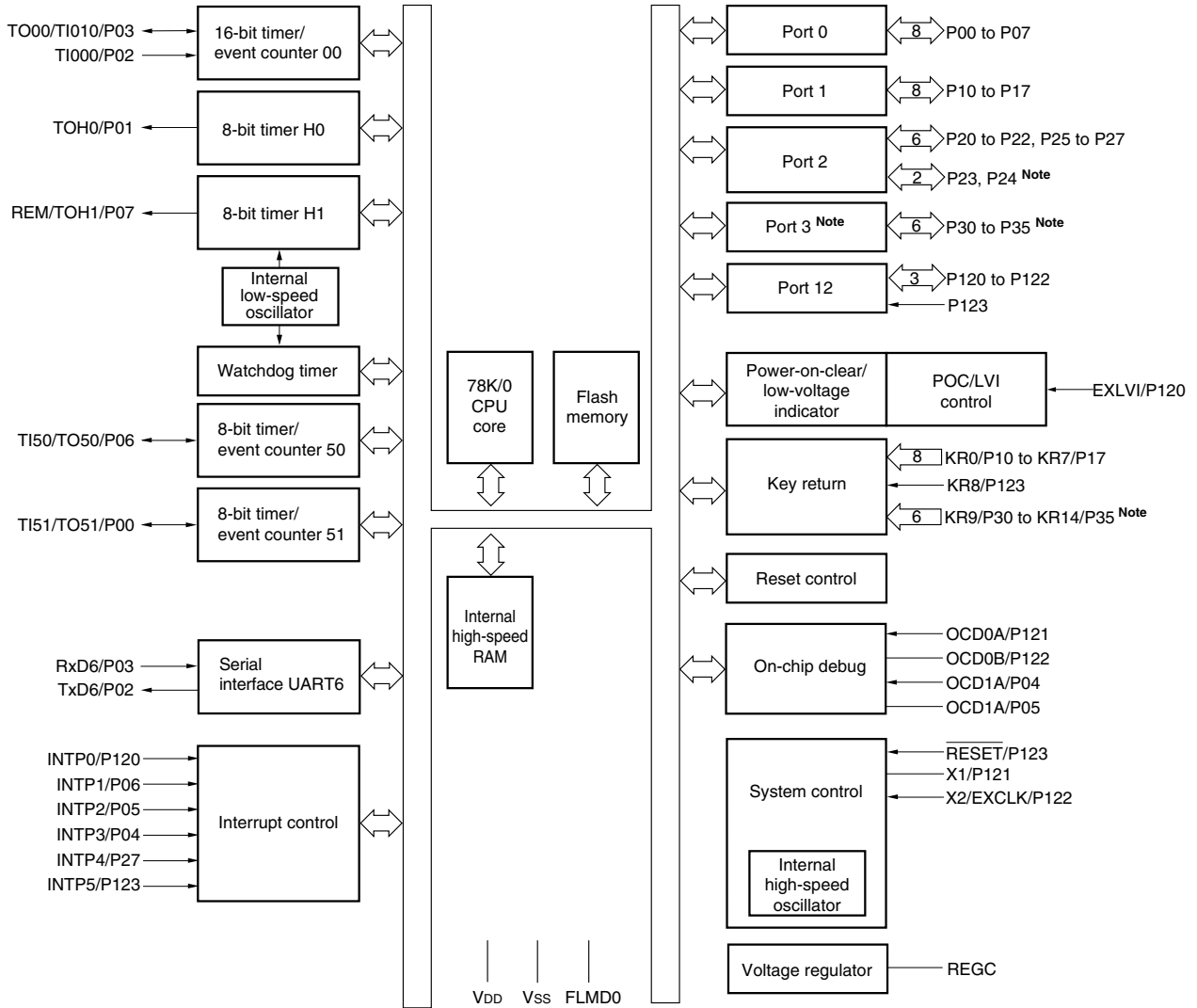
### Pin Identification

EXCLK:	External Clock Input (Main System Clock)	TO00, TO50, TO51, TOH0, TOH1:	Timer Output
EXLVI:	External potential Input for Low-voltage detector	TxD6:	Transmit Data
FLMD0:	Flash Programming Mode	V <sub>DD</sub> :	Power Supply
INTP0 to INTP5:	External Interrupt Input	V <sub>SS</sub> :	Ground
KR0 to KR14:	Key Return	X1, X2:	Crystal Oscillator (Main System Clock)
OCD0A, OCD0B, OCD1A, OCD1B:	On Chip Debug Input/Output		
P00 to P07:	Port 0		
P10 to P17:	Port 1		
P20 to P27:	Port 2		
P120 to P123:	Port 12		
REGC:	Regulator Capacitance		
REM:	Remote Control Output		
RESET:	Reset		
RxD6:	Receive Data		
TI000, TI010, TI50, TI51:	Timer Input		

1.6  $\mu$ PD179F11x, 179F12x Microcontrollers Lineup

ROM	RAM	30 Pins	38 Pins
32 KB	1 KB	$\mu$ PD179F114	$\mu$ PD179F114
24 KB		$\mu$ PD179F113	$\mu$ PD179F113
16 KB	768 B	$\mu$ PD179F112	$\mu$ PD179F112
8 KB	512 B	$\mu$ PD179F111	-
4 KB		$\mu$ PD179F110	-

1.7 Block Diagram



**Note** 38-pin products only

1.8 Outline of Functions

Item		μPD179F110	μPD179F111	μPD179F112	μPD179F113	μPD179F114	μPD179F122	μPD179F123	μPD179F124												
Internal memory (bytes)	Flash memory (self-programming supported) <sup>Note</sup>	4 K	8 K	16 K	24 K	32K	16 K	24 K	32K												
	High-speed RAM <sup>Note</sup>	512		768	1 K		768	1 K													
Memory space		64 KB																			
Main system clock (oscillation frequency)	High-speed system clock	X1 (crystal/ceramic) oscillation, external main system clock input (EXCLK) 4 MHz: V <sub>DD</sub> = 1.8 to 3.6 V																			
	Internal high-speed oscillation clock	Internal oscillation 4 MHz ± 2%: V <sub>DD</sub> = 1.8 to 3.6 V																			
Internal low-speed oscillation clock (for WDT)		Internal oscillation 240 kHz (TYP.): V <sub>DD</sub> = 1.8 to 3.6 V																			
General-purpose registers		8 bits × 32 registers (8 bits × 8 registers × 4 banks)																			
Minimum instruction execution time		<ul style="list-style-type: none"> <li>• 1 μs (V<sub>DD</sub> = 1.8 to 3.6 V, high-speed system clock: @ f<sub>XH</sub> = 2 or 4 MHz operation)</li> <li>• 0.5 μs (V<sub>DD</sub> = 2.0 to 3.6 V, high-speed system clock: @ f<sub>XH</sub> = 4 MHz operation)</li> </ul>																			
Instruction set		<ul style="list-style-type: none"> <li>• 8-/16-bit operation</li> <li>• Multiply/divide (8 bits × 8 bits, 16 bits ÷ 8 bits)</li> <li>• Bit manipulate (set, reset, test, and Boolean operation)</li> <li>• BCD adjust, etc.</li> </ul>																			
I/O ports		<table border="0" style="width: 100%;"> <tr> <td style="width: 50%;"><u>Total:</u></td> <td style="width: 25%; text-align: center;">26 (30-pin products)</td> <td style="width: 25%; text-align: center;">34 (38-pin products)</td> </tr> <tr> <td>N-ch open-drain output/CMOS I/O:</td> <td style="text-align: center;">24</td> <td style="text-align: center;">32</td> </tr> <tr> <td>P-ch open-drain output/CMOS I/O:</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> </tr> <tr> <td>CMOS Input:</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> </tr> </table>								<u>Total:</u>	26 (30-pin products)	34 (38-pin products)	N-ch open-drain output/CMOS I/O:	24	32	P-ch open-drain output/CMOS I/O:	1	1	CMOS Input:	1	1
<u>Total:</u>	26 (30-pin products)	34 (38-pin products)																			
N-ch open-drain output/CMOS I/O:	24	32																			
P-ch open-drain output/CMOS I/O:	1	1																			
CMOS Input:	1	1																			
Timers		<ul style="list-style-type: none"> <li>• 16-bit timer/event counter: 1 channel</li> <li>• 8-bit timer/event counter: 2 channels</li> <li>• 8-bit timer: 2 channels</li> <li>• Watchdog timer: 1 channel</li> </ul>																			
	Timer outputs	5 (PWM output: 2, carrier generator output for remote control: 1)																			
Serial interface		UART mode: 1 channel																			
Vectored interrupt sources	Internal	10																			
	External	8																			
Key interrupt		Key interrupt occurs by detecting falling edge of key input pins (KR0 and KR8).				Key interrupt occurs by detecting falling edge of key input pins (KR0 to KR14).															
Reset		<ul style="list-style-type: none"> <li>• Reset using <math>\overline{\text{RESET}}</math> pin</li> <li>• Internal reset by watchdog timer</li> <li>• Internal reset by power-on-clear</li> <li>• Internal reset by low-voltage detector</li> </ul>																			
On-chip debug function		Provided																			
Power supply voltage		V <sub>DD</sub> = 1.8 to 3.6 V																			
Operating ambient temperature		T <sub>A</sub> = -40 to +85°C																			
Package		30-pin plastic SSOP (7.62 mm (300))				38-pin plastic SSOP (7.62 mm (300))															

<R>

**Note** The internal flash memory capacity and internal high-speed RAM capacity can be changed using the internal memory size switching register (IMS).

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		Watchdog Timer
		TM00	TM50	TM51	TMH0	TMH1	
Function	Interval timer	1 channel	1 channel	1 channel	1 channel	1 channel	–
	External event counter	1 channel	1 channel	1 channel	–	–	–
	PWM output	–	–	–	1 output	1 output	–
	Pulse width measurement	2 inputs	–	–	–	–	–
	Square-wave output	1 output	1 output	1 output	–	–	–
	Carrier generator	–	–	<b>Note</b>	–	1 output <sup>Note</sup>	–
	Watchdog timer	–	–	–	–	–	1 channel
Interrupt source		2	1	1	1	1	–

**Note** TM51 and TMH1 can be used in combination as a carrier generator mode.

## CHAPTER 2 PIN FUNCTIONS

### 2.1 Pin Function List

#### (1) Port functions

Function Name	I/O	Function	After Reset	Alternate Function
P00	I/O	Port 0. 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 in 1-bit units. P00 to P06 can be set to N-ch open-drain output or CMOS I/O in 1-bit units. P07 can be set to P-ch open-drain output or CMOS I/O in 1-bit units. This pin can be used as a carrier generator output for remote control by specifying P-ch open-drain output.	Input port	TI51/TO51
P01				TOH0
P02				TI000/TxD6
P03				TI010/TO00/ RxD6
P04				INTP3/OCD1A
P05				INTP2/OCD1B
P06				TI50/TO50/INTP1
P07				REM/TOH1
P10 to P17	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 in 1-bit units. P10 to P17 can be set to N-ch open-drain output or CMOS I/O in 1-bit units.	Input port	KR0 to KR7
P20 to P22	I/O	Port 2. 38-pin products: 8-bit I/O port 30-pin products: 6-bit I/O port Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified in 1-bit units. P20 to P22, P23 <sup>Note</sup> , P24 <sup>Note</sup> , P25 to P27 can be set to N-ch open-drain output or CMOS I/O in 1-bit units.	Input port	–
P23 <sup>Note</sup> , P24 <sup>Note</sup>				–
P25, P26				–
P27				INTP4
P30 to P35 <sup>Note</sup>	I/O	Port 3 <sup>Note</sup> . 6-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified in 1-bit units. P30 to P35 can be set to N-ch open-drain output or CMOS I/O in 1-bit units.	Input port	KR9 to KR14 <sup>Note</sup>
P120	I/O	Port 12. 3-bit I/O port and 1-bit Input-only port. Input/output can be specified in 1-bit units in P120 to P122. Use of an on-chip pull-up resistor can be specified in only P120 and P123.	Input port	EXLVI/INTP0
P121				X1/OCD0A
P122				X2/EXCLK/OCD0B
P123	Input	P120 to P122 can be set to N-ch open-drain output or CMOS I/O in 1-bit units.		RESET/KR8/ INTP5

**Note** 38-pin products only



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

Function Name	I/O	Function	After Reset	Alternate Function
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				P06/TI50/TO50
INTP2				P05/OCD1B
INTP3				P04/OCD1A
INTP4				P27
INTP5				P123/ $\overline{\text{RESET}}$ /KR8
KR0 to KR7	Input	Key interrupt input	Input port	P10-P17
KR8				P123/ $\overline{\text{RESET}}$ /INTP5
KR9 to KR15 <sup>Note</sup>				P30 to P35 <sup>Note</sup>
REGC	–	Connecting regulator output stabilization capacitance for internal operation. Connect to V <sub>SS</sub> via a capacitor (0.47 $\mu$ F: recommended).	–	–
REM	Output	Remote control output	Input port	P07/TOH1
$\overline{\text{RESET}}$	Input	System reset input	Input port	P123/KR8/INTP5
RxD6	Input	Serial data input to asynchronous serial interface	Input port	P03/TI010/TO00
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	P02/TxD6
TI010		Capture trigger input to capture register (CR000) of 16-bit timer/event counter 00		P03/TO00/RxD6
TI50	Input	External count clock input to 8-bit timer/event counter 50	Input port	P06/TO50/INTP1
TI51		External count clock input to 8-bit timer/event counter 51		P00/TO51
TO00	Output	16-bit timer/event counter 00 output	Input port	P03/TI010/RxD6
TO50	Output	8-bit timer/event counter 50 output	Input port	P06/TI50/INTP1
TO51		8-bit timer/event counter 51 output		P00/TI51
TOH0	Output	8-bit timer H0 output	Input port	P01
TOH1		8-bit timer H1 output		P07/REM
TxD6	Output	Serial data output from asynchronous serial interface	Input port	P02/TI000
X1	Input	Connecting resonator for main system clock	Input port	P121/OCD0A
X2	–			P122/EXCLK/OCD0B
EXCLK	Input	External clock input for main system clock	Input port	P122/X2/OCD0B
V <sub>DD</sub>	–	Positive power supply	–	–
V <sub>SS</sub>	–	Ground potential	–	–
OCD0A	Input	Connection for on-chip debug mode setting pins	Input port	P121/X1
OCD1A				P04/INTP3
OCD0B	–			P122/X2/EXCLK
OCD1B				P05/INTP2

**Note** 38-pin products only

## 2.2 Description of Pin Functions

### 2.2.1 P00 to P07 (port 0)

P00 to P07 function as an 8-bit I/O port. These pins also function as timer I/O, serial data I/O, external interrupt request input, remote control output, and the setting connection for on-chip debug mode.

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

#### (1) Port mode

P00 to P07 function as an 8-bit I/O port. P00 to P07 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).

P00 to P06 can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 0 (POM0).

P07 can be set to P-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 0 (POM0). This pin can be used as a carrier generator output for remote control by specifying P-ch open-drain output.

#### (2) Control mode

P00 to P07 function as timer I/O, serial data I/O, external interrupt request input, remote control output, and the setting connection for on-chip debug mode.

##### (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) of 16-bit timer/event counter 00.

##### (c) TO00

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

##### (d) RxD6

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

##### (e) TxD6

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

##### (f) TI50, TI51

These are the pins for inputting an external count clock to 8-bit timer/event counters 50 and 51.

##### (g) TO50, TO51

These are the timer output pins of 8-bit timer/event counters 50 and 51.

##### (h) TOH0, TOH1

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

**(i) INTP1 to INTP3**

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

**(j) REM**

This is a pin for outputting remote control signal.

**(k) OCD1A, OCD1B**

These are the setting connection pins for on-chip debug mode.

**2.2.2 P10 to P17 (port 1)**

P10 to P17 function as an 8-bit I/O port. These pins also function as pins for key interrupt input.

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

**(1) Port mode**

P10 to P17 function as an 8-bit 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). P10 to P17 can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 1 (POM1).

**(2) Control mode****(a) KR0 to KR7**

P10 to P17 function as key interrupt input.

**2.2.3 P20 to P27 (port 2)**

P20 to P27 function as an 8-bit I/O port in the case of 38-pin products. P20 to P22, P25 to P27 function as a 6-bit I/O port in the case of 30-pin products. These pins also function as pins for external interrupt request input.

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

**(1) Port mode**

P20 to P27 function as an 8-bit I/O port in the case of 38-pin products. P20 to P22, P25 to P27 function as a 6-bit I/O port in the case of 30-pin products. These pins can be set to input or output port in 1-bit units using port mode register 2 (PM2). Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 2 (PU2). These pins can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 2 (POM2).

**(2) Control mode (P27 only)****(a) INTP4**

P27 functions as External interrupt request input pin.

**Caution** For the 30-pin products, be sure to set bits 3 and 4 of PM2 to “1”, and bits 3 and 4 of P2, PU2, and POM2 to “0”.

**Remark** P23 and P24 are provided to the 38-pin products only.

### 2.2.4 P30 to P35 (port 3) (38-pin products only)

P30 to P35 function as a 6-bit I/O port. These pins also function as pins for external interrupt request input and timer I/O.

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

#### (1) Port mode

P30 to P35 function as a 6-bit I/O port. P30 to P35 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). P30 to P35 can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 3 (POM3).

#### (2) Control mode

##### (a) KR9 to KR14

P30 to P35 function as key interrupt input.

**Caution** For the 30-pin products, be sure to set bits 0 to 5 of PM3 to “1”, and bits 0 to 5 of P3, PU3, and POM3 to “0”.

### 2.2.5 P120 to P123 (port 12)

P120 to P122 function as a 3-bit I/O port. P123 functions as an input-only 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, system reset input, key interrupt input, and the setting connection for on-chip debug mode.

When using the P123/ $\overline{\text{RESET}}$ /INTP5/KR8 pin for P123, INTP5, or KR8, set bit 3 (RSTM) of reset pin mode register (RSTMASK) to “1”.

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

#### (1) Port mode

P120 to P122 function as a 3-bit I/O port. P123 functions as an input-only port. P120 to P122 can be set to input or output port in 1-bit units using port mode register 12 (PM12). Only for P120 and P123, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 12 (PU12). P120 to P122 can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 12 (POM12).

#### (2) Control mode

P120 to P123 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, system reset input, key interrupt input, and the setting connection for on-chip debug mode.

##### (a) INTP0, INTP5

These are external interrupt request input pins 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) KR8**

This is a key interrupt input pin.

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

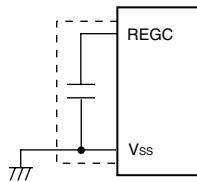
This is the active-low system reset input pin.

**(g) OCD0A, OCD0B**

These are the setting connection pins for on-chip debug mode.

**2.2.6 REGC**

This is the pin for connecting regulator output (2.0 V) stabilization capacitance for internal operation. Connect this pin to  $V_{SS}$  via a capacitor (0.47  $\mu\text{F}$ : recommended).



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

**2.2.7  $V_{DD}$** 

$V_{DD}$  is the positive power supply pin.

**2.2.8  $V_{SS}$** 

$V_{SS}$  is the ground potential pin.

**2.2.9 FLMD0**

This is a pin for setting flash memory programming mode. Pull the FLMD0 pin down by a resistor (100 k $\Omega$ ) when executing the on-board programming or self-programming.

Use FLMD0 at the same level as  $V_{SS}$  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 2-2** 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 2-1. Pin I/O Circuit Types

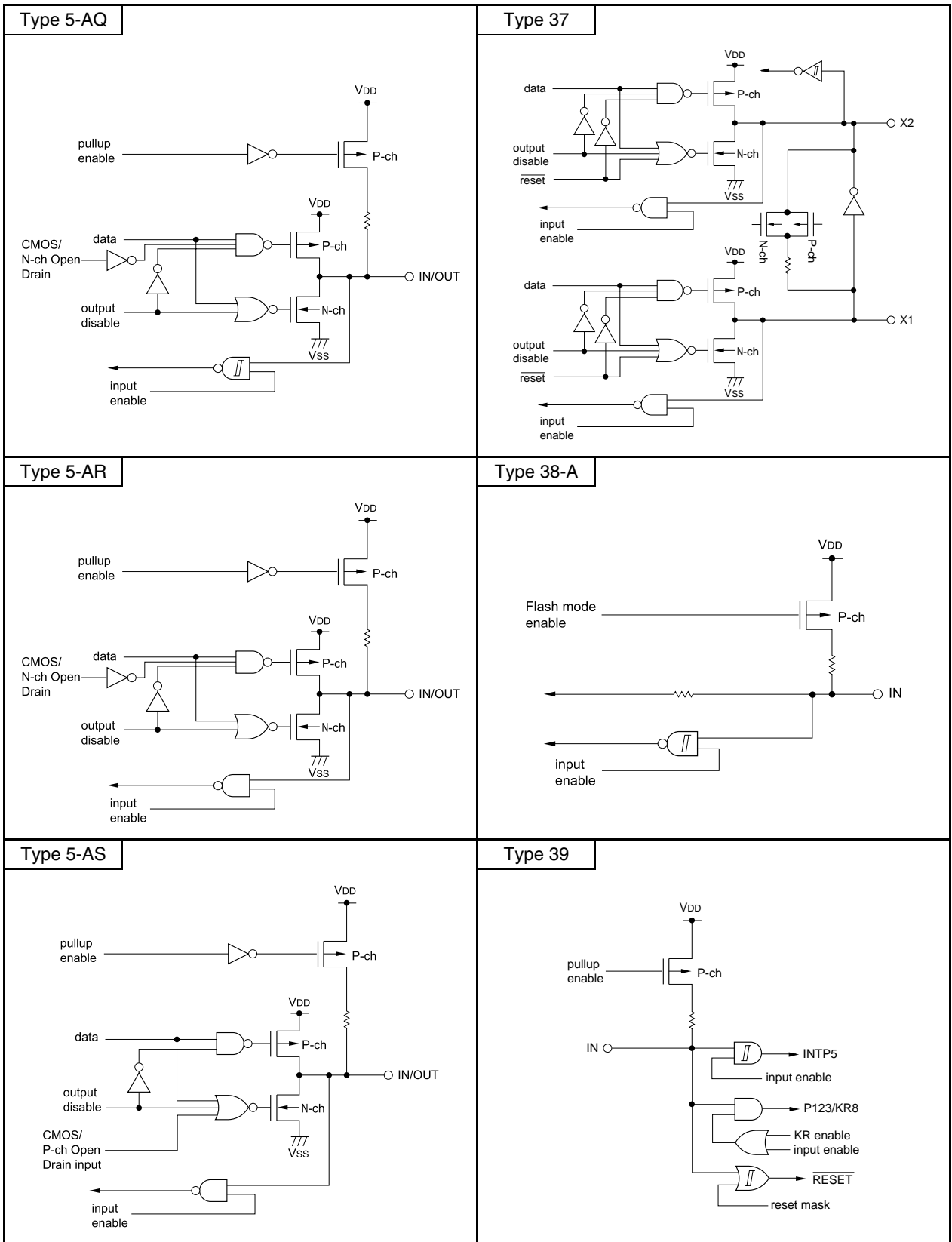
<R>

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P00/TI51/TO51	5-AQ	I/O	Input: Independently connect to V <sub>DD</sub> or V <sub>SS</sub> via a resistor. Output: Leave open.
P01/TOH0	5-AR		
P02/TI000/TxD6	5-AQ		
P03/TI010/TO00/RxD6			
P04/INTP3/OCD1A			
P05/INTP2/OCD1B			
P06/TI50/TO50/INTP1			
P07/REM/TOH1			
P10/KR0	5-AR		
P11/KR1			
P12/KR2			
P13/KR3			
P14/KR4			
P15/KR5			
P16/KR6			
P17/KR7			
P20			
P21			
P22			
P23 <sup>Note 1</sup>			
P24 <sup>Note 1</sup>		5-AQ	
P25			
P26			
P27/INTP4			
P30/KR9 <sup>Note 1</sup>	5-AR		
P31/KR10 <sup>Note 1</sup>			
P32/KR11 <sup>Note 1</sup>			
P33/KR12 <sup>Note 1</sup>			
P34/KR13 <sup>Note 1</sup>			
P35/KR14 <sup>Note 1</sup>			
P120/EXLVI/INTP0	5-AQ		
P121/X1/OCD0A	37		Input: Independently connect to V <sub>DD</sub> or V <sub>SS</sub> via a resistor. Output: Leave open.
P122/X2/EXCLK/OCD0B			
P123/RESET/KR8/INTP5	39	Input	Connect to V <sub>DD</sub> directly or via a resistor.
FLMD0	38-A	–	Connect to V <sub>SS</sub> <sup>Note 2</sup> .

**Notes 1.** P23, P24, P30 to P35 are provided to 38-pin products only.

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<sub>SS</sub> via a resistor (100 kΩ: recommended).

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



## CHAPTER 3 CPU ARCHITECTURE

### 3.1 Memory Space

Products in the  $\mu$ PD179F11x, 179F12x microcontrollers can access a 64 KB memory space. Figures 3-1 to 3-5 show the memory maps.

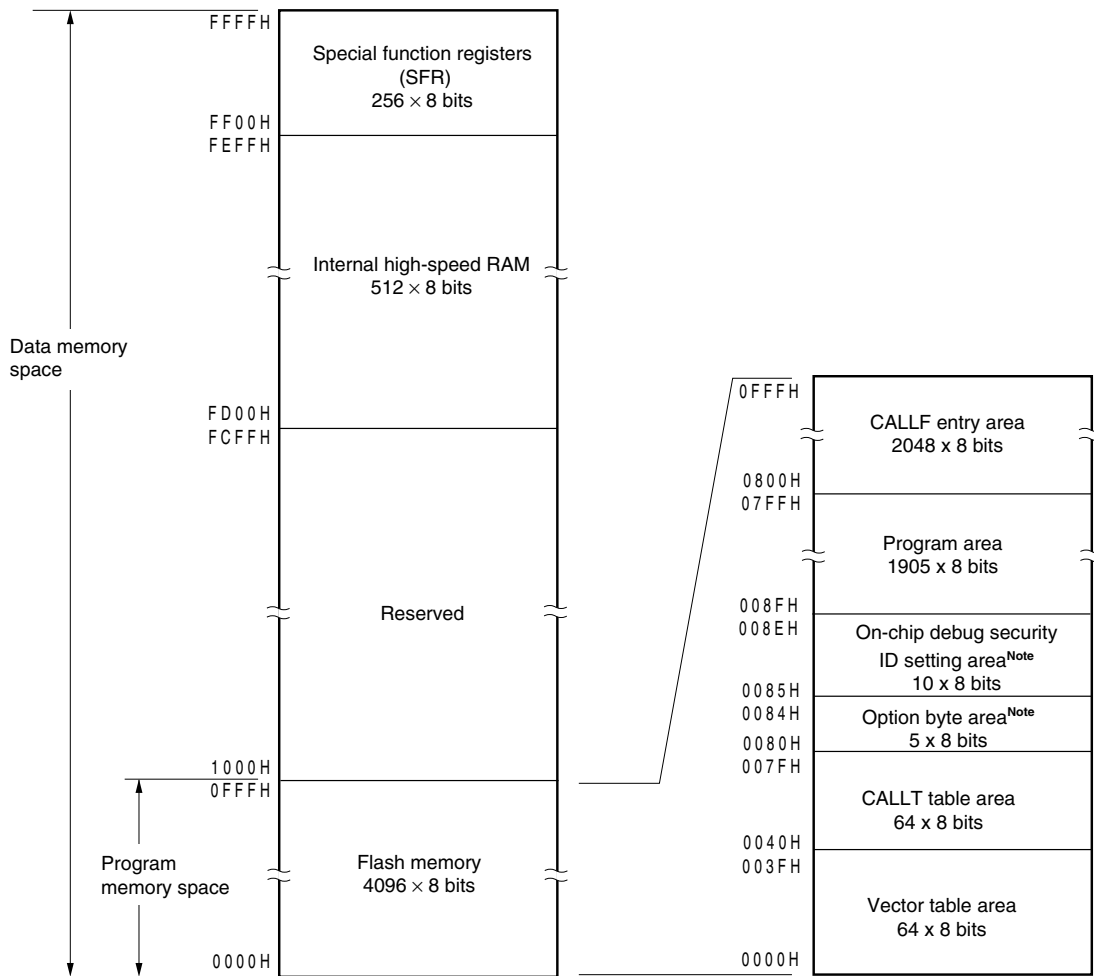
**Caution** Regardless of the internal memory capacity, the initial value of the internal memory size switching register (IMS) of all products in the  $\mu$ PD179F11x, 179F12x microcontrollers is fixed (IMS = CFH). Therefore, set the value corresponding to each product as indicated below.

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

Flash Memory Version ( $\mu$ PD179F11x, 179F12x microcontrollers)	IMS	ROM Capacity	Internal High-Speed RAM Capacity
$\mu$ PD179F110	41H	4 KB	512 bytes
$\mu$ PD179F111	42H	8 KB	
$\mu$ PD179F112, 179F122	04H	16 KB	768 bytes
$\mu$ PD179F113, 179F123	C6H	24 KB	1 KB
$\mu$ PD179F114, 179F124	C8H	32 KB	



Figure 3-1. Memory Map ( $\mu$ PD179F110)

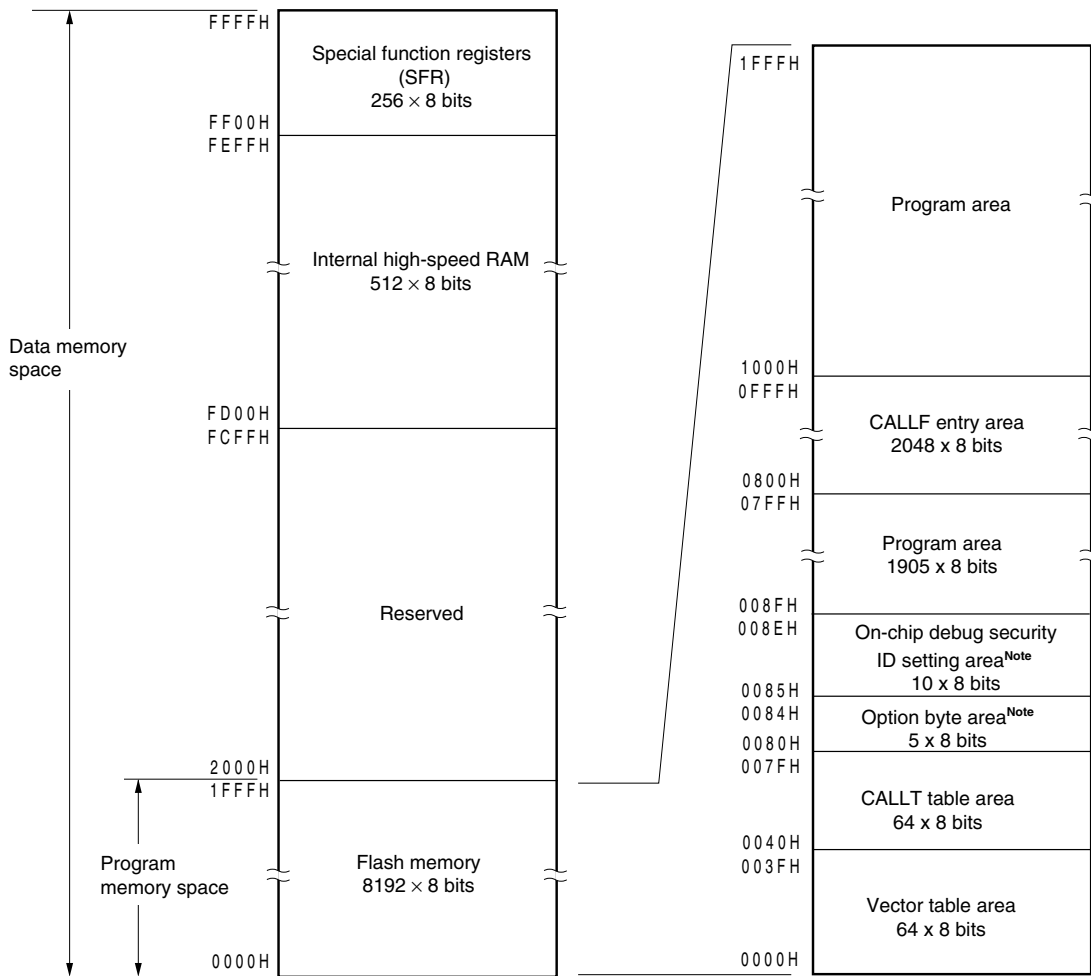


**Note** Set the option bytes to 0080H to 0084H.  
Set the on-chip debug security ID to 0085H to 008EH.

**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.**

0FFFH	Block 03H	1 KB
0C00H 0BFFH	Block 02H	
07FFH	Block 01H	
0400H 03FFH	Block 00H	
0000H		

Figure 3-2. Memory Map ( $\mu$ PD179F111)



**Note** Set the option bytes to 0080H to 0084H.  
Set the on-chip debug security ID to 0085H to 008EH.

**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**.

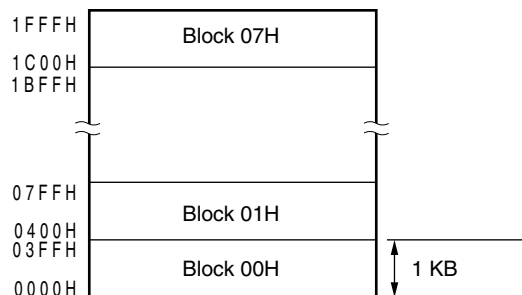
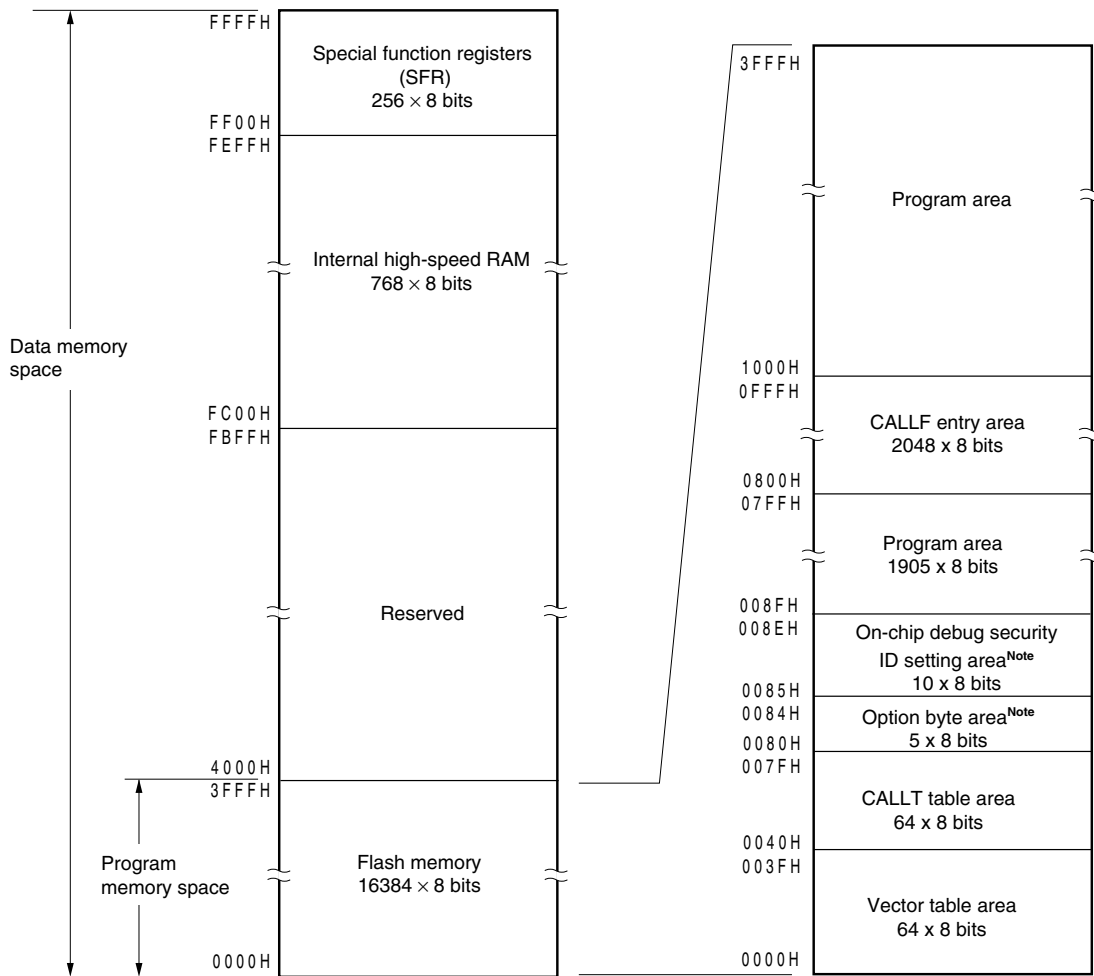


Figure 3-3. Memory Map ( $\mu$ PD179F112, 179F122)



**Note** Set the option bytes to 0080H to 0084H.  
Set the on-chip debug security ID to 0085H to 008EH.

**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**.

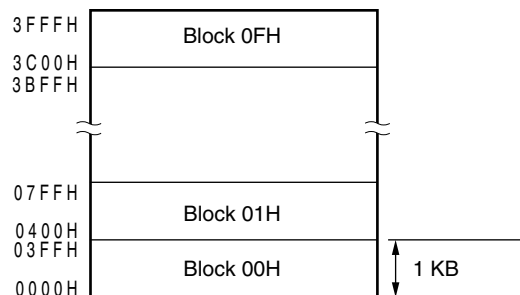
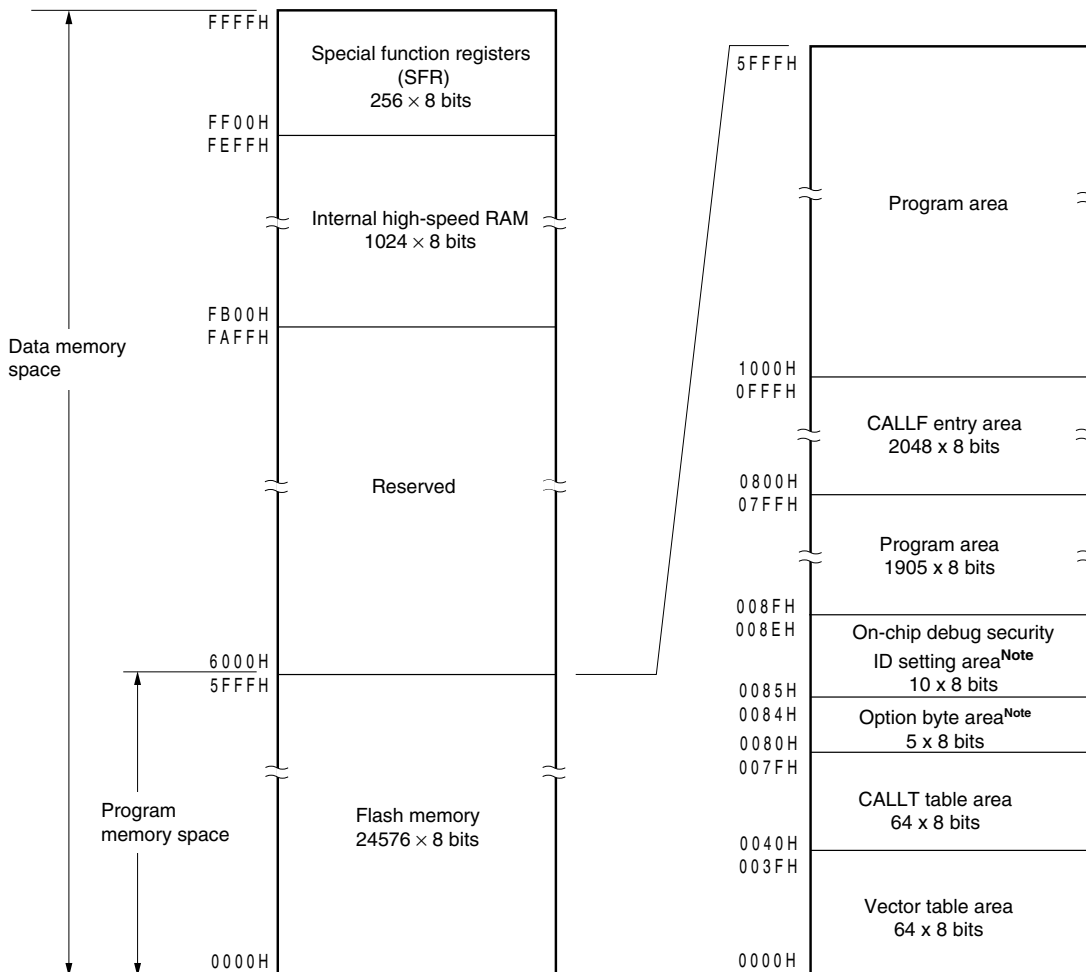


Figure 3-4. Memory Map ( $\mu$ PD179F113, 179F123)



**Note** Set the option bytes to 0080H to 0084H.  
Set the on-chip debug security ID to 0085H to 008EH.

**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**.

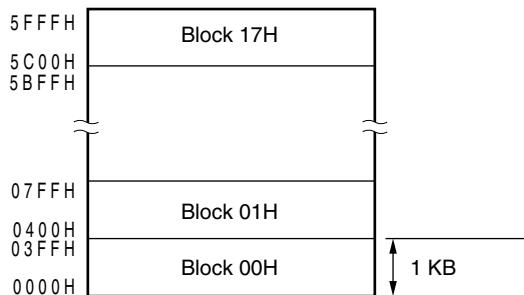
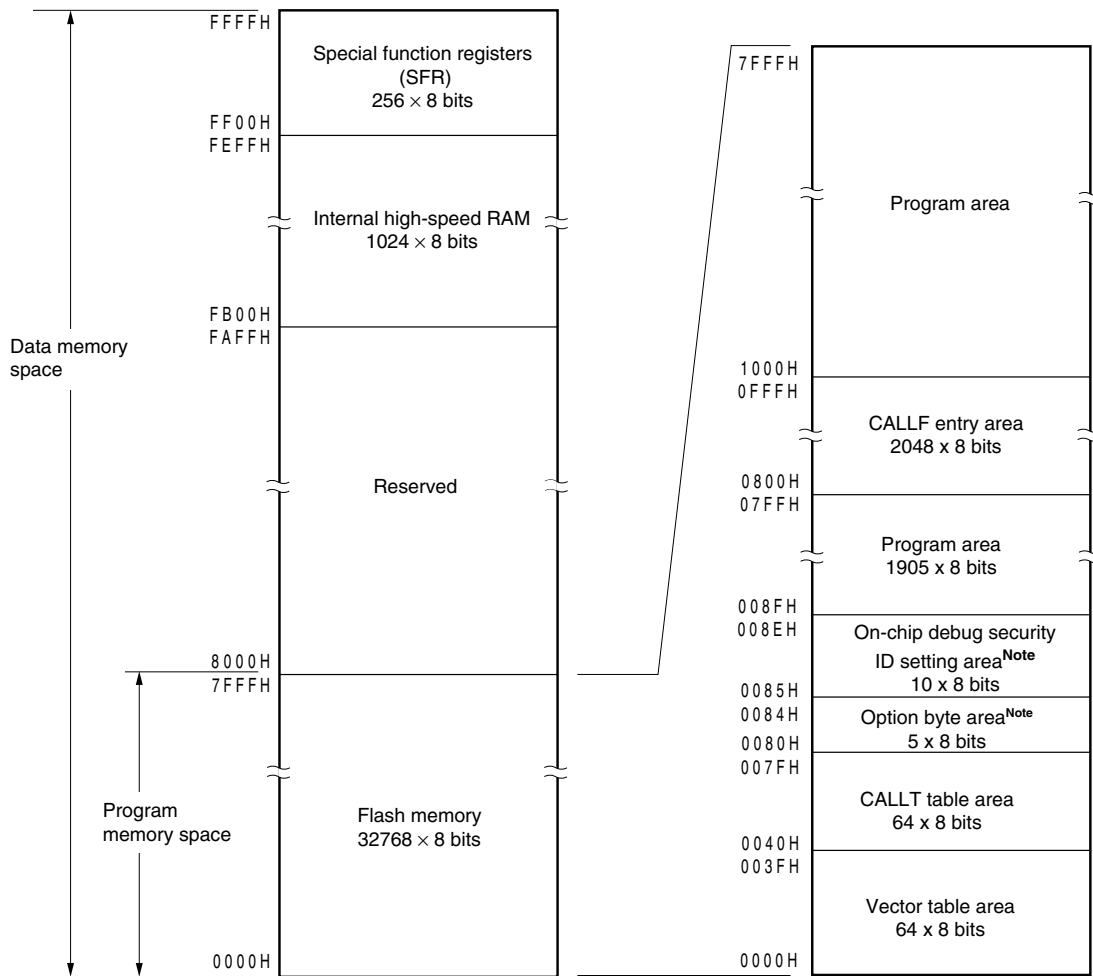
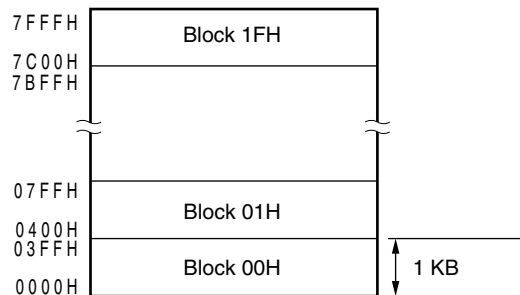


Figure 3-5. Memory Map ( $\mu$ PD179F114, 179F124)



**Note** Set the option bytes to 0080H to 0084H.  
Set the on-chip debug security ID to 0085H to 008EH.

**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**  $\mu$ PD179F110: Block numbers 00H to 03H  
 $\mu$ PD179F111: Block numbers 00H to 07H  
 $\mu$ PD179F112, 179F122: Block numbers 00H to 0FH  
 $\mu$ PD179F113, 179F123: Block numbers 00H to 17H  
 $\mu$ PD179F114, 179F124: 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).

$\mu$ PD179F11x, 179F12x microcontrollers products incorporate internal ROM (flash memory), as shown below.

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

Part Number	Internal ROM	
	Structure	Capacity
$\mu$ PD179F110	Flash memory	4096 $\times$ 8 bits (0000H to 0FFFH)
$\mu$ PD179F111		8192 $\times$ 8 bits (0000H to 1FFFH)
$\mu$ PD179F112, 179F122		16384 $\times$ 8 bits (0000H to 3FFFH)
$\mu$ PD179F113, 179F123		24576 $\times$ 8 bits (0000H to 5FFFH)
$\mu$ PD179F114, 179F124		32768 $\times$ 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 Address	Interrupt Source	Vector Table Address	Interrupt Source
0000H	RESET input, POC, LVI, WDT	0016H	INTST6
0004H	INTLVI	001AH	INTTMH1
0006H	INTP0	001CH	INTTMH0
0008H	INTP1	001EH	INTTM50
000AH	INTP2	0020H	INTTM000
000CH	INTP3	0022H	INTTM010
000EH	INTP4	002AH	INTTM51
0010H	INTP5	002CH	INTKR
0012H	INTSRE6	002EH	INTWT
0014H	INTSR6	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. For details, see **CHAPTER 17 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).

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

A 10-byte area of 0085H to 008EH can be used as an on-chip debug security ID setting area. For details, see **CHAPTER 19 ON-CHIP DEBUG FUNCTION**.

**3.1.2 Internal data memory space**

$\mu$ PD179F11x, 179F12x microcontrollers incorporate the following RAMs.

**(1) Internal high-speed RAM****Table 3-5. Internal High-Speed RAM Capacity**

Part Number	Internal High-Speed RAM
$\mu$ PD179F110	512 $\times$ 8 bits (FD00H to FEFH)
$\mu$ PD179F111	
$\mu$ PD179F112, 179F122	768 $\times$ 8 bits (FC00H to FEFH)
$\mu$ PD179F113, 179F123	1024 $\times$ 8 bits (FB00H to FEFH)
$\mu$ PD179F114, 179F124	

The 32-byte area FEE0H to FEFH 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.

**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  $\mu$ PD179F11x, 179F12x microcontrollers, 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 3-6 to 3-10 show correspondence between data memory and addressing. For details of each addressing mode, see **3.4 Operand Address Addressing**.



Figure 3-6. Correspondence Between Data Memory and Addressing ( $\mu$ PD179F110)

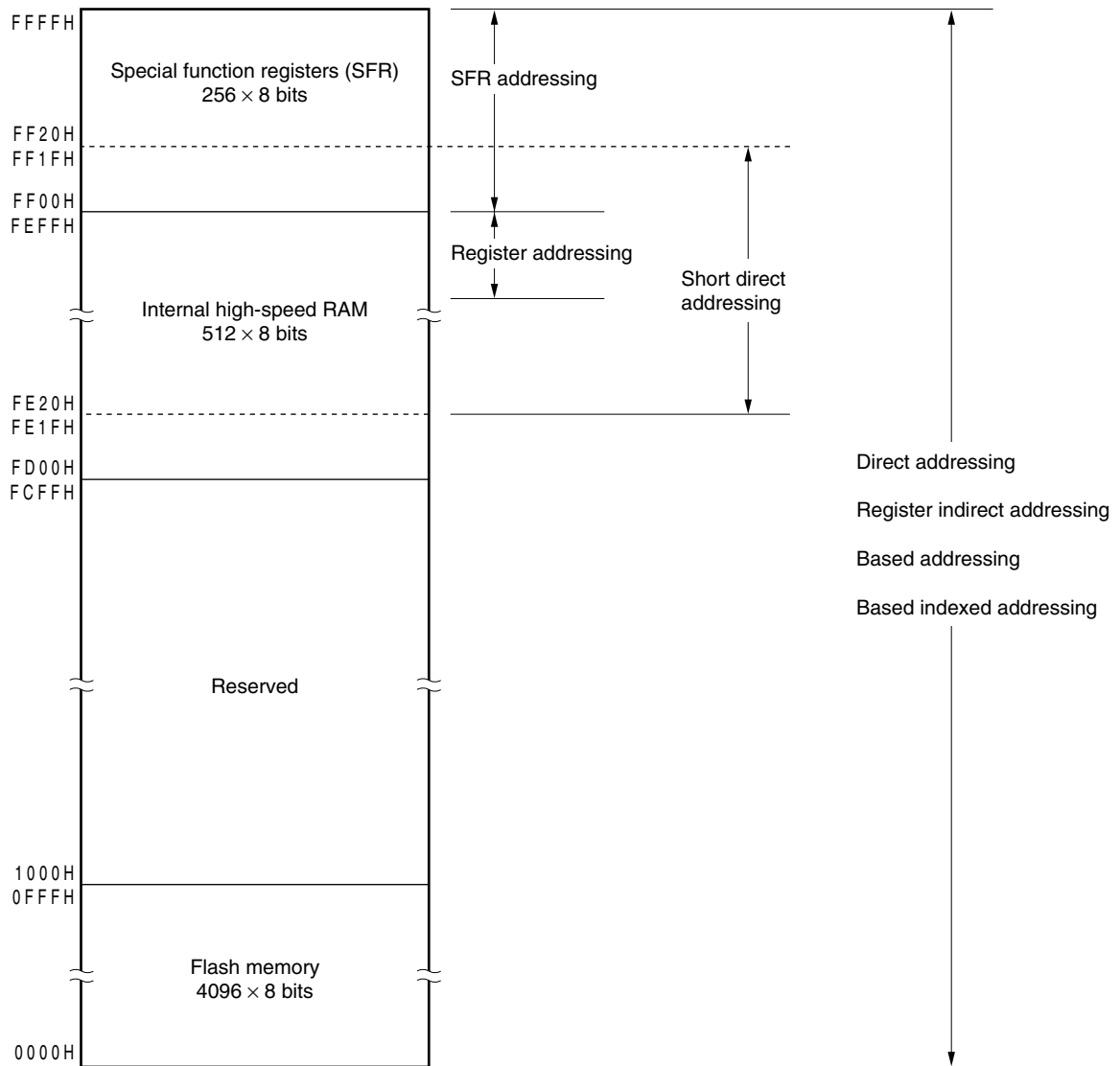


Figure 3-7. Correspondence Between Data Memory and Addressing ( $\mu$ PD179F111)

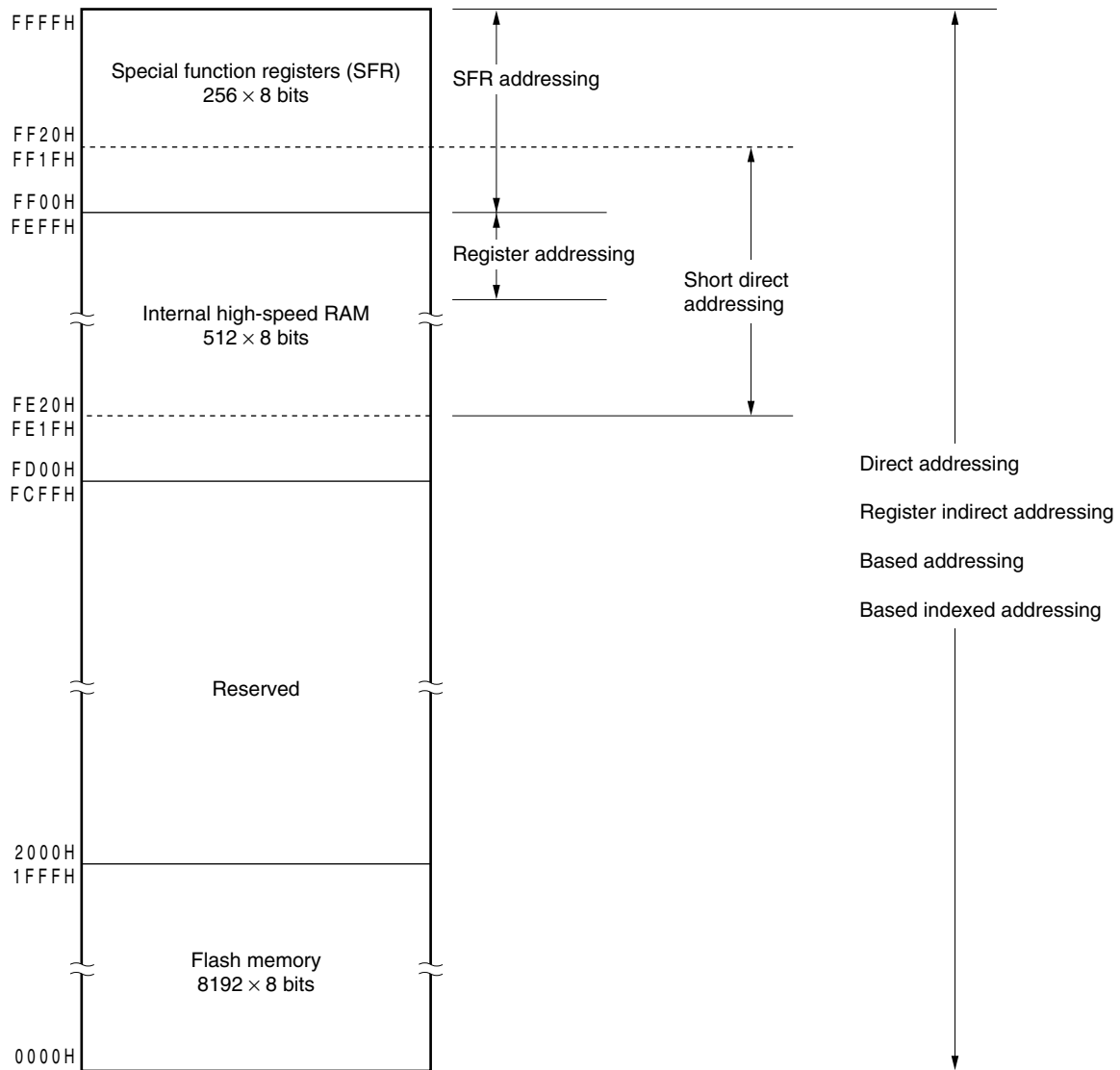


Figure 3-8. Correspondence Between Data Memory and Addressing ( $\mu$ PD179F112 and 179F122)

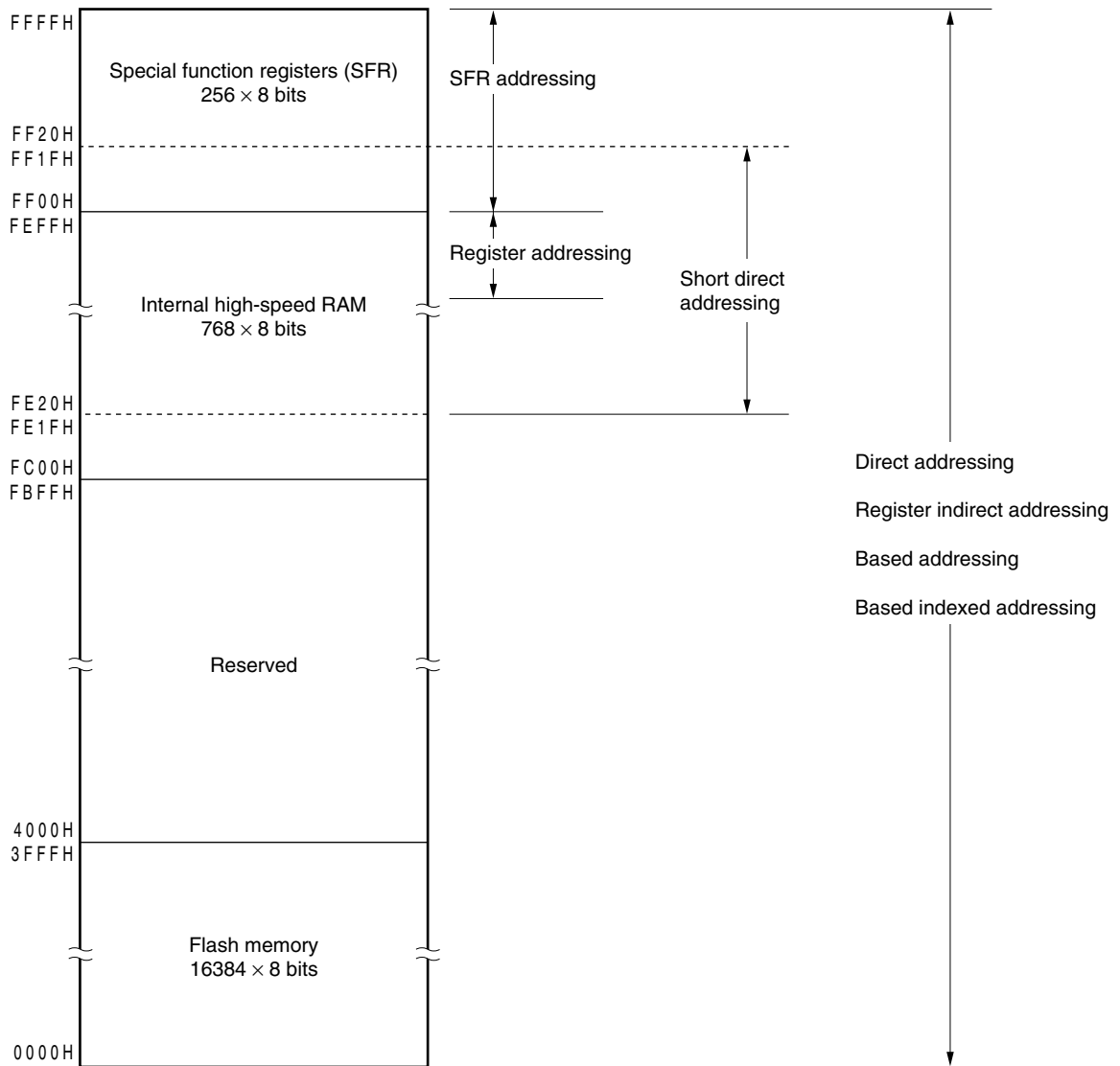


Figure 3-9 . Correspondence Between Data Memory and Addressing ( $\mu$ PD179F113 and 179F123)

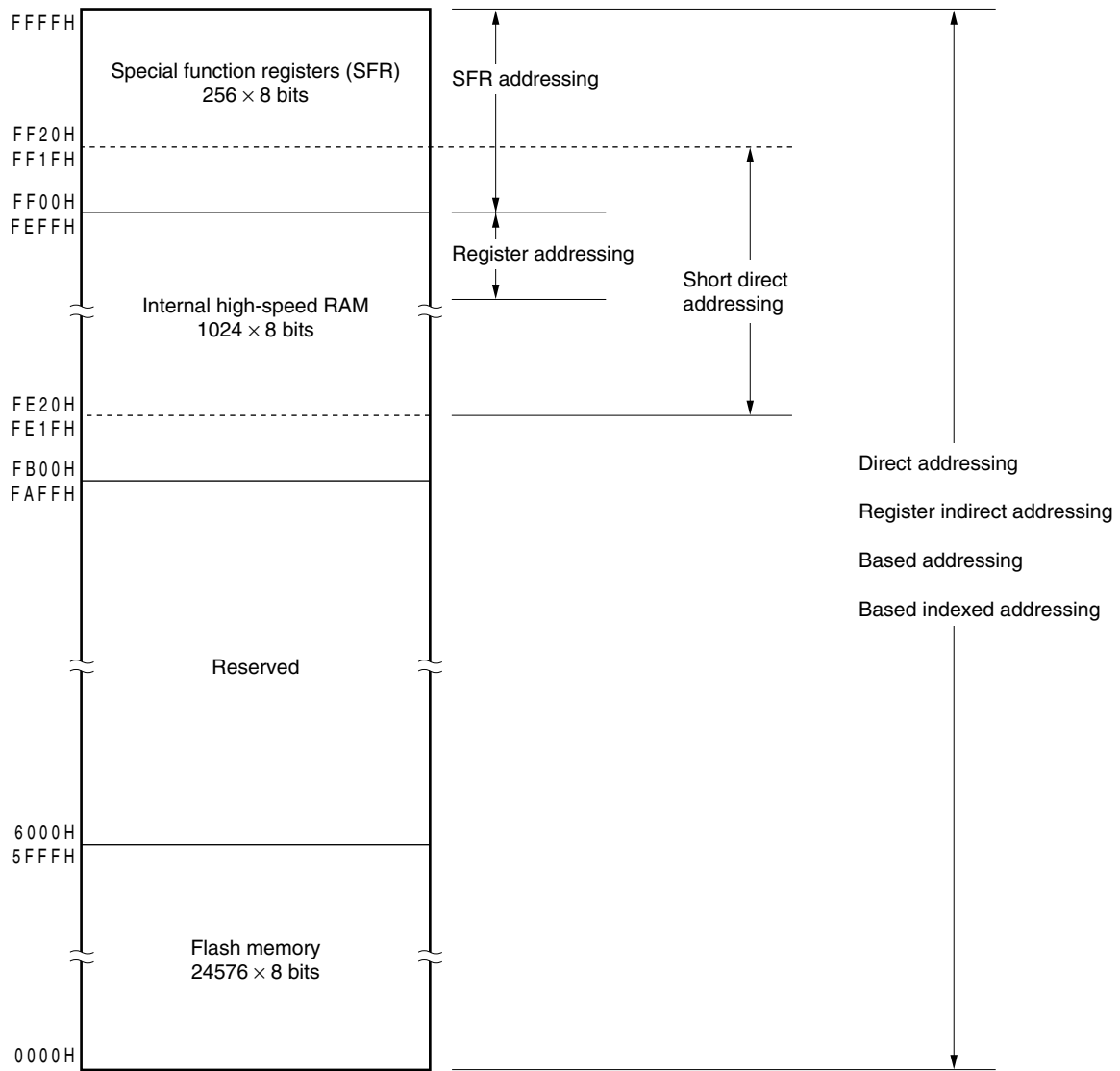
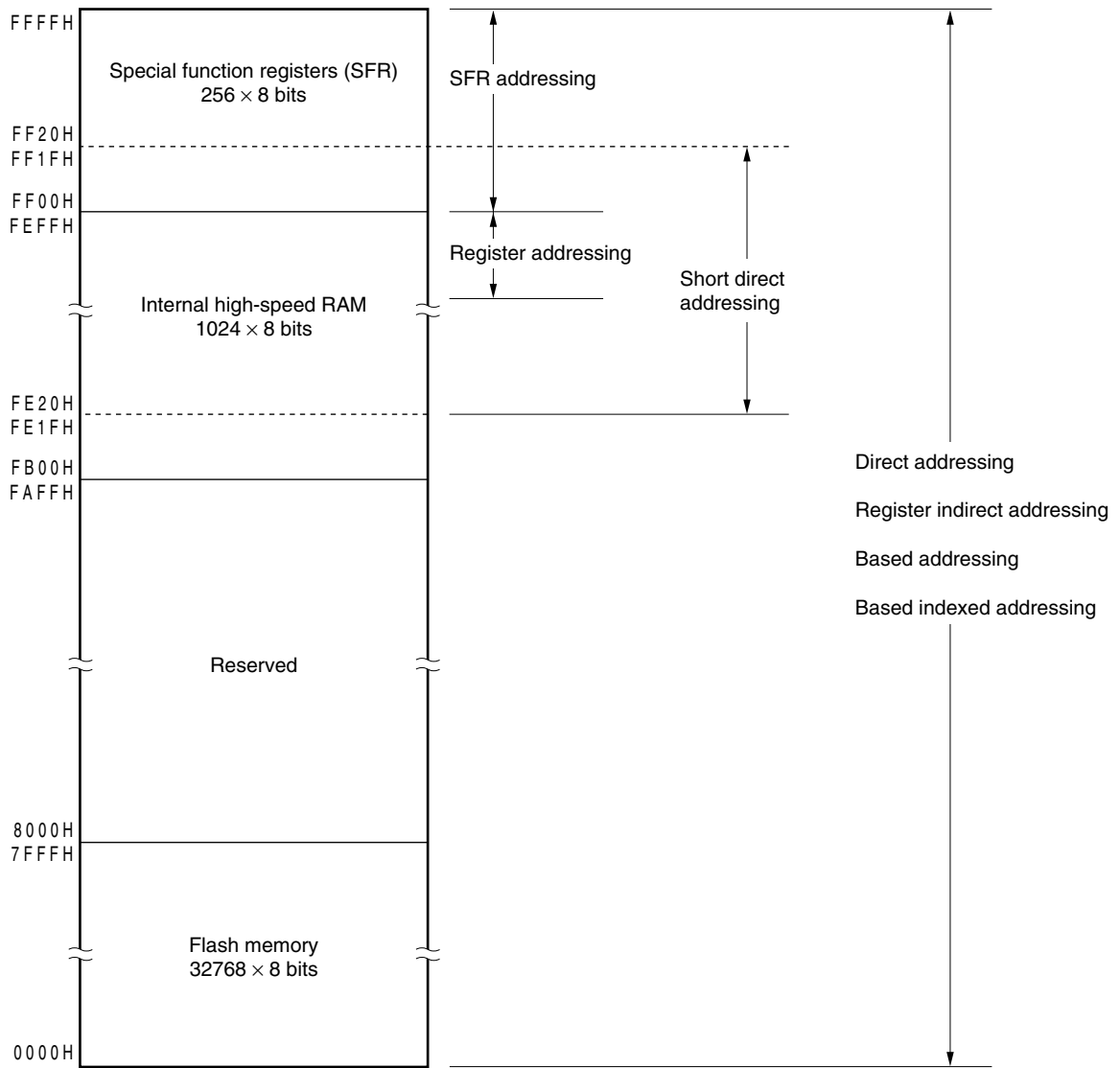


Figure 3-10. Correspondence Between Data Memory and Addressing ( $\mu$ PD179F114 and 179F124)



### 3.2 Processor Registers

The  $\mu$ PD179F11x, 179F12x microcontrollers 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).

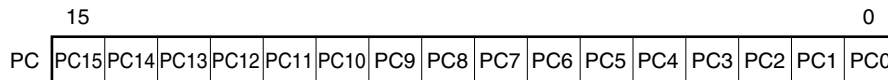
##### (1) Program counter (PC)

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 3-11. Format of Program Counter**



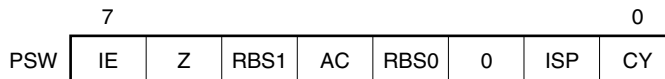
##### (2) Program status word (PSW)

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 interrupt request generation 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 3-12. 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) (see 11.3 (3) Priority specification flag registers (PR0L, PR0H, PR1L)) can not be acknowledged. Actual request acknowledgment is controlled by the interrupt enable flag (IE).

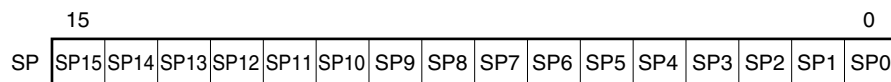
**(f) Carry flag (CY)**

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 3-13. Format of Stack Pointer**



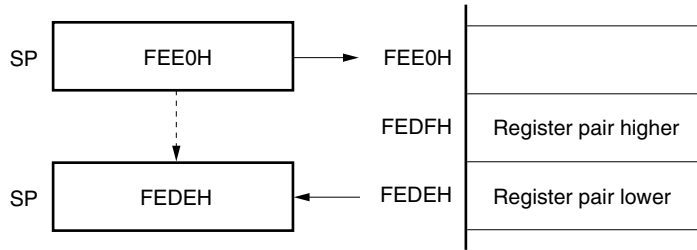
The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restored) from the stack memory.

Each stack operation saves/restores data as shown in Figures 3-14 and 3-15.

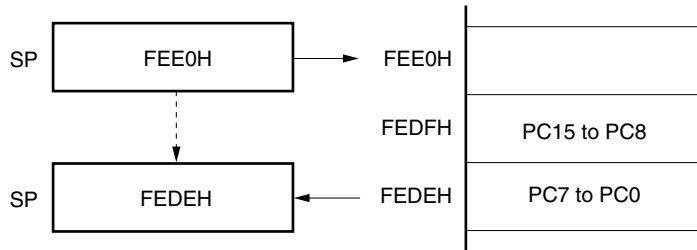
**Caution** Since reset signal generation makes the SP contents undefined, be sure to initialize the SP before using the stack.

Figure 3-14. 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)

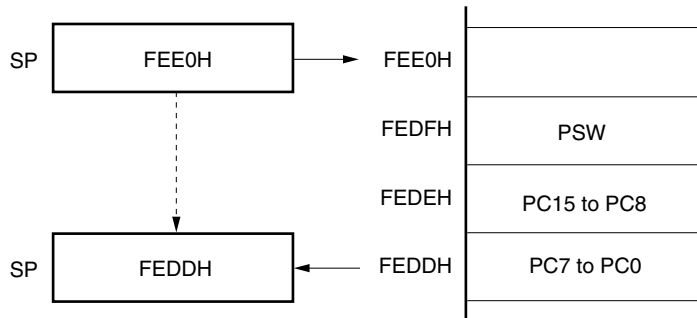
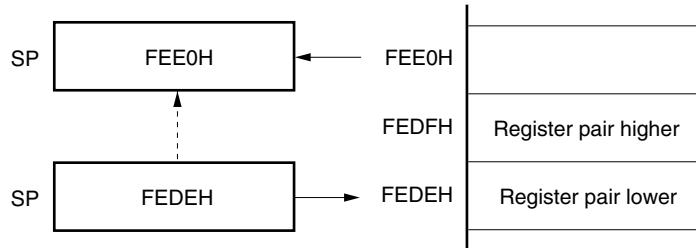


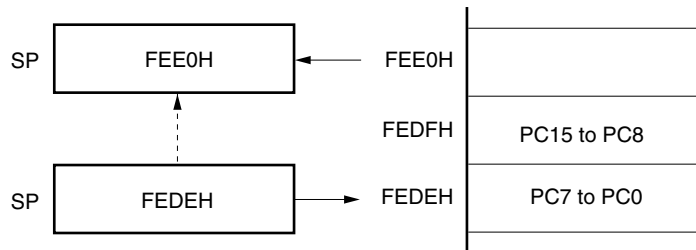


Figure 3-15. Data to Be Restored from Stack Memory

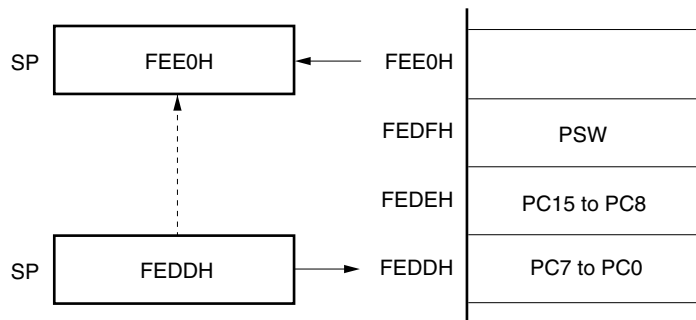
(a) POP rp instruction (when SP = FEDEH)



(b) RET instruction (when SP = FEDEH)



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



### 3.2.2 General-purpose registers

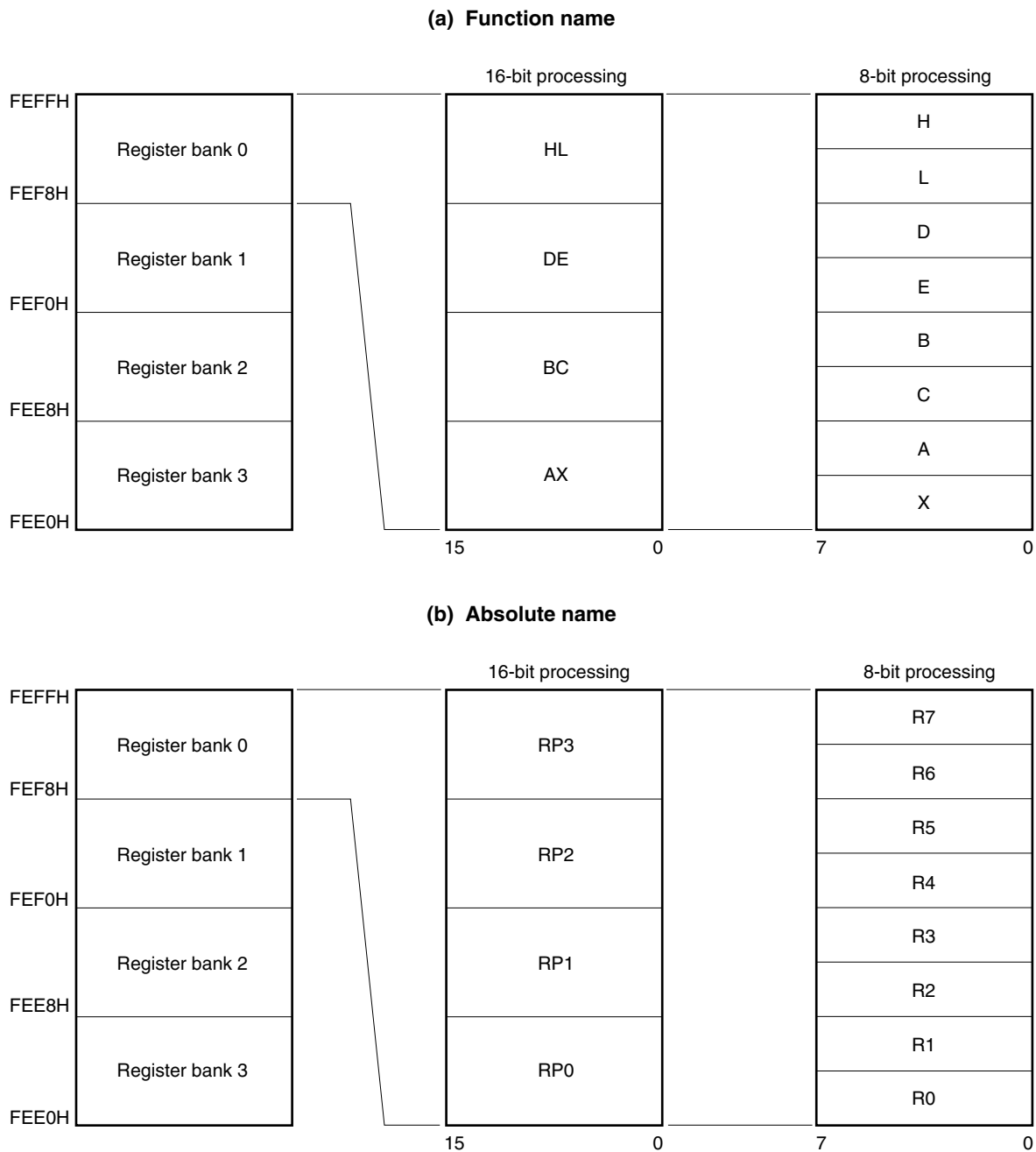
General-purpose registers are mapped at particular addresses (FEE0H to FEFFH) of the data memory. The general-purpose registers consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H).

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 3-16. Configuration of General-Purpose Registers**



### 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, ID78K0-QB, and SM+ for 78K0/KX2, 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 (1/3)

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
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
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
FF2CH	Port mode register 12	PM12	R/W	√	√	–	FFH
FF2EH	Reset pin mode register	RSTMASK	R/W	√	√	–	00H
FF30H	Pull-up resistor option register 0	PU0	R/W	√	√	–	00H
FF31H	Pull-up resistor option register 1	PU1	R/W	√	√	–	00H
FF32H	Pull-up resistor option register 2	PU2	R/W	√	√	–	00H
FF33H	Pull-up resistor option register 3	PU3	R/W	√	√	–	00H
FF35H	FLMD0 Pull-up/Pull-down control register	FPCTL	R/W	√	√	–	00H
FF37H	FLMD0 Pull-up/Pull-down enable register	FPEN	R/W	√	√	–	00H
FF38H	Port output mode resistor 0	POM0	R/W	√	√	–	00H
FF39H	Port output mode resistor 1	POM1	R/W	√	√	–	00H
FF3AH	Port output mode resistor 2	POM2	R/W	√	√	–	00H
FF3BH	Port output mode resistor 3	POM3	R/W	√	√	–	00H
FF3CH	Pull-up resistor option register 12	PU12	R/W	√	√	–	08H
FF3EH	Port output mode resistor 12	POM12	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

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

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulatable Bit Unit			After Reset
					1 Bit	8 Bits	16 Bits	
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
FF6EH	Key return mode register 0	KRML		R/W	√	√	–	00H
FF6FH	Key return mode register 1	KRMH		R/W	√	√	–	00H
FF8CH	Timer clock selection register 51	TCL51		R/W	√	√	–	00H
FF99H	Watchdog timer enable register	WDTE		R/W	–	√	–	Note 1 1AH/9AH
FF9FH	Clock operation mode select register	OSCCTL		R/W	√	√	–	00H
FFA0H	Internal oscillation mode register	RCM		R/W	√	√	–	80H <sup>Note 2</sup>
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
FFACH	Reset control flag register	RESF		R	–	√	–	00H <sup>Note 3</sup>
FFB0H	RAM Data Retention control register	LVDET		R/W	√	√	–	Undefined
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 <sup>Note 3</sup>
FFBFH	Low-voltage detection level selection register	LVIS		R/W	√	√	–	00H <sup>Note 3</sup>
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
FFE4H	Interrupt mask flag register 0L		MK0L	R/W	√	√		√
FFE5H	Interrupt mask flag register 0H	MK0H	R/W	√	√	FFH		
FFE6H	Interrupt mask flag register 1L	MK1	MK1L	R/W	√	√	–	FFH

**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 values of RESF, LVIM, and LVIS vary depending on the reset source.

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

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulatable Bit Unit			After Reset
					1 Bit	8 Bits	16 Bits	
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
FFF0H	Internal memory size switching register <sup>Note</sup>	IMS		R/W	–	√	–	CFH
FFF9H	Internal high-speed oscillator trimming register	R4MTRM2		R/W	–	√	–	00H
FFFBH	Processor clock control register	PCC		R/W	√	√	–	01H

**Note** Regardless of the internal memory capacity, the initial value of the internal memory size switching register (IMS) of all products in the  $\mu$ PD179F11x, 179F12x microcontrollers is fixed (IMS = CFH). Therefore, set the value corresponding to each product as indicated below.

Flash Memory Version ( $\mu$ PD179F11x, 179F12x microcontrollers)	IMS	ROM Capacity	Internal High-Speed RAM Capacity
$\mu$ PD179F110	41H	4 KB	512 bytes
$\mu$ PD179F111	42H	8 KB	
$\mu$ PD179F112, 179F122	04H	16 KB	768 bytes
$\mu$ PD179F113, 179F123	C6H	24 KB	1 KB
$\mu$ PD179F114, 179F124	C8H	32 KB	

### 3.3 Instruction Address Addressing

An instruction address is determined by contents of the program counter (PC), 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)**).

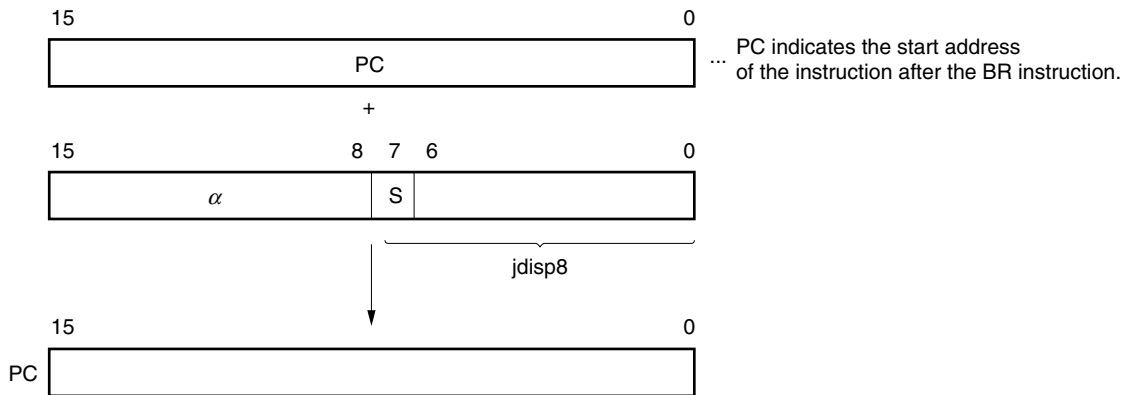
#### 3.3.1 Relative addressing

**[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.

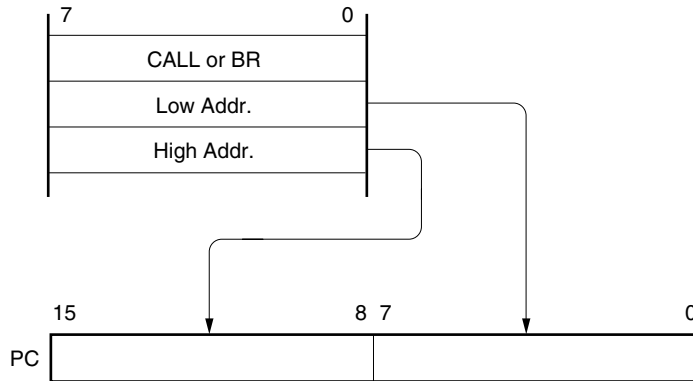
### 3.3.2 Immediate addressing

**[Function]**

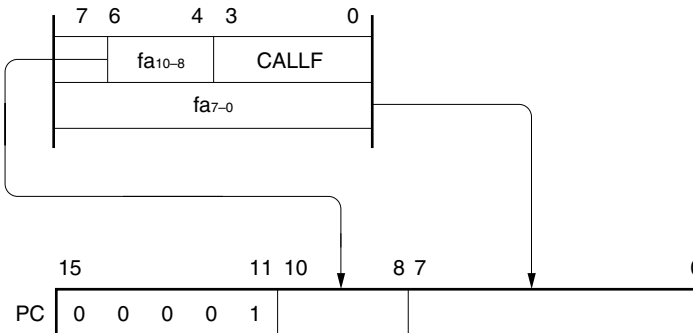
Immediate data in the instruction word is transferred to the program counter (PC) and branched.  
 This function is carried out when the CALL !addr16 or BR !addr16 or CALLF !addr11 instruction is executed.  
 CALL !addr16 and BR !addr16 instructions can be branched to the entire memory space.  
 The 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





<R>

### 3.3.3 Table indirect addressing

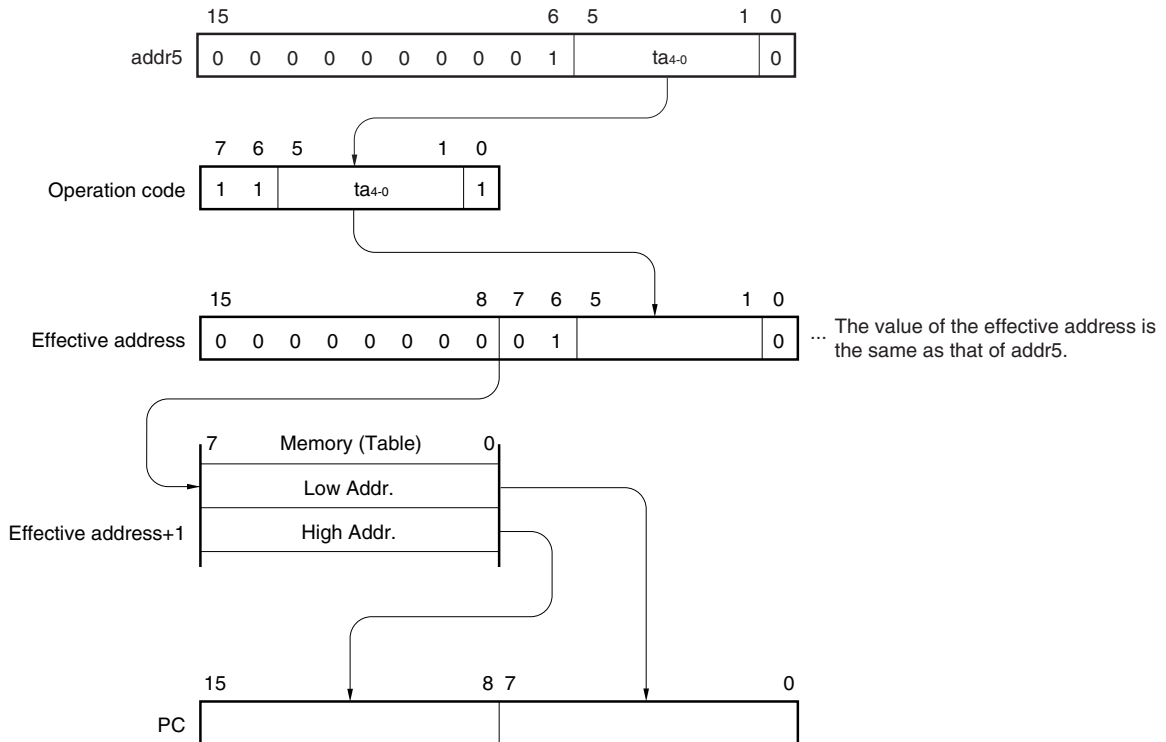
**[Function]**

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]**



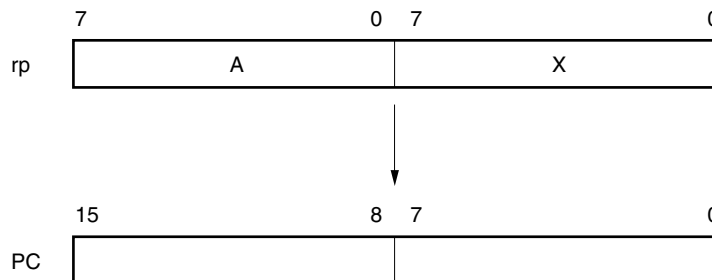
### 3.3.4 Register addressing

**[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  $\mu$ PD179F11x, 179F12x microcontrollers 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  $\times$  8-bit multiply instruction, the product of 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.

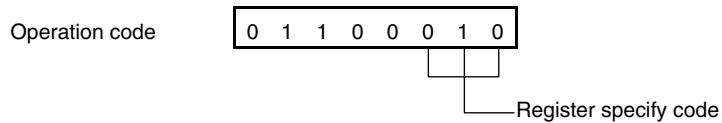
**[Operand format]**

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

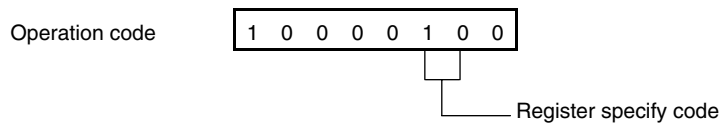
'r' and 'rp' can be described by absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

**[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.

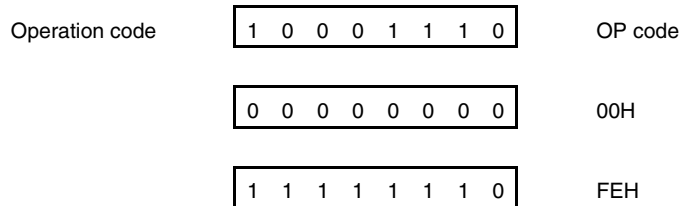
This addressing can be carried out for all of the memory spaces.

**[Operand format]**

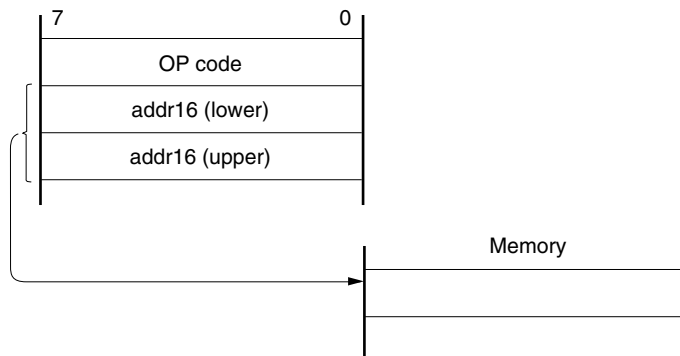
Identifier	Description
addr16	Label or 16-bit immediate data

**[Description example]**

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



**[Illustration]**





### 3.4.5 Special function register (SFR) addressing

**[Function]**

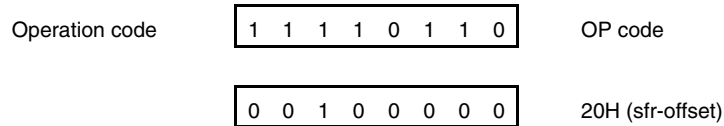
A memory-mapped special function register (SFR) is addressed with 8-bit immediate data in an instruction word. This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, the SFRs mapped at FF00H to FF1FH can be accessed with short direct addressing.

**[Operand format]**

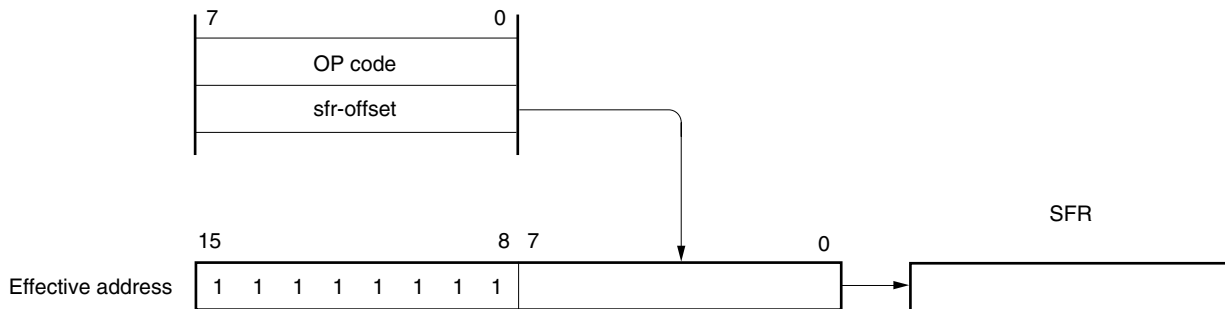
Identifier	Description
sfr	Special function register name
sfrp	16-bit manipulatable special function register name (even address only)

**[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.

**[Operand format]**

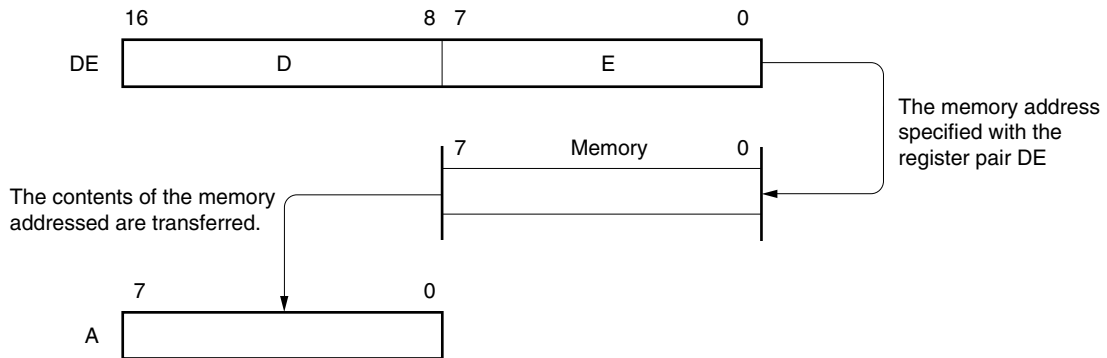
Identifier	Description
–	[DE], [HL]

**[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.

[Operand format]

Identifier	Description
-	[HL + byte]

[Description example]

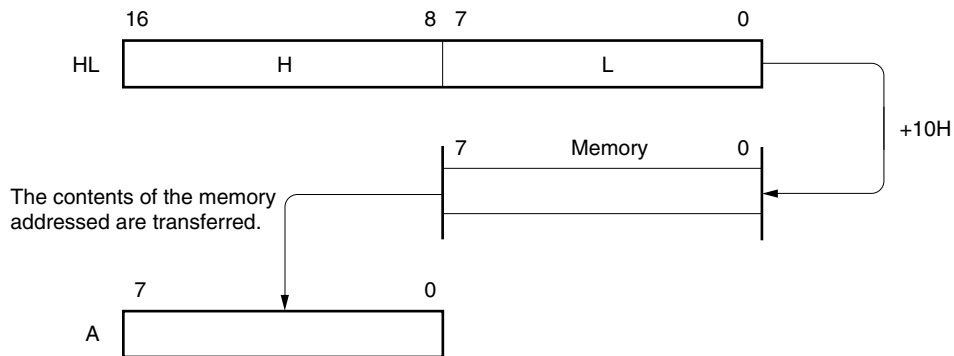
MOV A, [HL + 10H]; when setting byte to 10H

Operation code

1 0 1 0 1 1 1 0

0 0 0 1 0 0 0 0

[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.

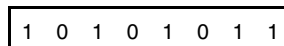
**[Operand format]**

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

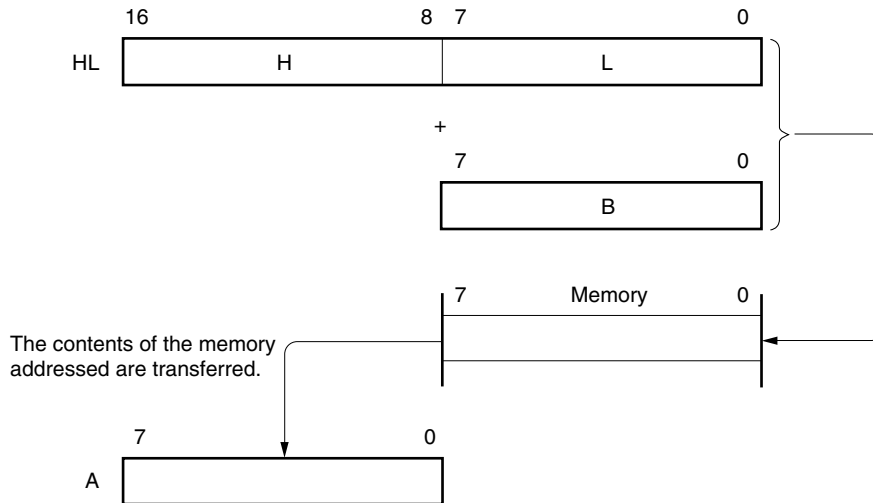
**[Description example]**

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

Operation code



**[Illustration]**



### 3.4.9 Stack addressing

**[Function]**

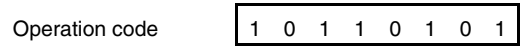
The stack area is indirectly addressed with the stack pointer (SP) contents.

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.

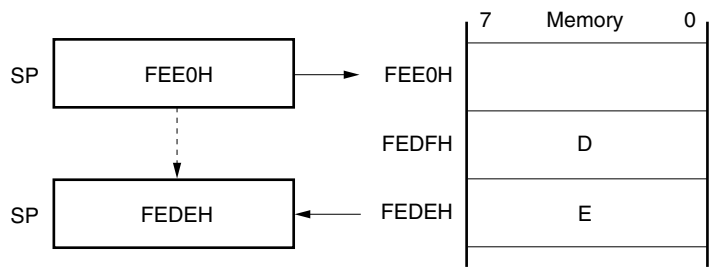
With stack addressing, only the internal high-speed RAM area can be accessed.

**[Description example]**

PUSH DE; when saving DE register



**[Illustration]**



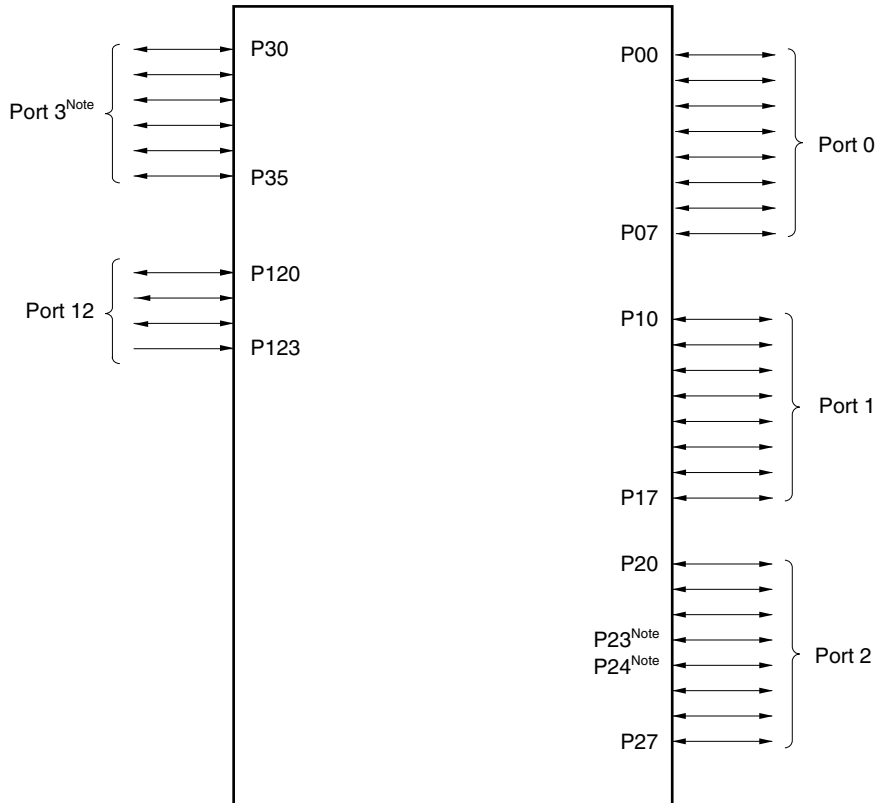
## CHAPTER 4 PORT FUNCTIONS

### 4.1 Port Functions

$\mu$ PD179F11x, 179F12x microcontrollers are provided with the ports shown in Figure 4-1, which enable variety of control operations. The functions of each port are shown in Table 4-1.

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**.

Figure 4-1. Port Types



**Note** 38-pin products only

Table 4-1. Port Functions

Function Name	I/O	Function	After Reset	Alternate Function
P00	I/O	Port 0. 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 in 1-bit units. P00 to P06 can be set to N-ch open-drain output or CMOS I/O in 1-bit units. P07 can be set to P-ch open-drain output or CMOS I/O in 1-bit units. This pin can be used as a carrier generator output for remote control by specifying P-ch open-drain output.	Input port	T151/TO51
P01				TOH0
P02				T1000/TxD6
P03				T1010/TO00/ RxD6
P04				INTP3/OCD1A
P05				INTP2/OCD1B
P06				T150/TO50/INTP1
P07				REM/TOH1
P10 to P17	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 in 1-bit units. P10 to P17 can be set to N-ch open-drain output or CMOS I/O in 1-bit units.	Input port	KR0 to KR7
P20 to P22	I/O	Port 2. 38-pin products: 8-bit I/O port 30-pin products: 6-bit I/O port Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified in 1-bit units. P20 to P22, P23 <sup>Note</sup> , P24 <sup>Note</sup> , P25 to P27 can be set to N-ch open-drain output or CMOS I/O in 1-bit units.	Input port	–
P23 <sup>Note</sup> , P24 <sup>Note</sup>				–
P25, P26				–
P27				INTP4
P30 to P35 <sup>Note</sup>	I/O	Port 3 <sup>Note</sup> . 6-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified in 1-bit units. P30 to P35 can be set to N-ch open-drain output or CMOS I/O in 1-bit units.	Input port	KR9 to KR14 <sup>Note</sup>
P120	I/O	Port 12. 3-bit I/O port and 1-bit Input-only port. Input/output can be specified in 1-bit units in P120 to P122. Use of an on-chip pull-up resistor can be specified in only P120 and P123.	Input port	EXLVI/INTP0
P121				X1/OCD0A
P122				X2/EXCLK/OCD0B
P123	Input	P120 to P122 can be set to N-ch open-drain output or CMOS I/O in 1-bit units.		RESET/KR8/ INTP5

**Note** 38-pin products only.

For the 30-pin products, be sure to set bits 3 and 4 of PM2, bits 0 to 5 of PM3 to “1”, and bits 3 and 4 of P2, PU2, and POM2, bits 0 to 5 of P3, PU3, and POM3 to “0”.

## 4.2 Port Configuration

Ports include the following hardware.

**Table 4-2. Port Configuration**

Item	Configuration
Control registers	Port mode register (PM0 to PM2, PM3 <sup>Note</sup> , PM12) Port register (P0 to P2, P3 <sup>Note</sup> , P12) Pull-up resistor option register (PU0 to PU2, PU3 <sup>Note</sup> , PU12) Port output mode register (POM0 to POM2, POM3 <sup>Note</sup> , POM12) Reset pin mode register (RSTMASK)
Port	<ul style="list-style-type: none"> <li data-bbox="418 575 1464 667">• 30-pin products      Total: 26 (N-ch open drain output or CMOS I/O: 24, P-ch open drain output or CMOS I/O: 1, CMOS Input: 1)</li> <li data-bbox="418 674 1464 764">• 38-pin products      Total: 34 (N-ch open drain output or CMOS I/O: 32, P-ch open drain output or CMOS I/O: 1, CMOS Input: 1)</li> </ul>
Pull-up resistor	<ul style="list-style-type: none"> <li data-bbox="418 777 1464 804">• 30-pin products      Total: 24</li> <li data-bbox="418 810 1464 837">• 38-pin products      Total: 32</li> </ul>

**Note**    38-pin products only

4.2.1 Port 0

Port 0 is an 8-bit 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 to P07 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).

P00 to P06 can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 0 (POM0).

P07 can be set to P-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 0 (POM0). This pin can be used as a carrier generator output for remote control by specifying P-ch open-drain output.

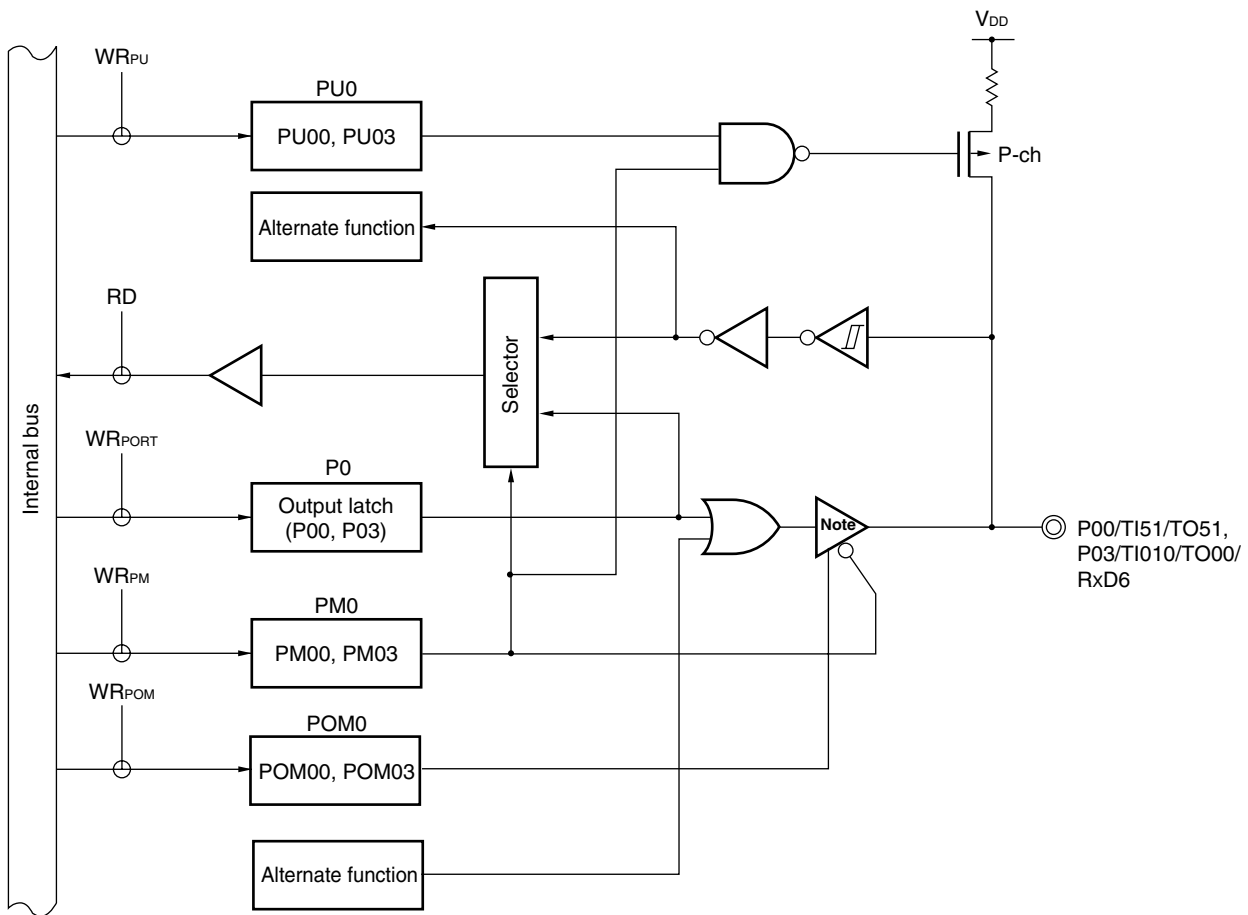
This port can also be used for timer I/O, serial data I/O, external interrupt request input, remote control output, and the setting connection for on-chip debug mode.

Reset signal generation sets port 0 to input mode.

Figures 4-2 to 4-6 show block diagrams of port 0.

<R>

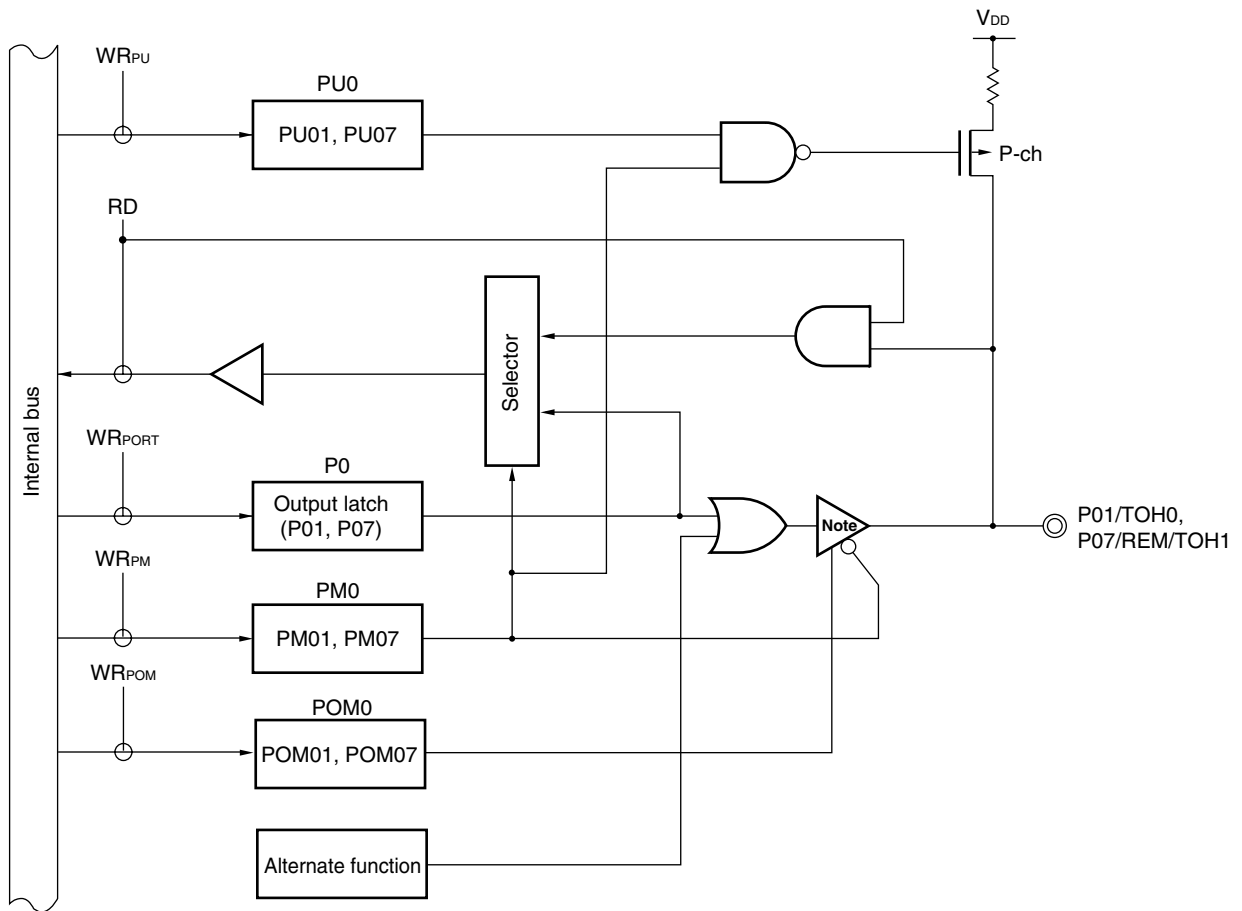
Figure 4-2. Block Diagram of P00 and P03



- P0: Port register 0
- PU0: Pull-up resistor option register 0
- PM0: Port mode register 0
- POM0: Port output mode register 0
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output

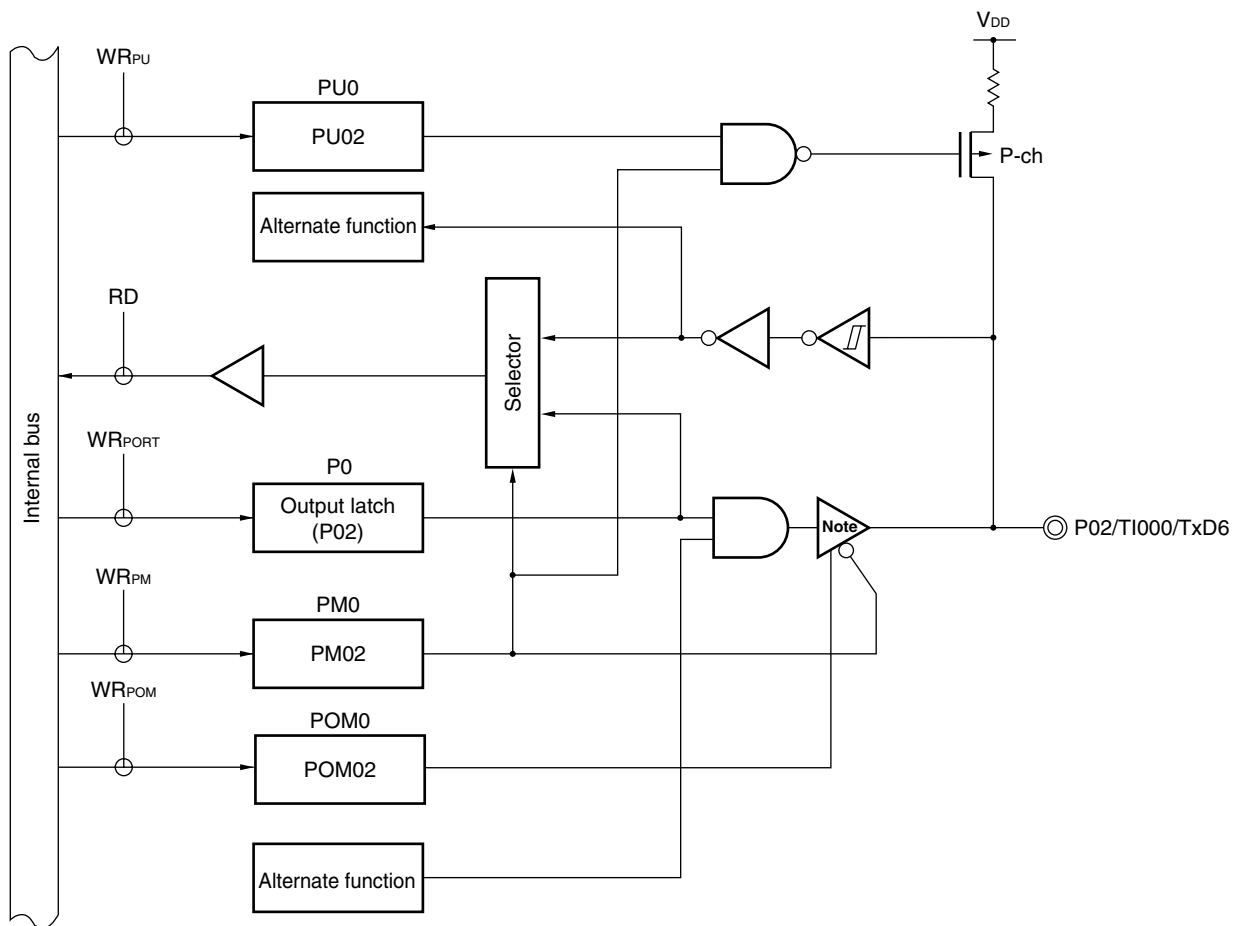
Figure 4-3. Block Diagram of P01 and P07



- P0: Port register 0
- PU0: Pull-up resistor option register 0
- PM0: Port mode register 0
- POM0: Port output mode register 0
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Note** For the P01, output buffer which can be switched to CMOS output or N-ch open-drain output. For the P07, output buffer which can be switched to CMOS output or P-ch open-drain output.

Figure 4-4. Block Diagram of P02

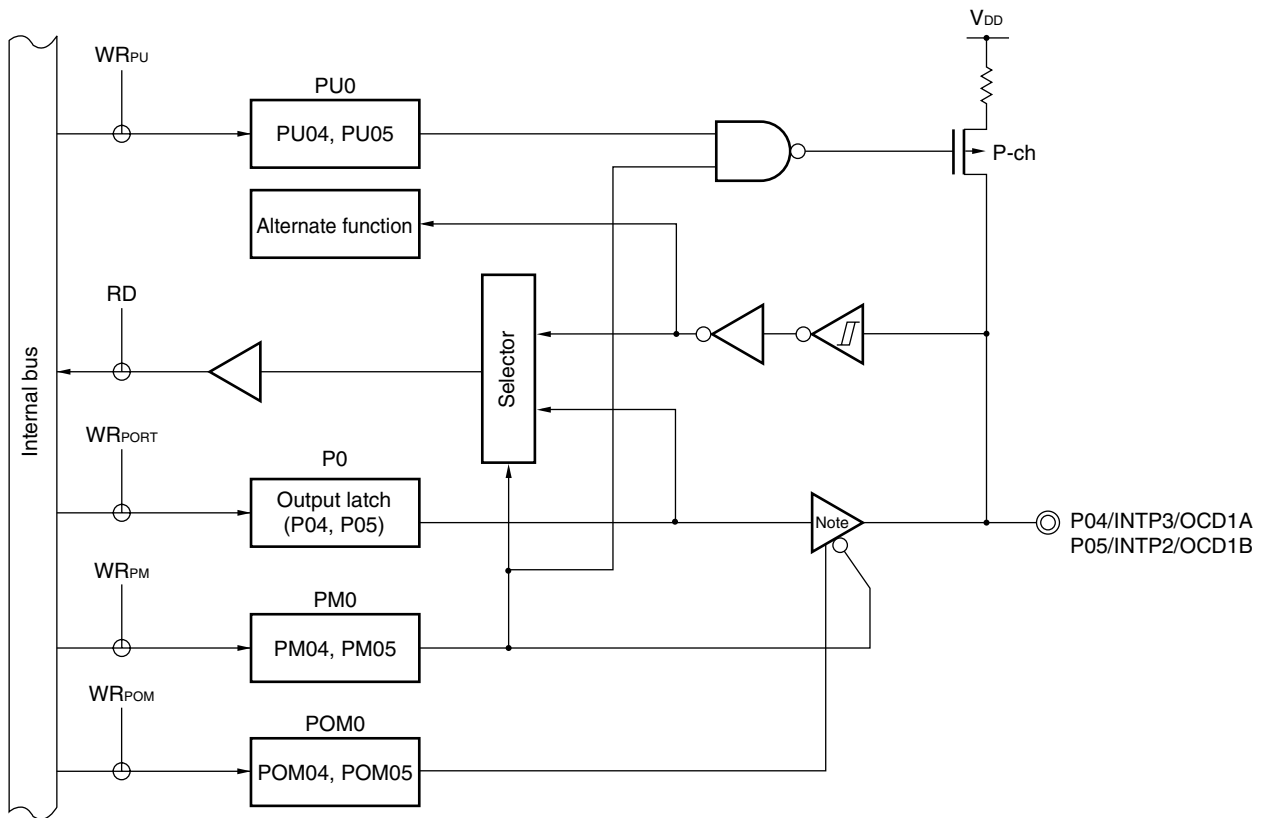


- P0: Port register 0
- PU0: Pull-up resistor option register 0
- PM0: Port mode register 0
- POM0: Port output mode register 0
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output



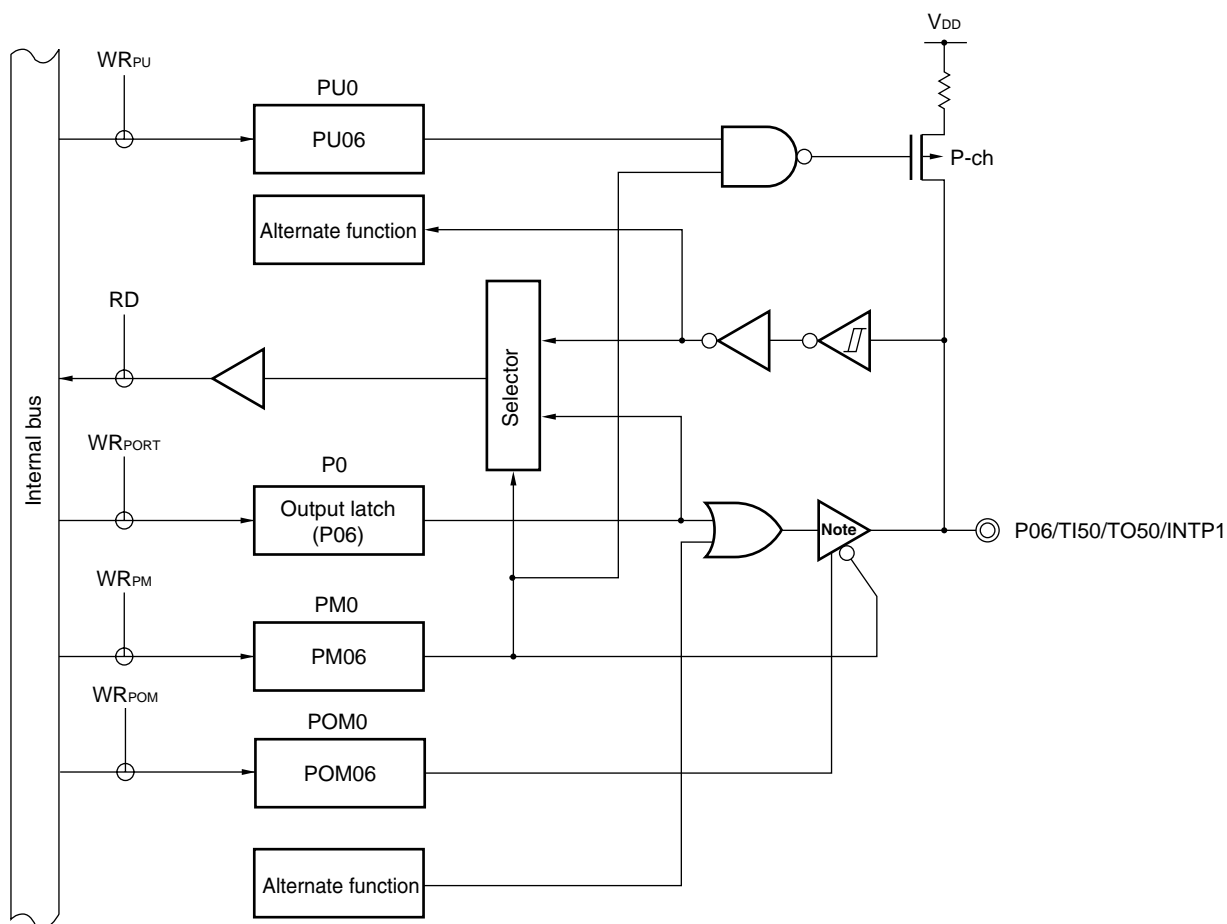
Figure 4-5. Block Diagram of P04 and P05



- P0: Port register 0
- PU0: Pull-up resistor option register 0
- PM0: Port mode register 0
- POM0: Port output mode register 0
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output

Figure 4-6. Block Diagram of P06



- P0: Port register 0
- PU0: Pull-up resistor option register 0
- PM0: Port mode register 0
- POM0: Port output mode register 0
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output

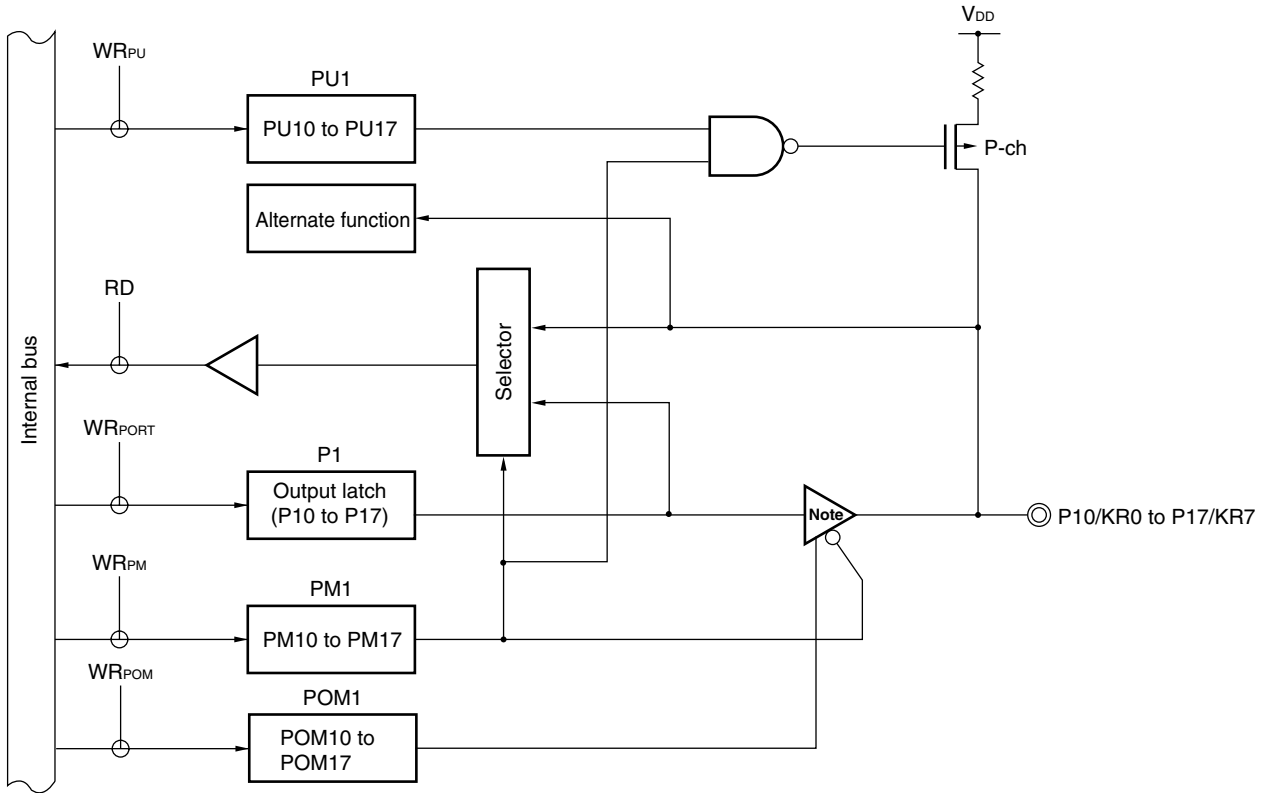
4.2.2 Port 1

Port 1 is an 8-bit 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). P10 to P17 can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 1 (POM1).

This port can also be used for key interrupt input.  
 Reset signal generation sets port 1 to input mode.

Figure 4-7 show block diagrams of port 1.

Figure 4-7. Block Diagram of P10 to P17



- P1: Port register 1
- PU1: Pull-up resistor option register 1
- PM1: Port mode register 1
- POM1: Port output mode register 1
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output

4.2.3 Port 2

Port 2 is a 6-bit I/O port with an output latch for 30-pin products. Port 2 is an 8-bit I/O port with an output latch for 38-pin products. Port 2 can be set to the input mode or output mode in 1-bit units using port mode register 0 (PM2). When the P20 to P27 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 2 (PU2). These pins can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 2 (POM2).

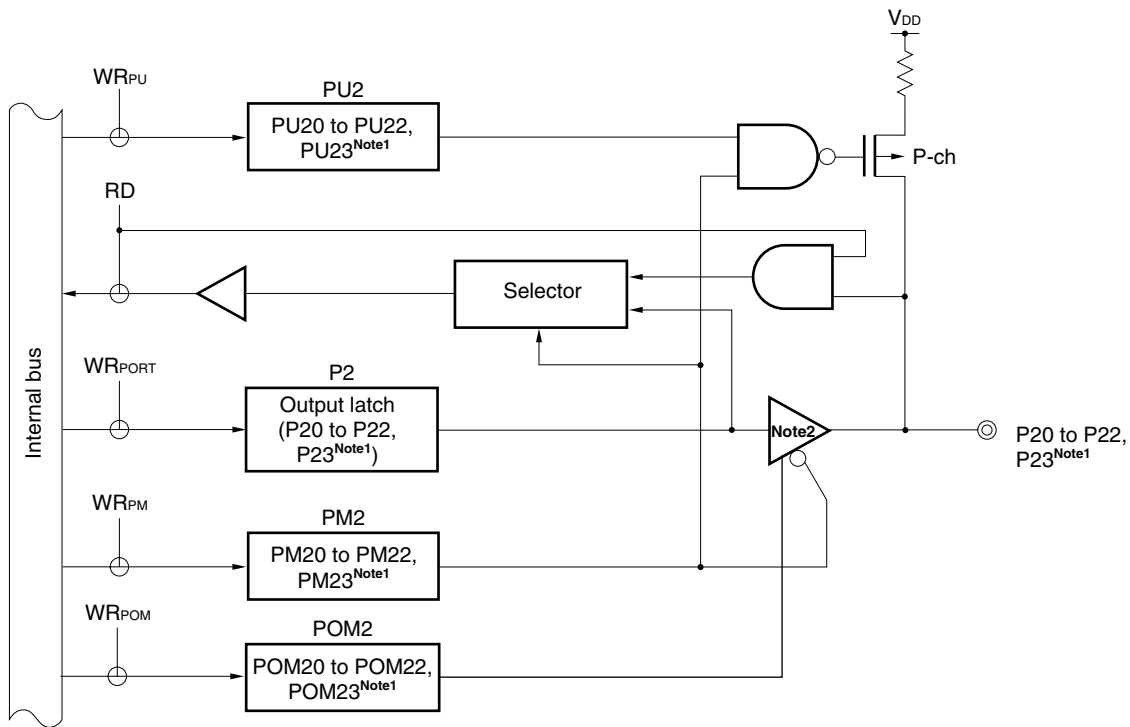
Only P27 can also be used for external interrupt request input.

Reset signal generation sets port 2 to input mode.

Figures 4-8 to 4-10 show block diagrams of port 2.

**Remark** 30-pin products: P20 to P22, P25, P26, P27/INTP4  
 38-pin products: P20 to P26, P27/INTP4

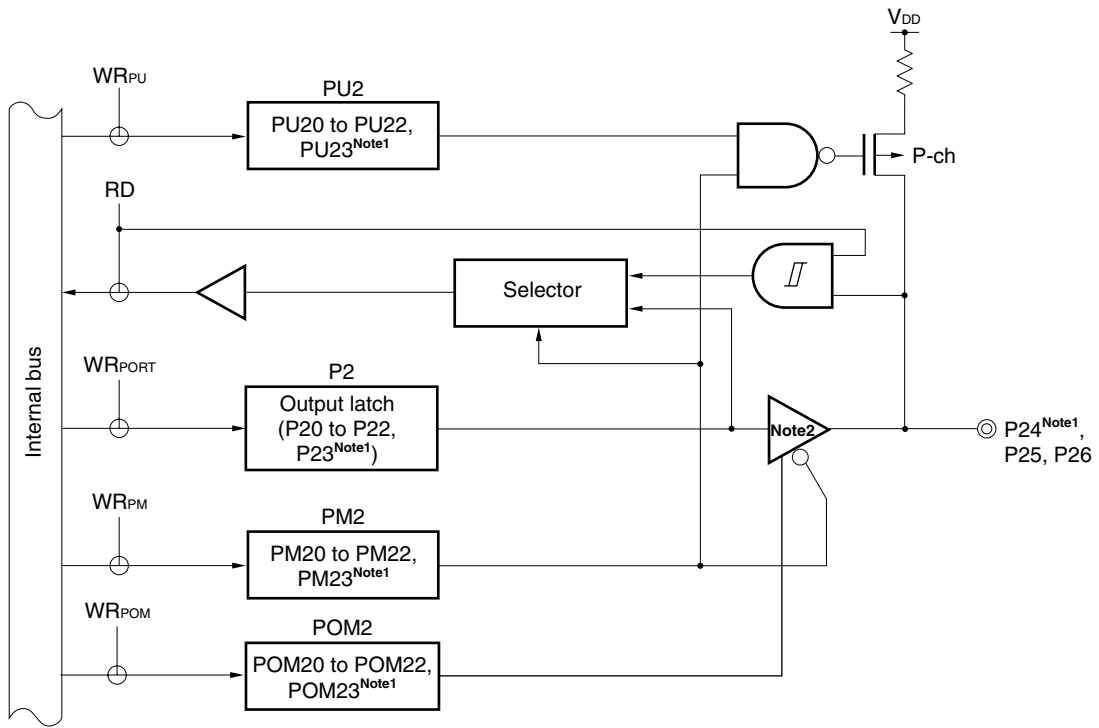
Figure 4-8. Block Diagram of P20 to P23



- P2: Port register 2
- PM2: Port mode register 2
- POM2: Port output mode register 2
- RD: Read signal
- WR<sub>xx</sub>: Write signal

- Notes 1.** 38-pin products only
2. Output buffer which can be switched to CMOS output or N-ch open-drain output

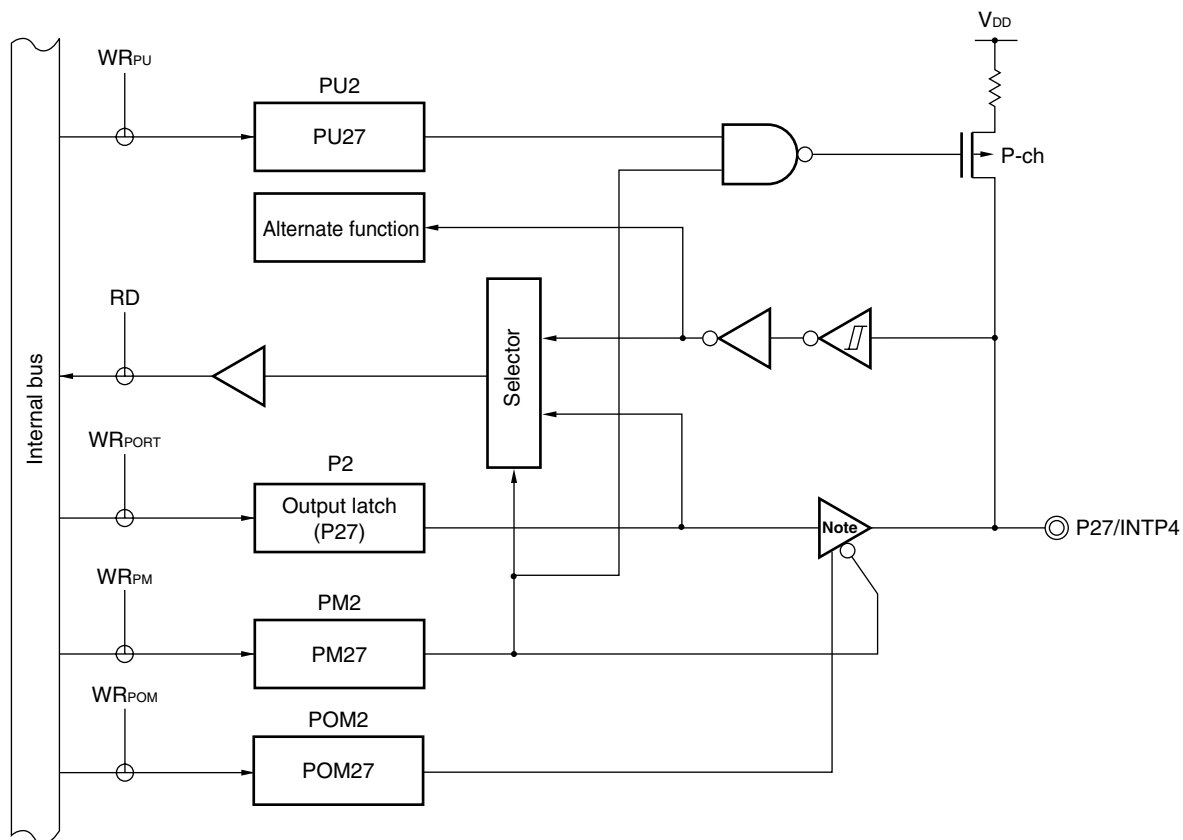
Figure 4-9. Block Diagram of P24 to P26



- P2: Port register 2
- PM2: Port mode register 2
- POM2: Port output mode register 2
- RD: Read signal
- WR<sub>xx</sub>: Write signal

- Notes 1.** 38-pin products only
2. Output buffer which can be switched to CMOS output or N-ch open-drain output

Figure 4-10. Block Diagram of P27



- P2: Port register 2
- PM2: Port mode register 2
- POM2: Port output mode register 2
- RD: Read signal
- $WR_{xx}$ : Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output

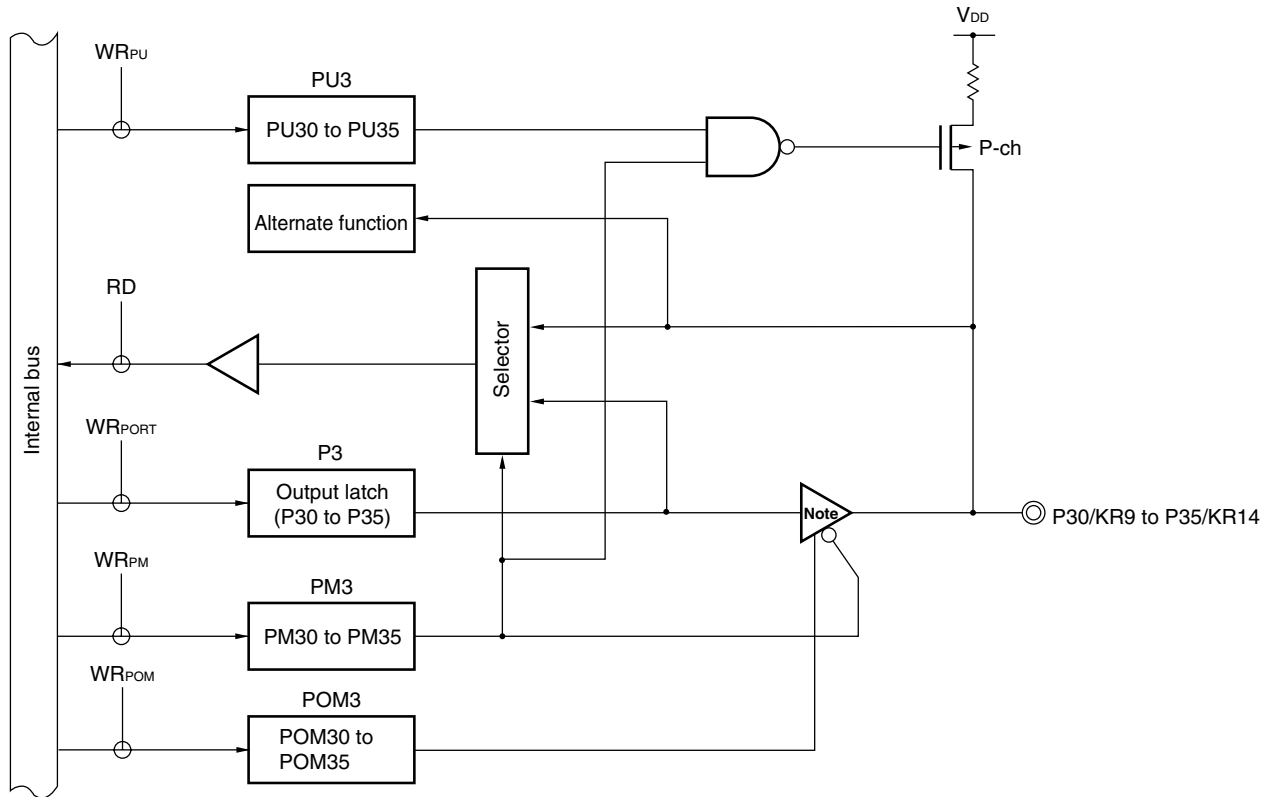
**4.2.4 Port 3 (38-pin products only)**

Port 3 is a 6-bit 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 0 (PM3). When the P30 to P35 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 3 (PU3). P30 to P35 can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 3 (POM3).

This port can also be used for key interrupt input.  
 Reset signal generation sets port 3 to input mode.

Figure 4-11 show block diagrams of port 3.

**Figure 4-11. Block Diagram of P30 to P35**



- P3: Port register 3
- PU3: Pull-up resistor option register 3
- PM3: Port mode register 3
- POM3: Port output mode register 3
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output

4.2.5 Port 12

P120 to P122 are a 3-bit I/O port, and P123 is an input-only port. P120 to P122 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 and P123, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 12 (PU12). P120 to P122 can be set to N-ch open-drain output or CMOS I/O in 1-bit units using port output mode register 12 (POM12).

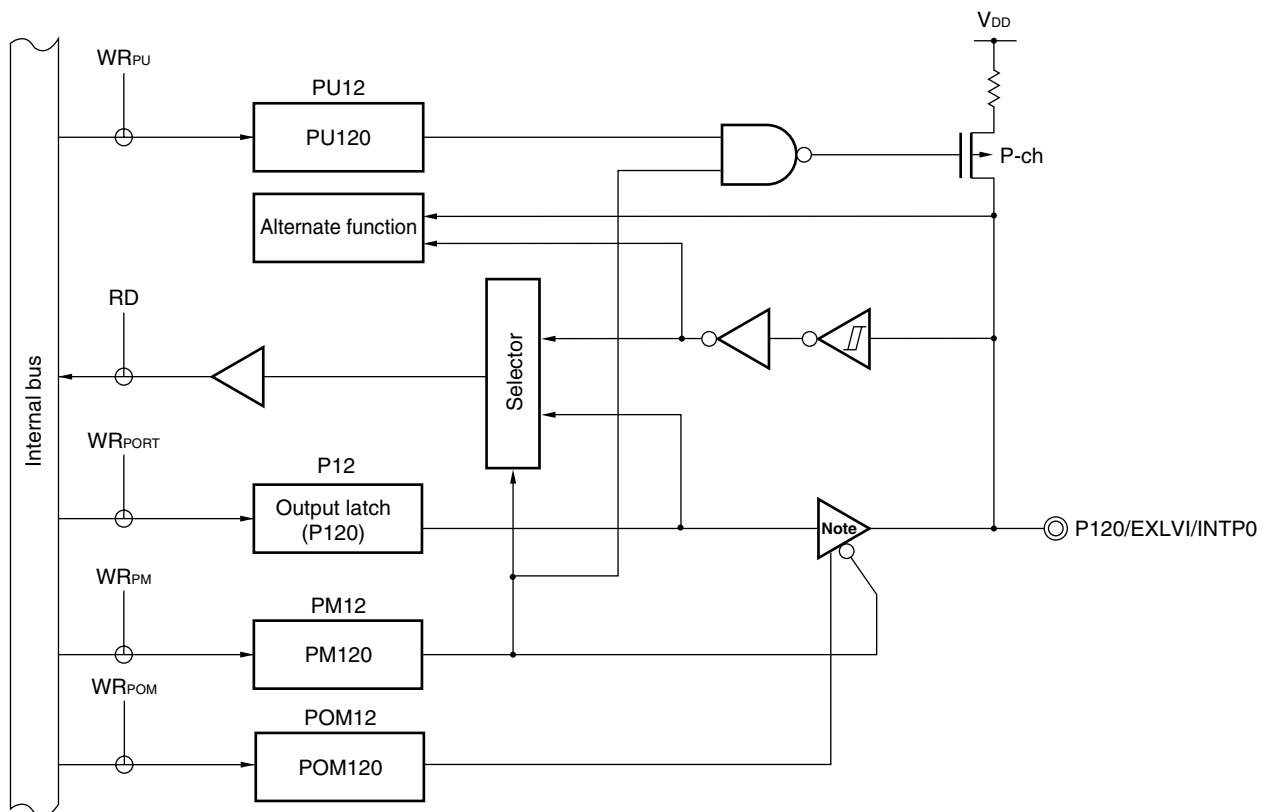
This port can also be used as pins for external interrupt request input, potential input for external low-voltage detection, connecting resonator for main system clock, external clock input for main system clock, system reset input, key interrupt input, and the setting connection for on-chip debug mode.

Reset signal generation sets port 12 to input mode.

Figures 4-12 and 4-14 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 4-12. Block Diagram of P120

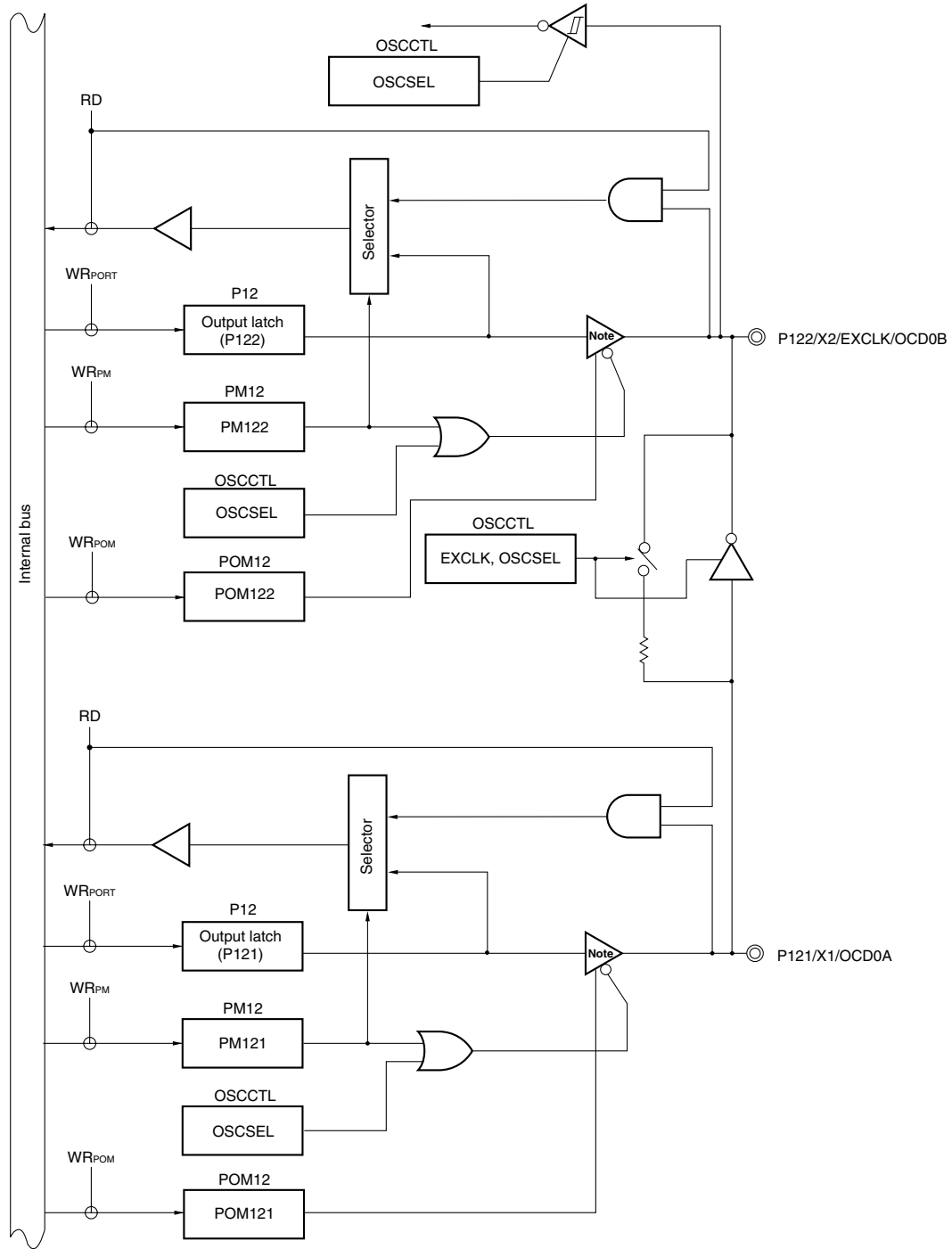


- P12: Port register 12
- PU12: Pull-up resistor option register 12
- PM12: Port mode register 12
- POM12: Port output mode register 12
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output



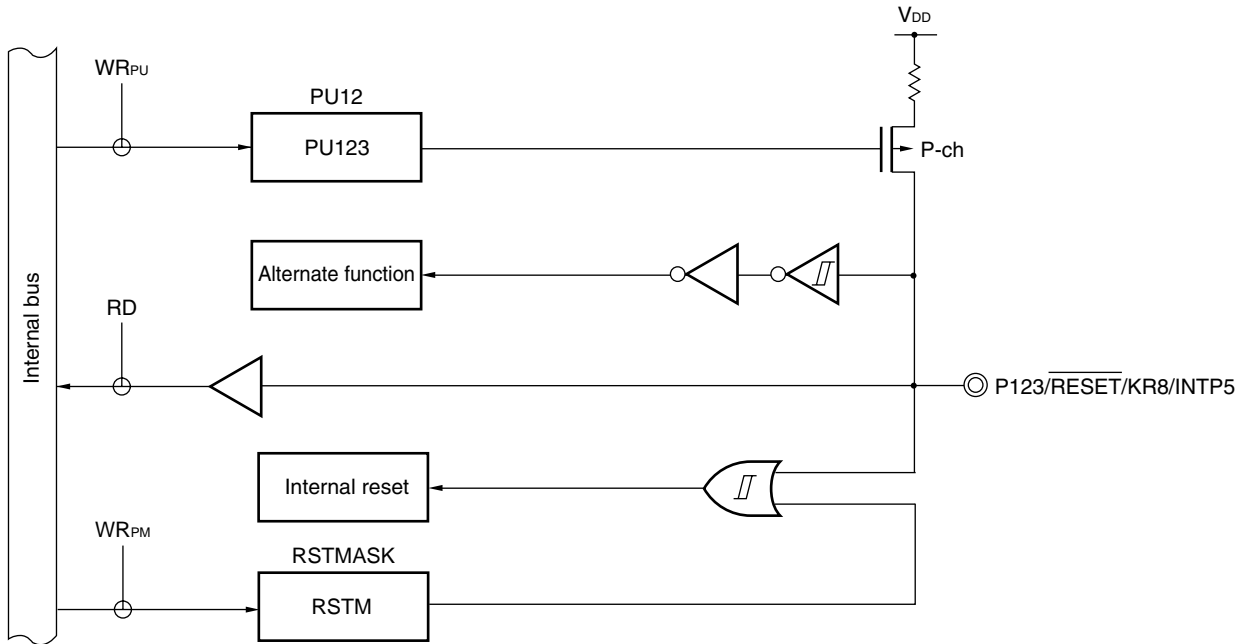
Figure 4-13. Block Diagram of P121 and P122



- 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<sub>xx</sub>: Write signal

**Note** Output buffer which can be switched to CMOS output or N-ch open-drain output

Figure 4-14. Block Diagram of P123



PU12: Pull-up resistor option register 12

RSTMASK: Reset pin mode register

RD: Read signal

WR<sub>xx</sub>: Write signal

**Remark** After reset, reset function and pull-up resistor are enabled (RSTM = 0, PU123 = 1).  
When using the P123/RESET/KR8/INTP5 pin for P123, KR8, or INTP5 set RSTM to “1”.

### 4.3 Registers Controlling Port Function

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

- Port mode registers (PM0 to PM2, PM3<sup>Note</sup>, PM12)
- Port registers (P0 to P2, P3<sup>Note</sup>, P12)
- Pull-up resistor option registers (PU0 to PU2, PU3<sup>Note</sup>, PU12)
- Port output mode registers (POM0 to POM2, POM3<sup>Note</sup>, POM12)
- Reset pin mode register (RSTMASK)

**Note** 38-pin products only

#### (1) Port mode registers (PM0 to PM2, PM3<sup>Note</sup>, PM12)

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, Output Latch, Pull-up Resistor Option Register, and Port Output Mode Register When Using Alternate Function**.

**Note** 38-pin products only

**Figure 4-15. Format of Port Mode Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FF21H	FFH	R/W
PM2	PM27	PM26	PM25	PM24 <sup>Note</sup>	PM23 <sup>Note</sup>	PM22	PM21	PM20	FF22H	FFH	R/W
PM3 <sup>Note</sup>	1	1	PM35 <sup>Note</sup>	PM34 <sup>Note</sup>	PM33 <sup>Note</sup>	PM32 <sup>Note</sup>	PM31 <sup>Note</sup>	PM30 <sup>Note</sup>	FF23H	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 3, 12; n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

**Note** 38-pin products only

**Caution** For the 30-pin products, be sure to set bits 3 and 4 of PM2, bits 0 to 5 of PM3 to “1”.

**(2) Port registers (P0 to P2, P3<sup>Note</sup>, and P12)**

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 value of the output latch is read.

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

Reset signal generation sets these registers to 00H.

**Note** 38-pin products only

**Figure 4-16. Format of Port 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	P27	P26	P25	P24 <sup>Note</sup>	P23 <sup>Note</sup>	P22	P21	P20	FF02H	00H (output latch)	R/W
P3 <sup>Note</sup>	0	0	P35 <sup>Note</sup>	P34 <sup>Note</sup>	P33 <sup>Note</sup>	P32 <sup>Note</sup>	P31 <sup>Note</sup>	P30 <sup>Note</sup>	FF03H	00H (output latch)	R/W
P12	0	0	0	0	P123	P122	P121	P120	FF0CH	00H (output latch)	R/W

Pmn	m = 0 to 3, 12; n = 0 to 7	
	Output data control (in output mode)	Input data read (in input mode)
0	Output 0	Input low level
1	Output 1	Input high level

**Note** 38-pin products only

**Caution** For the 30-pin products, be sure to set bits 3 and 4 of P2, bits 0 to 5 of P3 to “0”.

**(3) Pull-up resistor option registers (PU0 to PU2, PU3<sup>Note</sup>, and PU12)**

These registers specify whether the on-chip pull-up resistors of P00 to P07, P10 to P17, P20 to P22, P23<sup>Note</sup>, P24<sup>Note</sup>, P25 to P27, P30 to P35<sup>Note</sup>, P120, and P123 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 PU0, PU1, PU3<sup>Note</sup>, and PU12. 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 PU0, PU1, PU3<sup>Note</sup>, and PU12. These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

<R>

Reset signal generation sets these registers to 00H (08H for PU12).

**Note** 38-pin products only

**Figure 4-17. Format of Pull-up Resistor Option Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PU0	PU07	PU06	PU05	PU04	PU03	PU02	PU01	PU00	FF30H	00H	R/W
PU1	PU17	PU16	PU15	PU14	PU13	PU12	PU11	PU10	FF31H	00H	R/W
PU2	PU27	PU26	PU25	PU24 <sup>Note</sup>	PU23 <sup>Note</sup>	PU22	PU21	PU20	FF32H	00H	R/W
PU3 <sup>Note</sup>	0	0	PU35 <sup>Note</sup>	PU34 <sup>Note</sup>	PU33 <sup>Note</sup>	PU32 <sup>Note</sup>	PU31 <sup>Note</sup>	PU30 <sup>Note</sup>	FF33H	00H	R/W
PU12	0	0	0	0	PU123	0	0	PU120	FF3CH	08H	R/W

PUmn	Pmn pin on-chip pull-up resistor selection (m = 0 to 3, 12; n = 0 to 7)
0	On-chip pull-up resistor not connected
1	On-chip pull-up resistor connected

**Note** 38-pin products only

**Caution** For the 30-pin products, be sure to set bits 3 and 4 of PU2, bits 0 to 5 of PU3 to “0”.

**(4) Port output mode resistors (POM0 to POM2, POM3<sup>Note</sup>, and POM12)**

These registers set the output mode of P00 to P07, P10 to P17, P20 to P22, P23<sup>Note</sup>, P24<sup>Note</sup>, P25 to P27, P30 to P35<sup>Note</sup>, P120 to P122.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction. Reset signal generation sets these registers to 00H.

**Note** 38-pin products only

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

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POM0	POM07	POM06	POM05	POM04	POM03	POM02	POM01	POM00	FF38H	00H	R/W
POM1	POM17	POM16	POM15	POM14	POM13	POM12	POM11	POM10	FF39H	00H	R/W
POM2	POM27	POM26	POM25	POM24 <sup>Note</sup>	POM23 <sup>Note</sup>	POM22	POM21	POM20	FF3AH	00H	R/W
POM3 <sup>Note</sup>	0	0	POM35 <sup>Note</sup>	POM34 <sup>Note</sup>	POM33 <sup>Note</sup>	POM32 <sup>Note</sup>	POM31 <sup>Note</sup>	POM30 <sup>Note</sup>	FF3BH	00H	R/W
POM12	0	0	0	0	0	POM122	POM121	POM120	FF3EH	00H	R/W

POMmn	Pmn pin output mode selection (m = 0 to 3, 12; n = 0 to 7)
0	CMOS output
1	N-ch open-drain output (P07:P-ch open-drain output)

**Note** 38-pin products only

**Caution** For the 30-pin products, be sure to set bits 3 and 4 of POM2, bits 0 to 5 of POM3 to “0”.

**(5) Reset pin mode register (RSTMASK)**

RSTMASK is the register that controls whether the alternate function other than the reset function of  $\overline{\text{RESET}}$ /P123/KR8/INTP5 pin is enabled or disabled.

This register can be set by a 1-bit or 8-bit memory manipulation instruction. Reset signal generation sets these registers to 00H.

**Figure 4-19. Format of Reset Pin Mode Register (RSTMASK)**

Address: FF2EH After reset: 00H R/W

Symbol	7	6	5	4	<3>	2	1	0
RSTMASK	0	0	0	0	RSTM	0	0	0

RSTM	Controlling the alternate function other than the reset function is enabled or disabled
0	The alternate function is disabled. ( $\overline{\text{RESET}}$ /P123/KR8/INTP5 pin functions as $\overline{\text{RESET}}$ pin.)
1	The alternate function is enabled. ( $\overline{\text{RESET}}$ /P123/KR8/INTP5 pin does not function as $\overline{\text{RESET}}$ pin.)

## 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.

<R> **4.5 Settings of Port Mode Register, Output Latch, Pull-up Resistor Option Register, and Port Output Mode Register When Using Alternate Function**

To use the alternate function of a port pin, set the port mode register, output latch, pull-up resistor option register, and port output mode register as shown in Table 4-3.

**Table 4-3. Settings of Port Mode Register, Output Latch , Pull-up Resistor Option Register, and Port Output Mode Register When Using Alternate Function**

Pin Name	Alternate Function		PM <sub>xx</sub>	P <sub>xx</sub>	PU <sub>xx</sub>	POM <sub>xx</sub>
	Function Name	I/O				
P00	TI51	Input	1	×	×	×
	TO51	Output	0	0	×	×
P01	TOH0	Output	0	0	×	×
P02	TI000	Input	1	×	×	×
	TXD6	Output	0	1	×	×
P03	TI010	Input	1	×	×	×
	TO00	Output	0	0	×	×
	RXD6	Input	1	×	×	×
P04	INTP3	Input	1	×	×	×
P05	INTP2	Input	1	×	×	×
P06	TI50	Input	1	×	×	×
	TO50	Output	0	0	×	×
	INTP1	Input	1	×	×	×
P07	REM	Output	0	0	×	1
	TOH1	Output	0	0	×	0
P10 to P17	KR0 to KR7	Input	1	×	1	×
P27	INTP4	Input	1	×	×	×
P30 to P35 <sup>Note</sup>	KR9 to KR14 <sup>Note</sup>	Input	1	×	1	×
P120	EXLVI	Input	1	×	×	×
	INTP0	Input	1	×	×	×
P121	X1	–	×	×	0	×
P122	X2	–	×	×	0	×
	EXCLK	Input	1	×	×	×
P123	$\overline{\text{RESET}}$	Input	–	×	×	×
	KR8	Input	–	×	1	×
	INTP5	Input	–	×	×	×

**Note** 38-pin products only

- Remark 1.** ×: Don't care  
 PM<sub>xx</sub>: Port mode register  
 P<sub>xx</sub>: Port output latch  
 PU<sub>xx</sub>: Pull-up resistor option register  
 POM<sub>xx</sub>: Port output mode register

- The P04/INTP3, P05/INTP2, P121/X1, and P122/X2/EXCLK pins can be used as on-chip debug mode setting pins (OCD1A, OCD1B, OCD0A, OCD0B) when the on-chip debug function is used. For how to



connect an on-chip debug emulator with programming function (QB-MINI2), see **CHAPTER 19 ON-CHIP DEBUG FUNCTION**.

#### 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  $\mu$ PD179F11x, 179F12x microcontrollers.

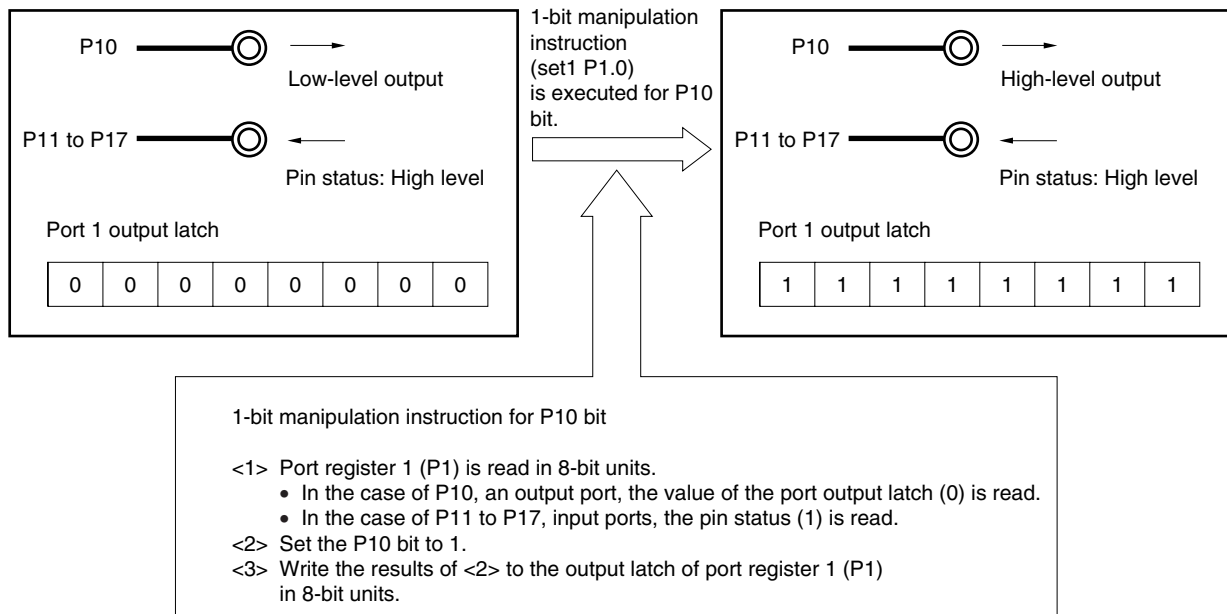
- <1> The Pn register is read in 8-bit units.
- <2> The targeted one bit is manipulated.
- <3> The Pn register is written in 8-bit units.

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 4 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  $f_{RH} = 4 \text{ MHz} \pm 2\%$ . 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 4 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

<R> This circuit oscillates a clock of  $f_{RL} = 240 \text{ kHz (TYP.)}$ . 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  $f_{RL}$  or  $f_{RL}/2^7$  is selected)

- Remarks**
1.  $f_x$ : X1 clock oscillation frequency
  2.  $f_{RH}$ : Internal high-speed oscillation clock frequency
  3.  $f_{EXCLK}$ : External main system clock frequency
  4.  $f_{RL}$ : Internal low-speed oscillation clock frequency

## 5.2 Configuration of Clock Generator

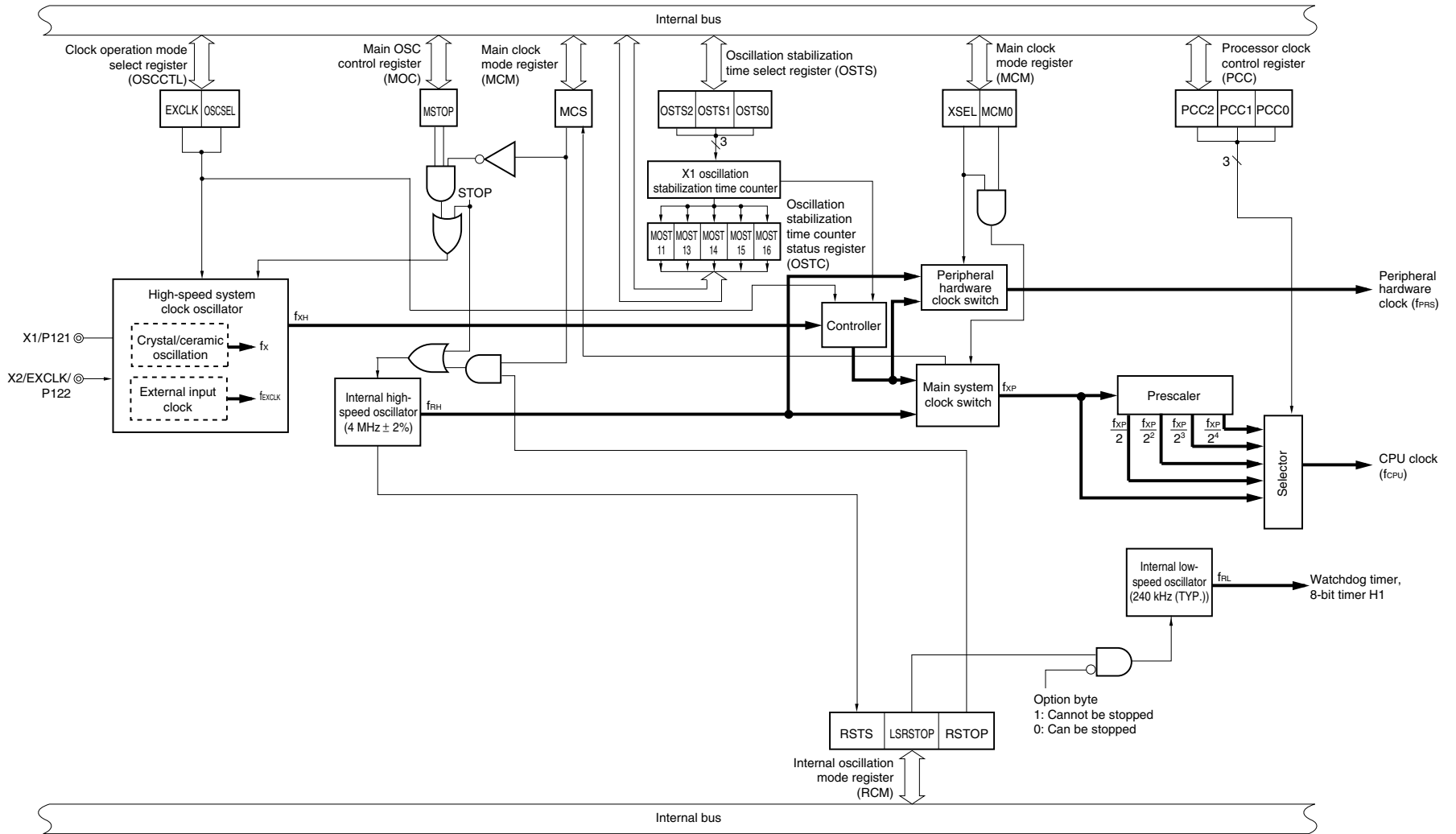
The clock generator includes the following hardware.

**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

<R>

Figure 5-1. Block Diagram of Clock Generator



- Remarks**
1.  $f_X$ : X1 clock oscillation frequency
  2.  $f_{RH}$ : Internal high-speed oscillation clock frequency
  3.  $f_{EXCLK}$ : External main system clock frequency
  4.  $f_{XH}$ : High-speed system clock frequency
  5.  $f_{XP}$ : Main system clock frequency
  6.  $f_{PRS}$ : Peripheral hardware clock frequency
  7.  $f_{CPU}$ : CPU clock frequency
  8.  $f_{RL}$ : 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, and the gain of the on-chip oscillator. 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

<R> **Caution** 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).

**Remark** f<sub>xH</sub>: High-speed system clock 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<sup>Note</sup>

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

PCC2	PCC1	PCC0	CPU clock (f <sub>CPU</sub> ) selection	
			FLMD0 = 0	FLMD0 = 1
0	0	0	f <sub>XP</sub>	f <sub>RH</sub>
0	0	1	f <sub>XP</sub> /2 (default)	f <sub>RH</sub> /2 (default)
0	1	0	f <sub>XP</sub> /2 <sup>2</sup>	f <sub>RH</sub> /2 <sup>2</sup>
0	1	1	f <sub>XP</sub> /2 <sup>3</sup>	f <sub>RH</sub> /2 <sup>3</sup>
1	0	0	f <sub>XP</sub> /2 <sup>4</sup>	f <sub>RH</sub> /2 <sup>4</sup>
Other than above			Setting prohibited	

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

**Caution** Be sure to clear bits 3 to 7 to “0”.

**Remark** f<sub>XP</sub>: Main system clock oscillation frequency

The fastest instruction can be executed in 2 clocks of the CPU clock in the  $\mu$ PD179F11x, 179F12x microcontrollers. Therefore, the relationship between the CPU clock (f<sub>CPU</sub>) 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 <sub>CPU</sub> )	Minimum Instruction Execution Time: 2/f <sub>CPU</sub>	
	Main System Clock	
	High-Speed System Clock <sup>Note1</sup>	Internal High-Speed Oscillation Clock <sup>Note1</sup>
	At 4 MHz Operation	At 4 MHz $\pm$ 2 % Operation
f <sub>XP</sub>	0.5 $\mu$ s <sup>Note2</sup>	0.5 $\mu$ s (TYP.) <sup>Note2</sup>
f <sub>XP</sub> /2	1 $\mu$ s	1 $\mu$ s (TYP.)
f <sub>XP</sub> /2 <sup>2</sup>	2 $\mu$ s	2 $\mu$ s (TYP.)
f <sub>XP</sub> /2 <sup>3</sup>	4 $\mu$ s	4 $\mu$ s (TYP.)
f <sub>XP</sub> /2 <sup>4</sup>	8 $\mu$ s	8 $\mu$ s (TYP.)

- Notes 1.** 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**).
- 2.** It is settable only when V<sub>DD</sub> = 2.0 to 3.6 V.

**(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<sup>Note 1</sup>.

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

Address: FFA0H After reset: 80H<sup>Note 1</sup> R/W<sup>Note 2</sup>

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)
  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<sup>Note</sup>

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 <sub>XP</sub> )	Peripheral hardware clock (f <sub>PRS</sub> )
0	0	Internal high-speed oscillation clock (f <sub>RH</sub> )	Internal high-speed oscillation clock (f <sub>RH</sub> )
0	1		High-speed system clock (f <sub>XH</sub> )
1	0		
1	1		

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<sub>PRS</sub> 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<sub>RL</sub>” or “f<sub>RL</sub>/2<sup>7</sup>” 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 (T1000 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)**

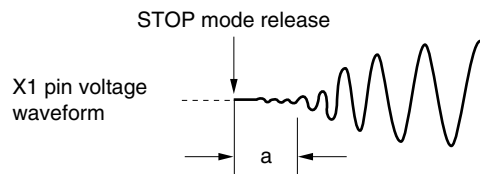
Address: FFA3H After reset: 00H R

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			
					$f_x = 1 \text{ MHz}$	$f_x = 2 \text{ MHz}$	$f_x = 4 \text{ MHz}$	
0	0	0	0	0	Less than $2^{11}/f_x$	Less than 2.04 ms	Less than 1.02 ms	Less than 510 $\mu\text{s}$
1	0	0	0	0	$2^{11}/f_x \text{ min.}$	2.04 ms min	1.02 ms min	510 $\mu\text{s}$ min
1	1	0	0	0	$2^{13}/f_x \text{ min.}$	8.20 ms min	4.10 ms min	2.04 ms min
1	1	1	0	0	$2^{14}/f_x \text{ min.}$	16.38 ms min.	8.19 ms min.	4.10 ms min
1	1	1	1	0	$2^{15}/f_x \text{ min.}$	32.77 ms min.	16.38 ms min.	8.19 ms min.
1	1	1	1	1	$2^{16}/f_x \text{ min.}$	65.45 ms min.	32.77 ms min.	16.38 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 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  $\leq$  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.
  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

**(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)**

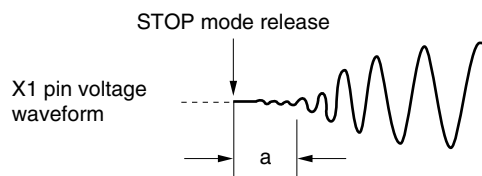
Address: FFA4H After reset: 05H R/W

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			
			$f_x = 1 \text{ MHz}$	$f_x = 2 \text{ MHz}$	$f_x = 4 \text{ MHz}$	
0	0	1	$2^{11}/f_x$	2.04 ms	1.02 ms	510 $\mu\text{s}$
0	1	0	$2^{13}/f_x$	8.19 ms	4.10 ms	2.04 m
0	1	1	$2^{14}/f_x$	16.38 ms	8.19 ms	4.10 ms
1	0	0	$2^{15}/f_x$	32.77 ms	16.38 ms	8.19 ms
1	0	1	$2^{16}/f_x$	65.45 ms	32.77 ms	16.38 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

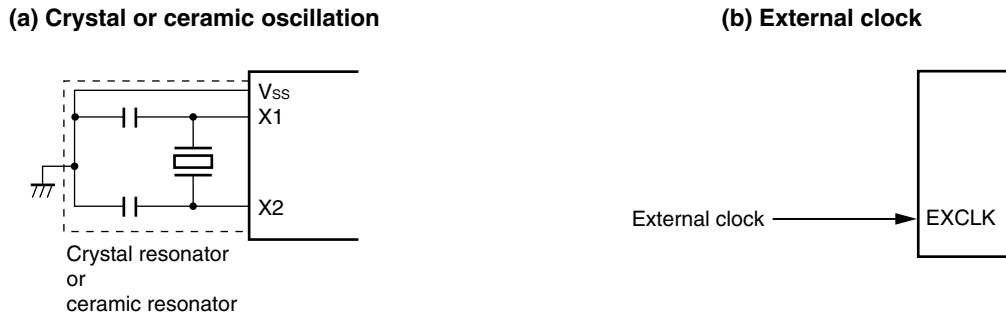
### 5.4.1 X1 oscillator

The X1 oscillator oscillates with a crystal resonator or ceramic resonator (1 to 4 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



<R>

**Caution** 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<sub>SS</sub>.
- Do not ground the capacitor to a ground pattern through which a high current flows.
- Do not fetch signals from the oscillator.

Cautions are listed on the Figure 5-10.

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

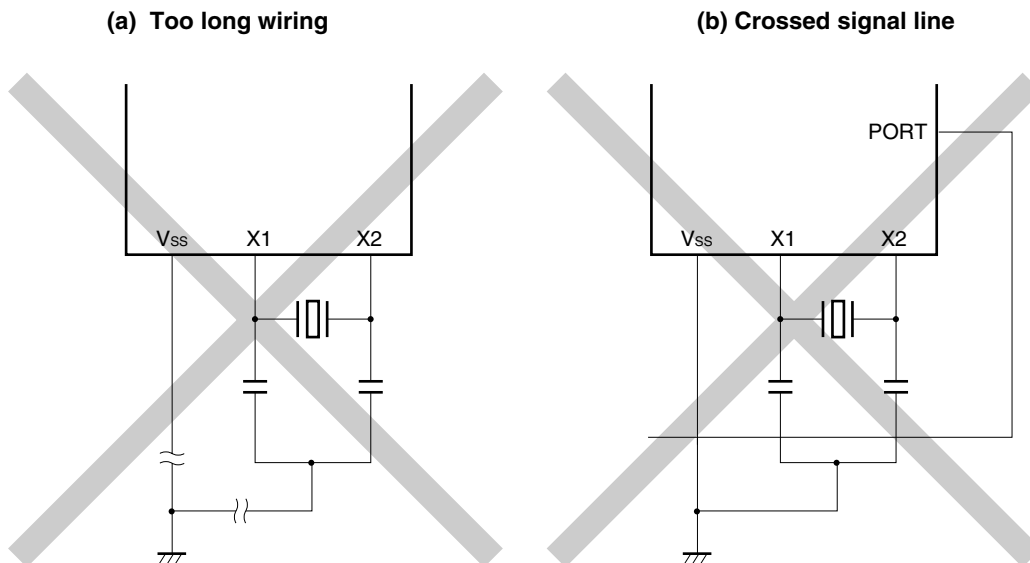
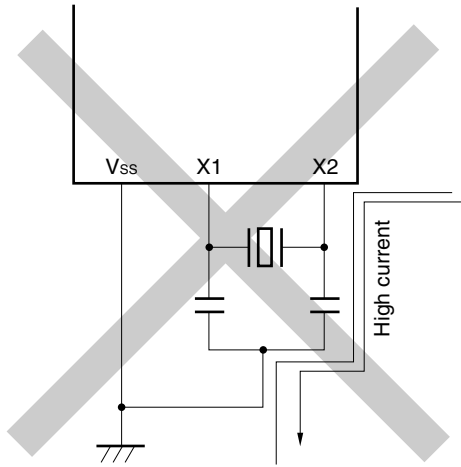
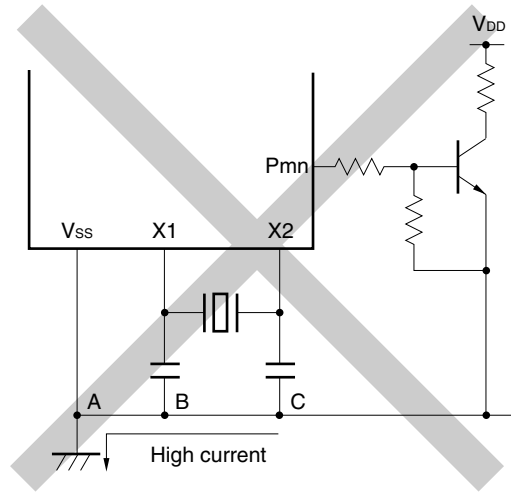


Figure 5-10. Examples of Incorrect Resonator Connection (2/2)

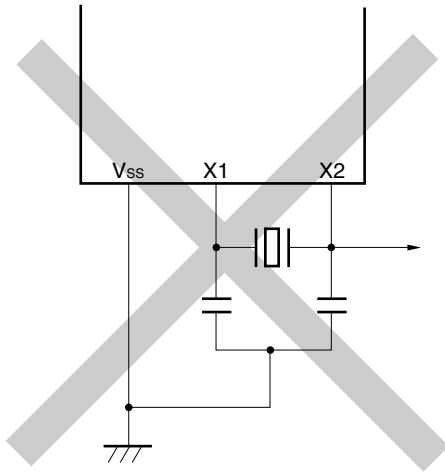
(c) Wiring near high alternating current



(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  $\mu$ PD179F11x, 179F12x microcontrollers. Oscillation can be controlled by the internal oscillation mode register (RCM).

After a reset release, the internal high-speed oscillator automatically starts oscillation ( $4 \text{ MHz} \pm 2\%$ ).

#### 5.4.3 Internal low-speed oscillator

The internal low-speed oscillator is incorporated in the  $\mu$ PD179F11x, 179F12x microcontrollers.

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).

<R> 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 various clocks 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  $\mu$ PD179F11x, 179F12x microcontrollers, thus enabling the following.

#### (1) Enhancement of security function

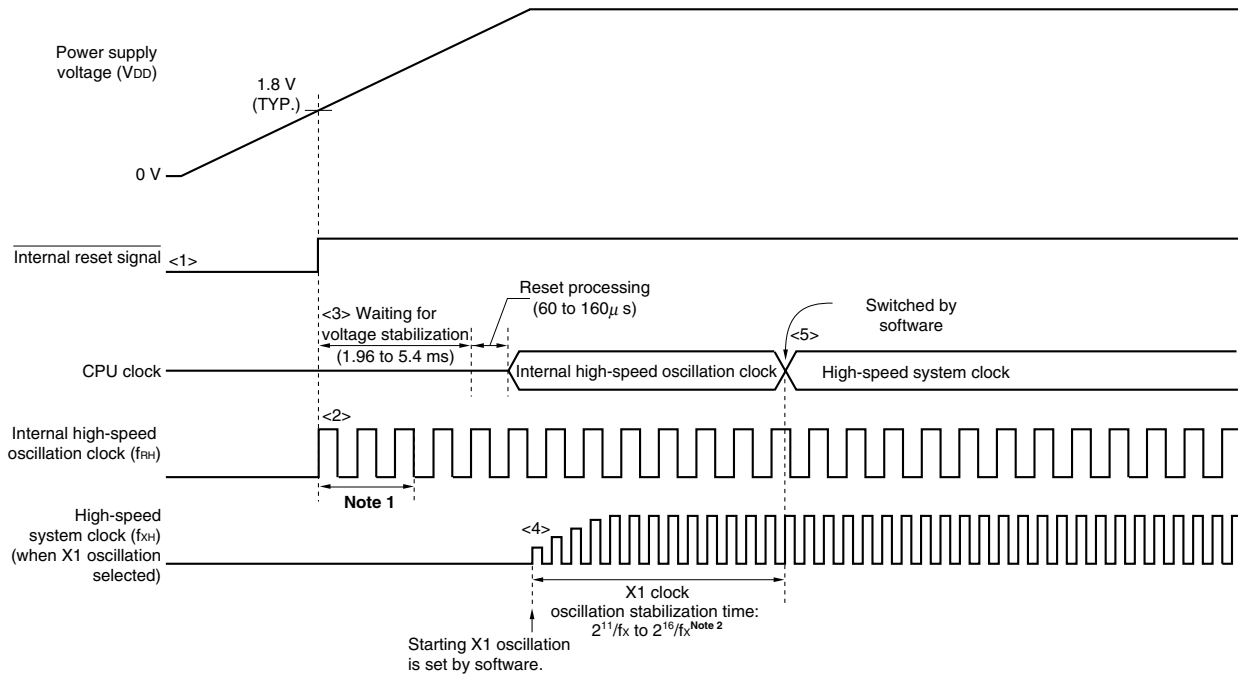
When the X1 clock is set as the CPU clock by the default setting, the device cannot operate if the X1 clock is damaged or badly connected and therefore does not operate after reset is released. However, the start clock of the CPU is the internal high-speed oscillation clock, so the device can be started 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



- <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 1.8 V (TYP.), the reset is released and the internal high-speed oscillator automatically starts oscillation.
- <3> 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 or XT1 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. The internal voltage stabilization time includes the oscillation accuracy stabilization time of the internal high-speed oscillation clock.
  2. 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**).



## 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/OCD0A and P122/X2/EXCLK/OCD0B 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.

- Cautions**
1. Do not change the value of EXCLK and OSCSEL while the X1 clock is operating.
  2. Set the X1 clock after the supply voltage has reached the operable voltage of the clock to be used (see CHAPTER 21 ELECTRICAL SPECIFICATIONS).

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

<1> Setting P121/X1/OCD0A and P122/X2/EXCLK/OCD0B 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.

**Cautions 1. Do not change the value of EXCLK and OSCSEL while the external main system clock is operating.**

**2. Set the external main system clock after the supply voltage has reached the operable voltage of the clock to be used (see CHAPTER 21 ELECTRICAL SPECIFICATIONS).**

**(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<sup>Note</sup>

(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 <sub>XP</sub> )	Peripheral Hardware Clock (f <sub>PRS</sub> )
1	1	High-speed system clock (f <sub>XH</sub> )	High-speed system clock (f <sub>XH</sub> )

**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> Setting the main system clock as the CPU clock and selecting the division ratio (PCC register)

When CSS is cleared to 0, the main system clock is supplied to the CPU. To select the CPU clock division ratio, use PCC0, PCC1, and PCC2.

PCC2	PCC1	PCC0	CPU Clock (f <sub>CPU</sub> ) Selection	
			FLMD0 = 0	FLMD0 = 1
0	0	0	f <sub>XP</sub>	f <sub>RH</sub>
0	0	1	f <sub>XP</sub> /2 (default)	f <sub>RH</sub> /2 (default)
0	1	0	f <sub>XP</sub> /2 <sup>2</sup>	f <sub>RH</sub> /2 <sup>2</sup>
0	1	1	f <sub>XP</sub> /2 <sup>3</sup>	f <sub>RH</sub> /2 <sup>3</sup>
1	0	0	f <sub>XP</sub> /2 <sup>4</sup>	f <sub>RH</sub> /2 <sup>4</sup>
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 13 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 (PCC and MCM registers)  
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 the internal high-speed oscillation 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<sup>Note 1</sup>**

- <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<sup>Note 2</sup>.

**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<sup>Note</sup>  
 (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<sup>Note</sup>  
 (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 <sub>XP</sub> )	Peripheral Hardware Clock (f <sub>PRS</sub> )
0	0	Internal high-speed oscillation clock (f <sub>RH</sub> )	Internal high-speed oscillation clock (f <sub>RH</sub> )
0	1		High-speed system clock (f <sub>XH</sub> )
1	0		

- <3> Selecting the CPU clock division ratio (PCC register)  
 When CSS is cleared to 0, the main system clock is supplied to the CPU. To select the CPU clock division ratio, use PCC0, PCC1, and PCC2.

PCC2	PCC1	PCC0	CPU Clock (f <sub>CPU</sub> ) Selection	
			FLMD0 = 0	FLMD0 = 1
0	0	0	f <sub>XP</sub>	f <sub>RH</sub>
0	0	1	f <sub>XP</sub> /2 (default)	f <sub>RH</sub> /2 (default)
0	1	0	f <sub>XP</sub> /2 <sup>2</sup>	f <sub>RH</sub> /2 <sup>2</sup>
0	1	1	f <sub>XP</sub> /2 <sup>3</sup>	f <sub>RH</sub> /2 <sup>3</sup>
1	0	0	f <sub>XP</sub> /2 <sup>4</sup>	f <sub>RH</sub> /2 <sup>4</sup>
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 13 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 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 (PCC and MCM registers)

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 the high-speed system 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.

<R>

**(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

- Remarks**
1. XSEL: Bit 2 of the main clock mode register (MCM)
  2. MCM0: Bit 0 of MCM
  3. EXCLK: Bit 7 of the clock operation mode select register (OSCCTL)
  4. ×: don't care

5.6.5 CPU clock status transition diagram

Figure 5-12 shows the CPU clock status transition diagram of this product.

Figure 5-12. CPU Clock Status Transition Diagram

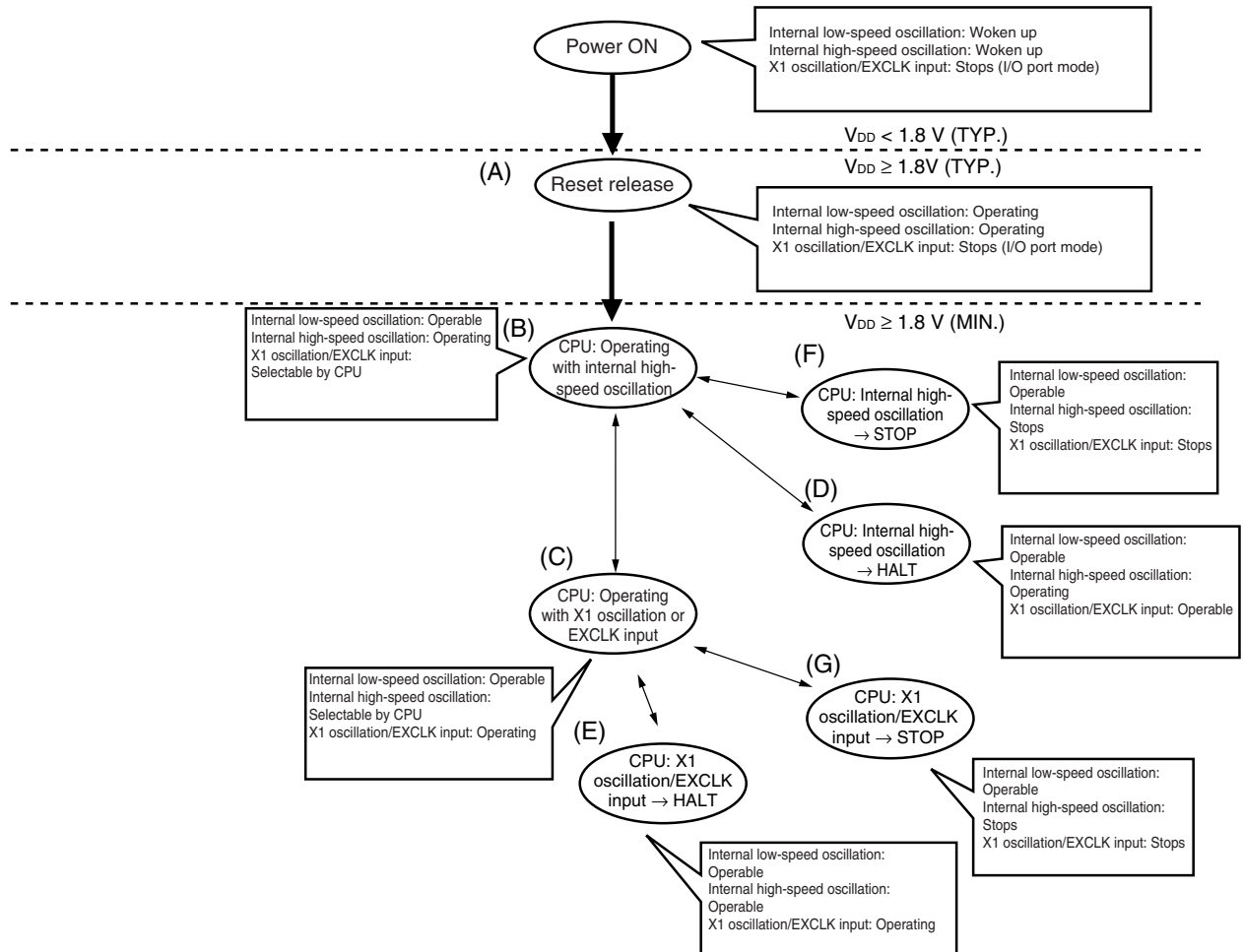


Table 5-4 shows transition of the CPU clock and examples of setting the SFR registers.

**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: $1 \text{ MHz} \leq f_{XH} \leq 4 \text{ MHz}$ )	0	1	0	Must be checked	1	1
(A) → (B) → (C) (external main clock: $1 \text{ MHz} \leq f_{XH} \leq 4 \text{ MHz}$ )	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 <sup>Note</sup>	MCM0
(B) → (C) (X1 clock: $1 \text{ MHz} \leq f_{XH} \leq 4 \text{ MHz}$ )	0	1	0	Must be checked	1	1
(B) → (C) (external main clock: $1 \text{ MHz} \leq f_{XH} \leq 4 \text{ MHz}$ )	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

**Note** The value of this flag can be changed only once after a reset release. This setting is not necessary if it has already been set.

**Caution** Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 21 ELECTRICAL SPECIFICATIONS).

- Remarks**
- (A) to (G) in Table 5-4 correspond to (A) to (G) in Figure 5-12.
  - 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

<R>

**Caution** Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 21 ELECTRICAL SPECIFICATIONS).

(5) HALT mode (D) set while CPU is operating with internal high-speed oscillation clock (B), HALT mode (E) set while CPU is operating with high-speed system clock (C)

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

(6) STOP mode (F) set while CPU is operating with internal high-speed oscillation clock (B), STOP mode (G) set while CPU is operating with high-speed system clock (C)

(Setting sequence) →

Status Transition	Setting	
(B) → (F) (C) → (G)	Stopping peripheral functions that cannot operate in STOP mode	Executing STOP instruction

- Remarks**
- (A) to (G) in Table 5-4 correspond to (A) to (G) in Figure 5-12.
  - MCM0: Bit 0 of the main clock mode register (MCM)  
RSTS, RSTOP: Bits 7 and 0 of the internal oscillation mode register (RCM)

**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 • 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 • 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 • 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 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	/			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		
0	1	1				2 clocks			2 clocks			2 clocks			2 clocks		
1	0	0				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	$1 + 2f_{XH}/f_{RH}$ clock	

**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  $f_{RH} = 1$  MHz,  $f_{XH} = 4$  MHz)

$$1 + 2f_{RH}/f_{XH} = 1 + 2 \times 1/4 = 1 + 2 \times 0.25 = 1 + 0.5 = 1.5 \rightarrow 1 \text{ clock}$$

### 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 a clock other than the internal high-speed oscillation clock)	RSTOP = 1
X1 clock	MCS = 0 (The CPU is operating on a clock other than the high-speed system 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  $\mu$ PD179F11x, 179F12x microcontrollers.

**Table 5-9. Peripheral Hardware and Source Clocks**

Source Clock		Peripheral Hardware Clock (f <sub>PRS</sub> )	Internal Low-Speed Oscillation Clock (f <sub>RL</sub> )	TM50 Output	External Clock from Peripheral Hardware Pins
Peripheral Hardware					
16-bit timer/ event counter 00		Y	N	N	Y (TI000 pin)
8-bit timer/ event counter	50	Y	N	N	Y (TI50 pin)
	51	Y	N	N	Y (TI51 pin)
8-bit timer	H0	Y	N	Y	N
	H1	Y	Y	N	N
Watchdog timer		N	Y	N	N
Clock output <sup>Note 2</sup>		Y	N	N	N
Serial interface	UART6	Y	N	Y	N

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

### 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

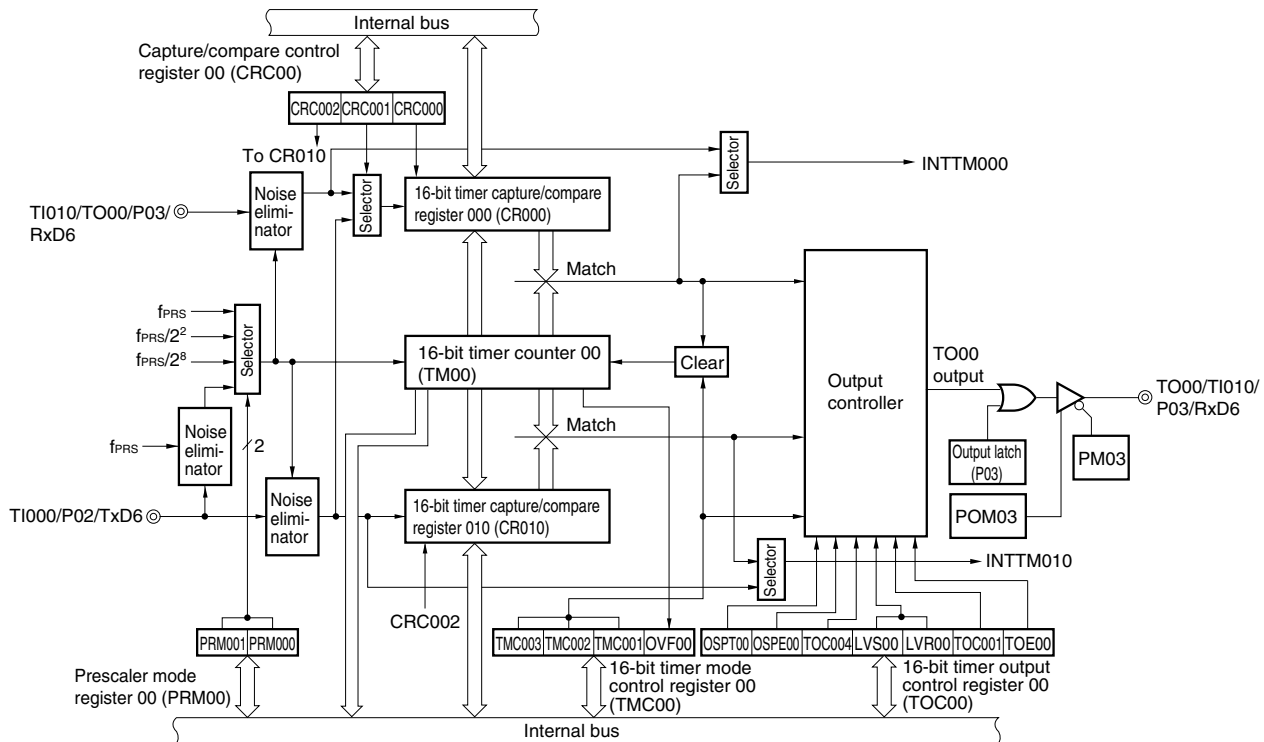
16-bit timer/event counter 00 includes the following hardware.

**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 pins
Timer output	TO00 pin, 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) Port output mode register 0 (POM0)

Figure 6-1 shows the block diagram.

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



**Caution 1.** The valid edge of TI010 and timer output (TO00) cannot be used for the P03 pin at the same time. Select either of the functions.

**Cautions 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.

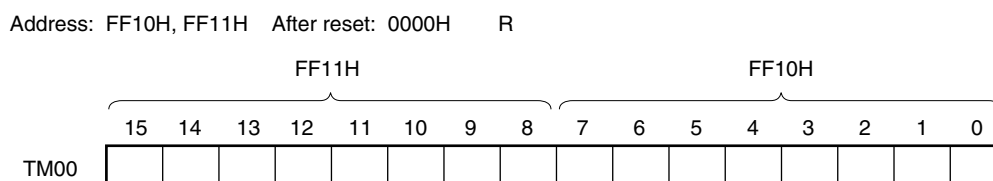
### (1) 16-bit timer counter 00 (TM00)

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.

If the count value is read during operation, then input of the count clock is temporarily stopped, and the count value at that point is read.

**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

**Cautions 1.** Even if TM00 is read, the value is not captured by CR010.

2. If TM00 is referred to during a timer count, a timer count will be stopped during reference processing, and a timer count is resumed after reference processing is finished.

Therefore, if processing which refers to TM00 is performed, an error will arise at a timer count.

### (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.

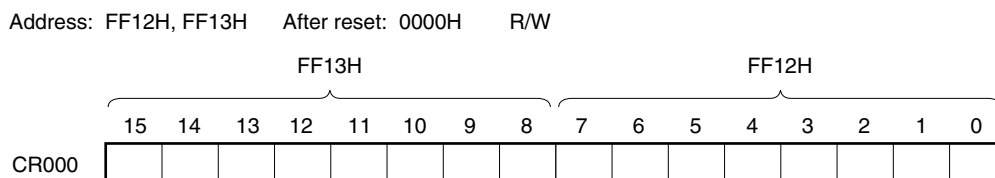
Change the value of CR000 while the timer is stopped (TMC003 and TMC002 = 00).

The value of CR010 can be changed during operation if the value has been set in a specific way. For details, see

#### 6.5.1 Rewriting CR010 during TM00 operation.

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

Reset signal generation sets 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**

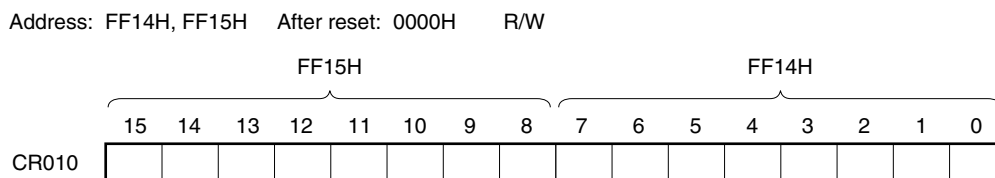
The value set in CR000 is constantly compared with the TM00 count value, and an interrupt request signal (INTTM000) is generated if they match. The value is held until CR000 is rewritten.

**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.



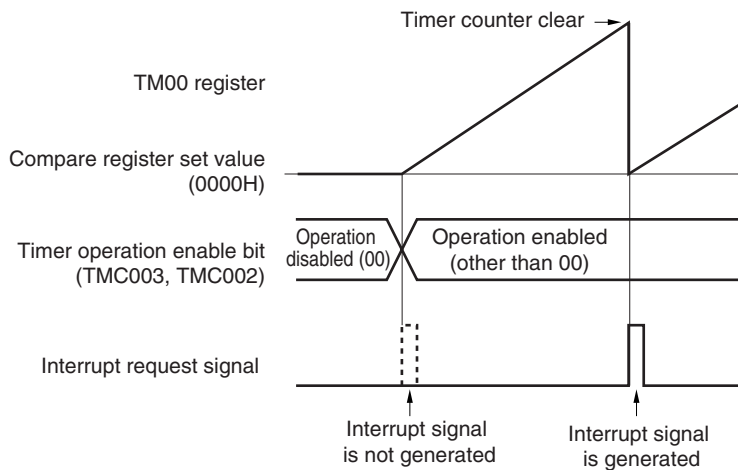
**(iii) Setting range when CR000 or CR010 is used as a compare register**

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 <sup>Note</sup> ≤ 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 <sup>Note</sup> ≤ N ≤ FFFFH	0000H <sup>Note</sup> ≤ M ≤ FFFFH
Operation as free-running timer		
Operation as PPG output	M < N ≤ FFFFH	0000H <sup>Note</sup> ≤ M < N
Operation as one-shot pulse output	0000H <sup>Note</sup> ≤ N ≤ FFFFH (N ≠ M)	0000H <sup>Note</sup> ≤ 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.

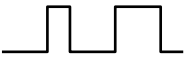

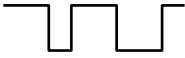


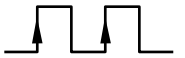
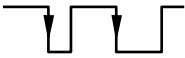


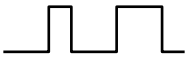
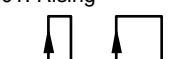


- When the timer counter is cleared due to overflow
- When the timer counter is cleared due to TI000 pin valid edge (when clear & start mode is entered by TI000 pin valid edge input)
- When the timer counter is cleared due to compare match (when clear & start mode is entered by match between TM00 and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



**Remarks 1.** N: CR000 register set value, M: CR010 register set value

**2.** For details of TMC003 and TMC002, see **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 of CR000	CRC001 = 1 TI000 pin input (reverse phase) 	Set values of ES001 and ES000 Position of edge to be captured	CRC001 bit = 0 TI010 pin input 	Set values of ES101 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 
	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 <sup>Note</sup> 	Set values of ES001 and ES000 Position of edge to be captured		
		01: Rising 		
		00: Falling 		
		11: Both edges 		
	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: See 6.3 (2) Capture/compare control register 00 (CRC00).  
ES101, ES100, ES001, ES000: See 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)
- Port output mode register 0 (POM0)

#### (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 sets TMC00 to 00H.

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

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

Address: FFBAH 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 <sup>Note</sup>
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"> <li>Match between TM00 and CR000 or match between TM00 and CR010</li> </ul>
1	<ul style="list-style-type: none"> <li>Match between TM00 and CR000 or match between TM00 and CR010</li> <li>Trigger input of TI000 pin valid edge</li> </ul>

OVF00	TM00 overflow flag
Clear (0)	Clears OVF00 to 0 or TMC003 and TMC002 = 00
Set (1)	Overflow occurs.
<p>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 &amp; start mode entered by TI000 pin valid edge input, and clear &amp; start mode entered upon a match between TM00 and CR000). It can also be set to 1 by writing 1 to OVF00.</p>	

**Note** The TI000 pin valid edge is set by bits 5 and 4 (ES001, 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: FFBC<sub>H</sub> 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

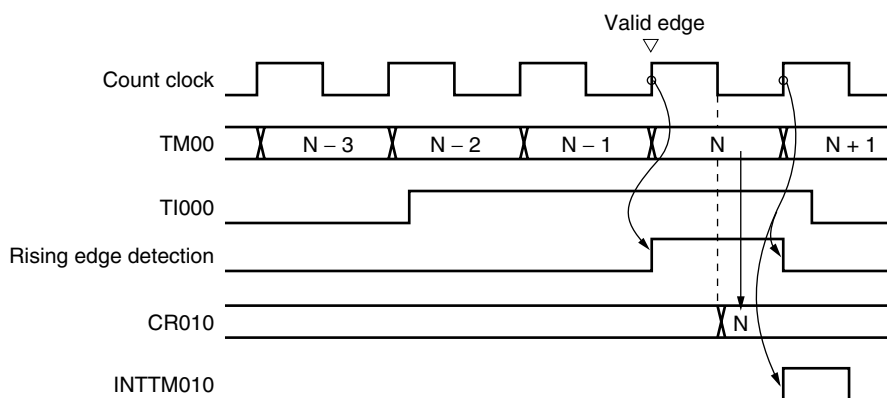
CRC001	CR000 capture trigger selection
0	Captures on valid edge of TI010 pin
1	Captures on valid edge of TI000 pin by reverse phase <sup>Note</sup>
The valid edge of the TI010 and TI000 pin is set by PRM00. If ES001 and ES000 are set to 11 (both edges) when CRC001 is 1, the valid edge of the TI000 pin 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 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 (see **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: FFBDH 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 TI000 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"> <li>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.</li> <li>Be sure to set LVS00 and LVR00 when TOE00 = 1. LVS00, LVR00, and TOE00 being simultaneously set to 1 is prohibited.</li> <li>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.</li> <li>The values of LVS00 and LVR00 are always 0 when they are read.</li> <li>For how to set LVS00 and LVR00, see <b>6.5.2 Setting LVS00 and LVR00</b>.</li> <li>The actual TO00/TI010/RxD6/P03 pin output is determined depending on PM03 and P03, besides TO00 output.</li> </ul>								
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 P03 pin at the same time. Select either of the functions.



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

Address: FFBBH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PRM00	ES101	ES100	ES001	ES000	0	0	PRM001	PRM000

ES101	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

ES001	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		
			$f_{PRS} = 2 \text{ MHz}$	$f_{PRS} = 4 \text{ MHz}$
0	0	$f_{PRS}$	2 MHz	4 MHz
0	1	$f_{PRS}/2^2$	500 kHz	1 MHz
1	0	$f_{PRS}/2^8$	7.81 kHz	15.625 kHz
1	1	TI000 valid edge <sup>Note</sup>		

**Note** The external clock from the TI000 pin requires a pulse longer than twice the cycle of the peripheral hardware clock ( $f_{PRS}$ ).

**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 P03/TO00/TI010/RxD6 pin for timer output, set PM03 and the output latches of P03 to 0.

When using the P02/TI000/TxD6 and P03/TO00/TI010/RxD6 pins for timer input, set PM02 and PM03 to 1. At this time, the output latches of P02 and P03 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	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00

PM0n	P0n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

**(6) Port output mode resistors (POM0)**

This register set the output mode of port 0.

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

Reset signal generation sets these registers to 00H.

**Figure 6-11. Format of Pull-up Resistor Option Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POM0	POM07	POM06	POM05	POM04	POM03	POM02	POM01	POM00	FF38H	00H	R/W

POM0n	P0n pin output mode selection (n = 0 to 7)
0	CMOS output
1	N-ch open-drain output (P07:P-ch open-drain output)

## 6.4 Operation of 16-bit Timer/Event Counter 00

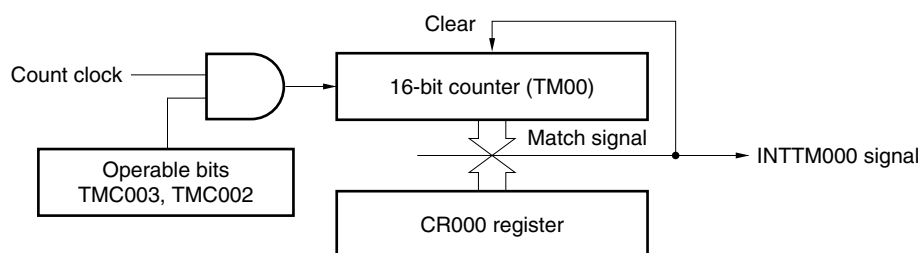
### 6.4.1 Interval timer operation

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, see **6.3 (5) Port mode register 0 (PM0)** and **(6) Port output mode resistors (POM0)**.
- 2.** For how to enable the INTTM000 interrupt, see **CHAPTER 11 INTERRUPT FUNCTIONS**.

**Figure 6-12. Block Diagram of Interval Timer Operation**



**Figure 6-13. Basic Timing Example of Interval Timer Operation**

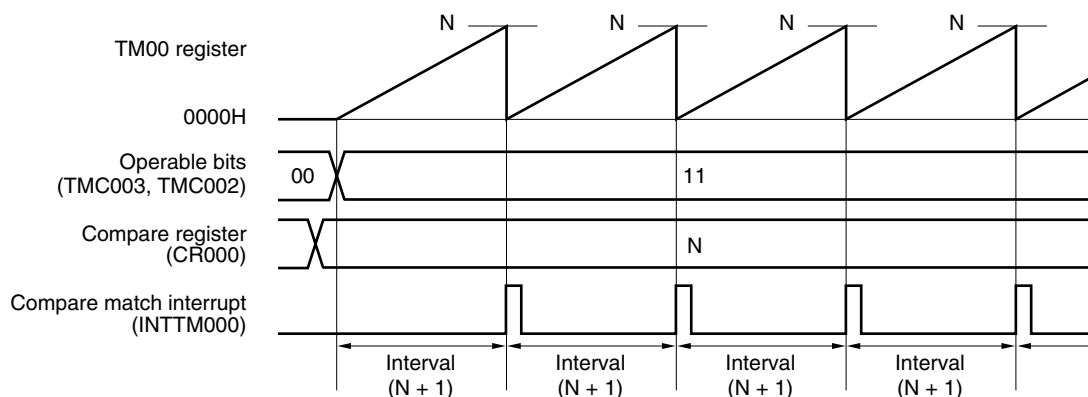


Figure 6-14. Example of Register Settings for Interval Timer Operation

## (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	0	0	0

## (d) Prescaler mode register 00 (PRM00)

ES101	ES100	ES001	ES000	3	2	PRM001	PRM000
0	0	0	0	0	0	0/1	0/1

Selects count clock

## (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 =  $(M + 1) \times$  Count clock cycle

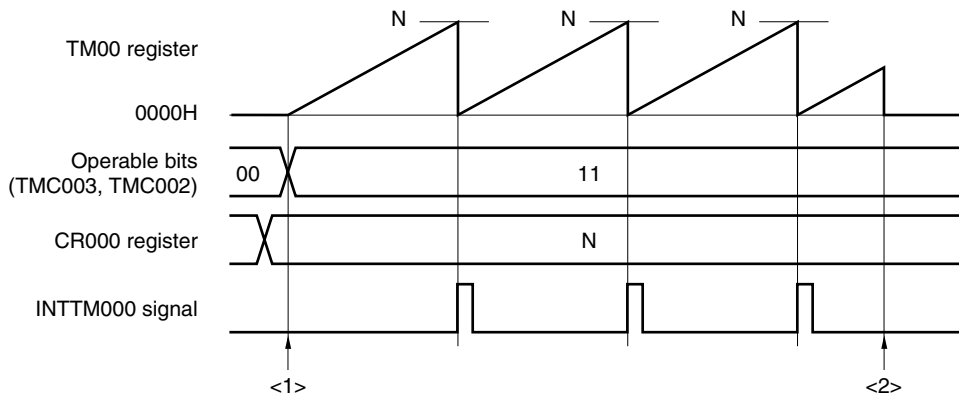
Setting CR000 to 0000H is prohibited.

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

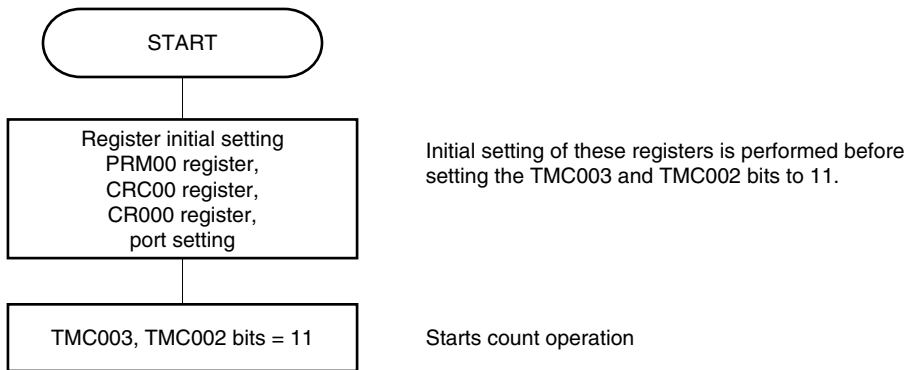
Usually, CR010 is not used for the interval timer function. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

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

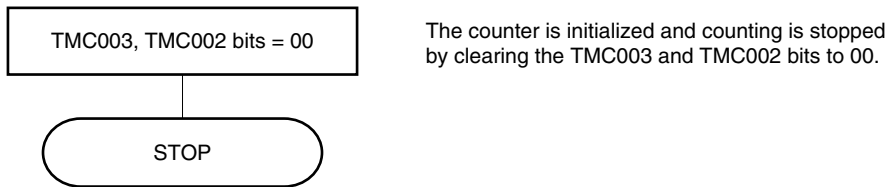
Figure 6-15. Example of Software Processing for Interval Timer Function



<1> Count operation start flow



<2> Count operation stop flow



### 6.4.2 Square wave output operation

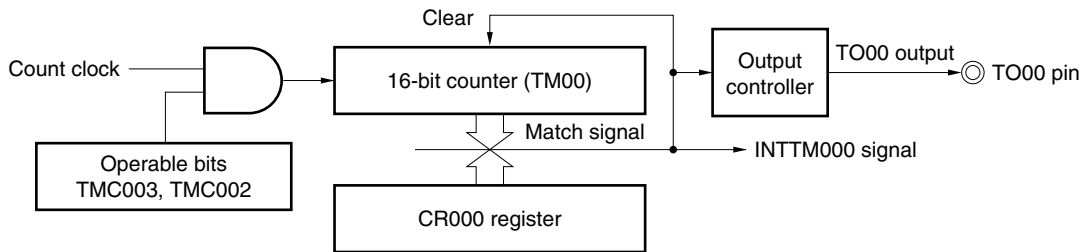
When 16-bit timer/event counter 00 operates as an interval timer (see 6.4.1), a square wave can be output from the TO00 pin by setting the 16-bit timer output control register 00 (TOC00) to 03H.

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 TO00 to output a square wave.

- Remarks 1.** For the setting of I/O pins, see 6.3 (5) **Port mode register 0 (PM0)** and (6) **Port output mode resistors (POM0)**.
- 2.** For how to enable the INTTM000 signal interrupt, see **CHAPTER 11 INTERRUPT FUNCTIONS**.

**Figure 6-16. Block Diagram of Square Wave Output Operation**



**Figure 6-17. Basic Timing Example of Square Wave Output Operation**

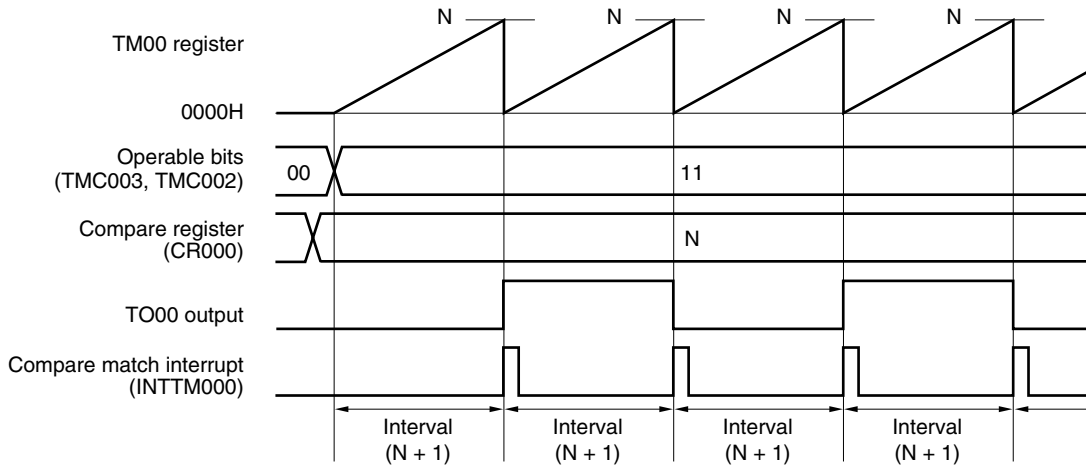
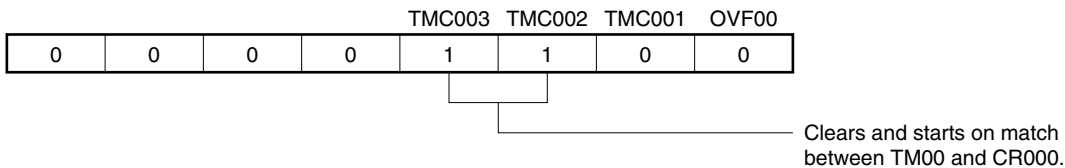
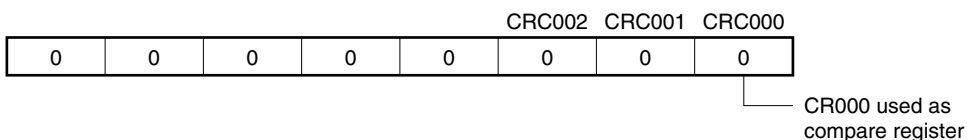


Figure 6-18. Example of Register Settings for Square Wave Output Operation

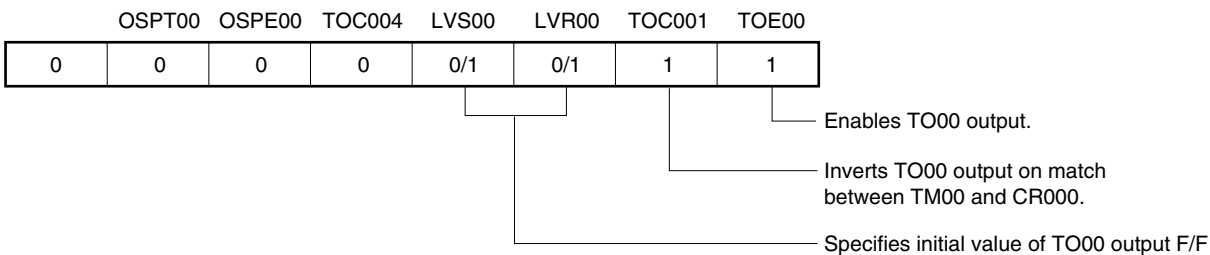
## (a) 16-bit timer mode control register 00 (TMC00)



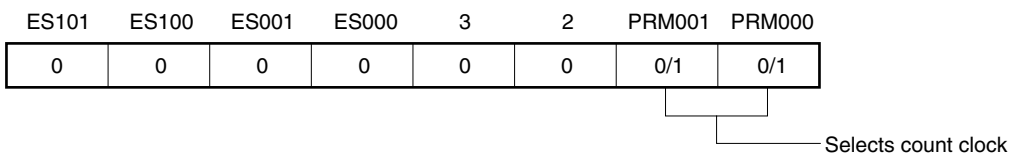
## (b) Capture/compare control register 00 (CRC00)



## (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.

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

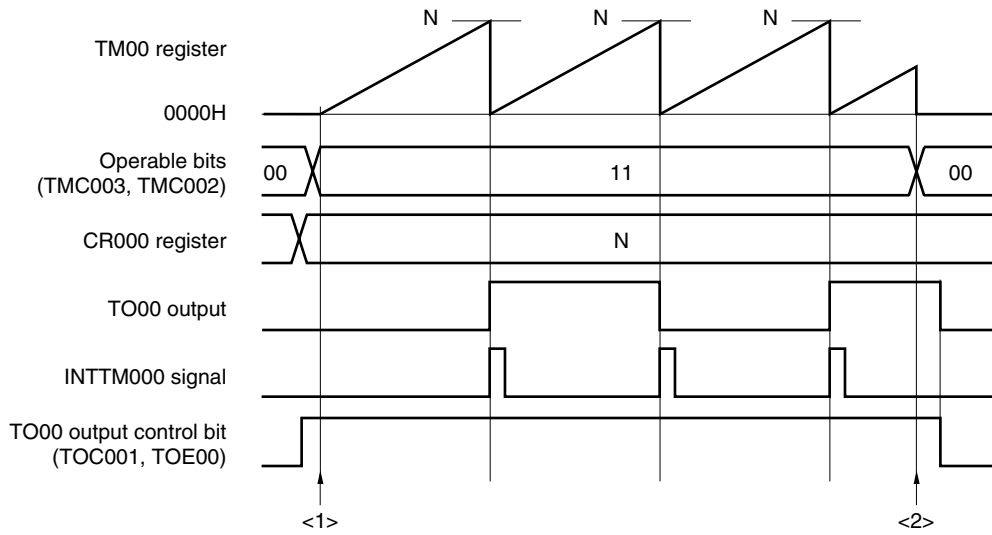
Setting CR000 to 0000H is prohibited.

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

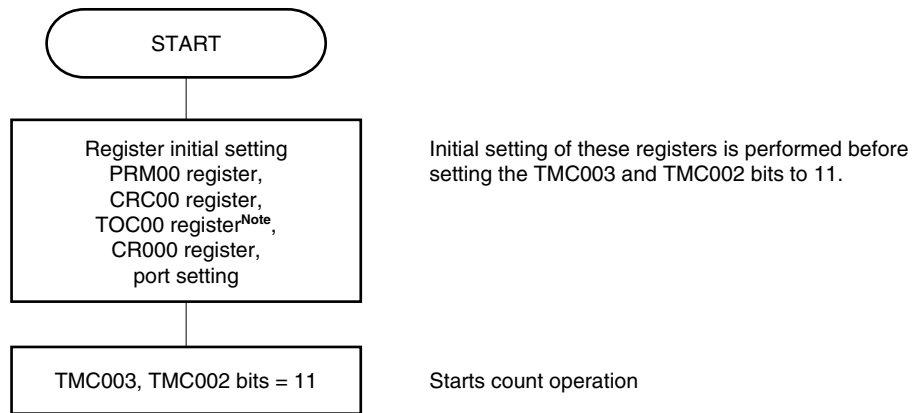
Usually, CR010 is not used for the square wave output function. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

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

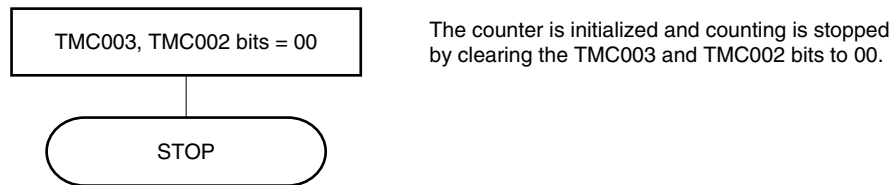
Figure 6-19. Example of Software Processing for Square Wave Output Function



<1> Count operation start flow



<2> Count operation stop flow



**Note** Care must be exercised when setting TOC00. For details, see 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  $\times$  (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  $\times$  (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, see **6.3 (5) Port mode register 0 (PM0)**.

**2.** For how to enable the INTTM000 signal interrupt, see **CHAPTER 11 INTERRUPT FUNCTIONS**.

**Figure 6-20. Block Diagram of External Event Counter Operation**

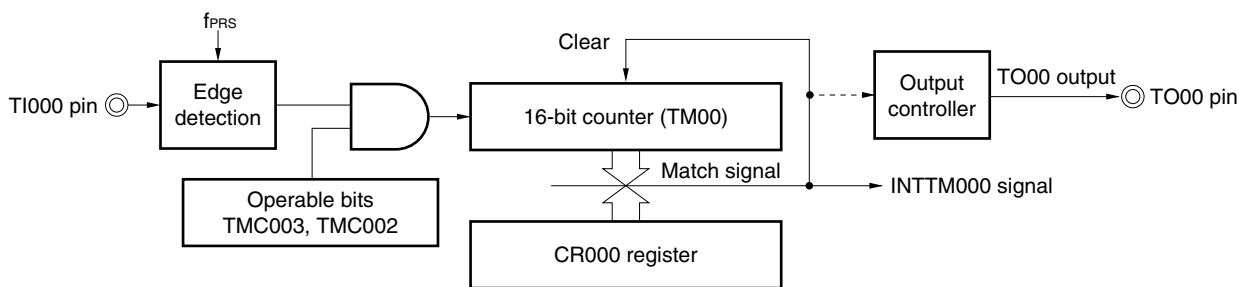
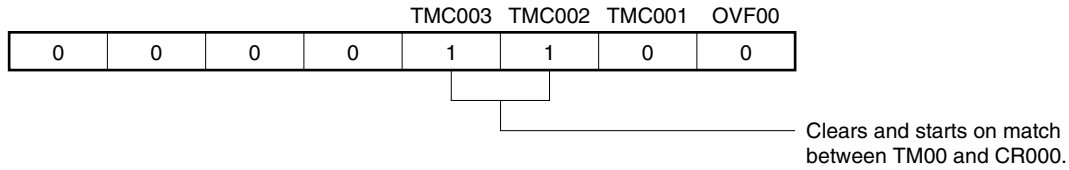
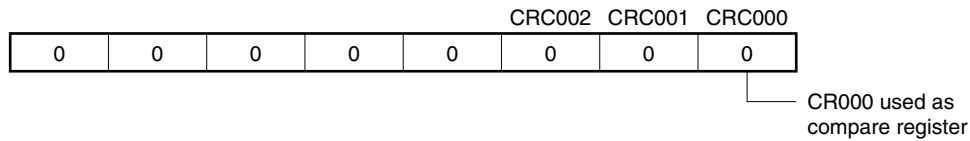
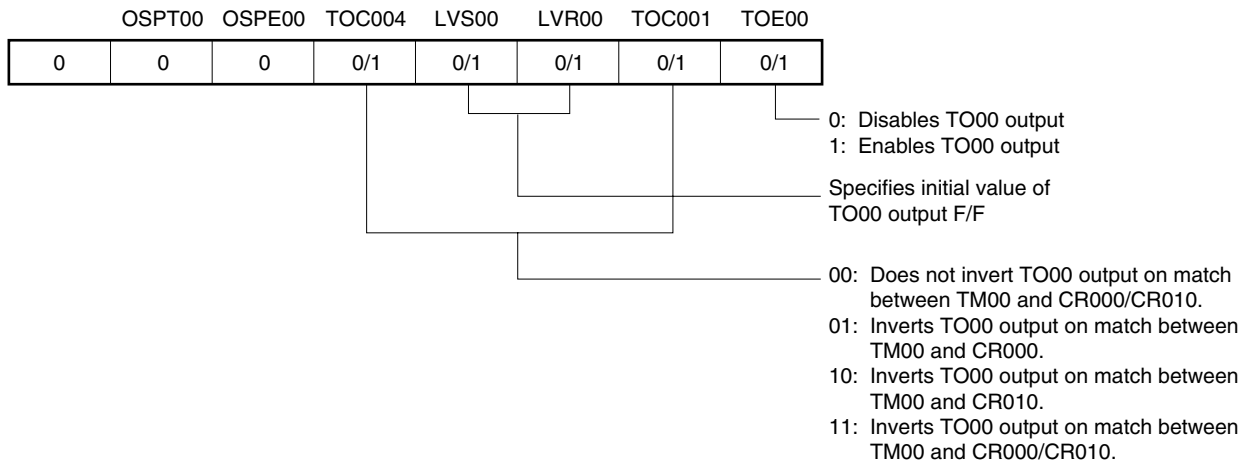
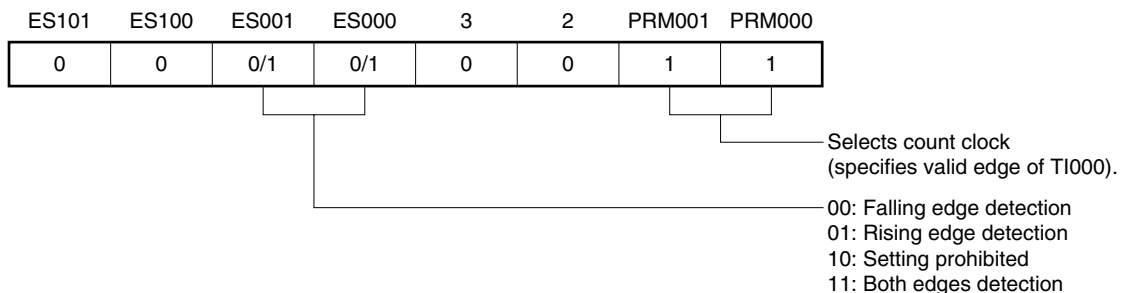


Figure 6-21. Example of Register Settings in External Event Counter Mode (1/2)

**(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)****(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.

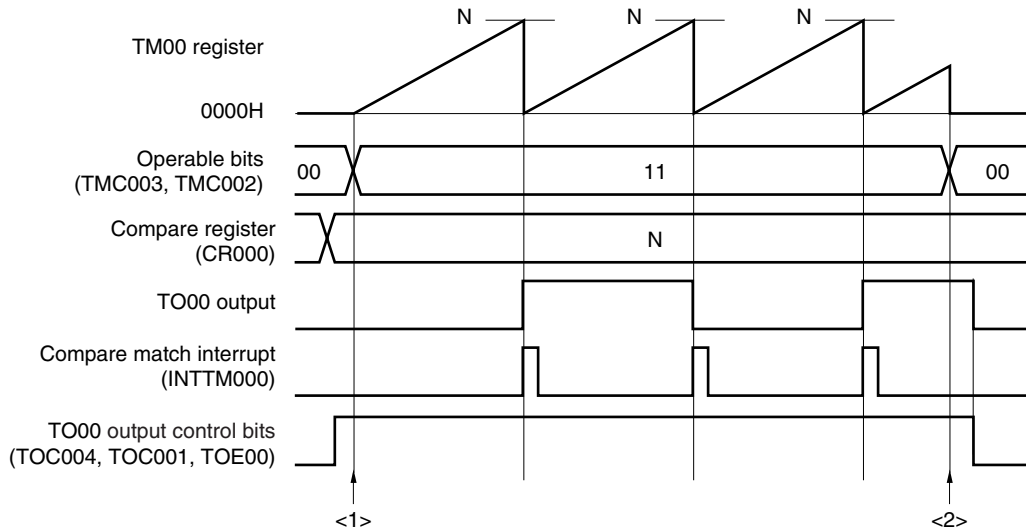
Figure 6-21. Example of Register Settings in External Event Counter Mode (2/2)

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

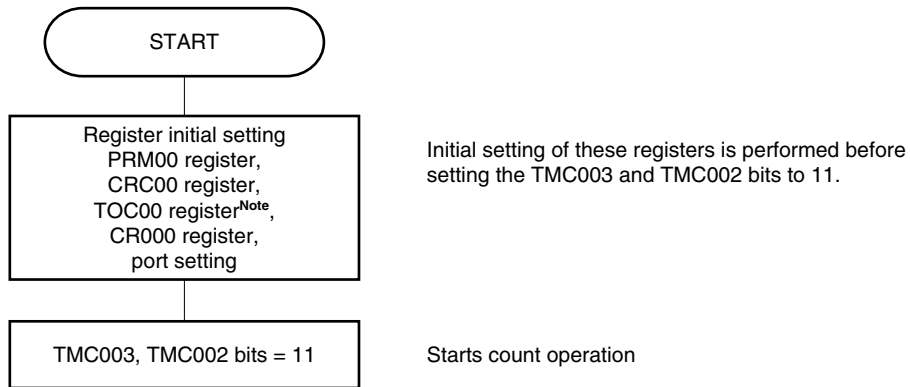
Usually, CR010 is not used in the external event counter mode. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

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

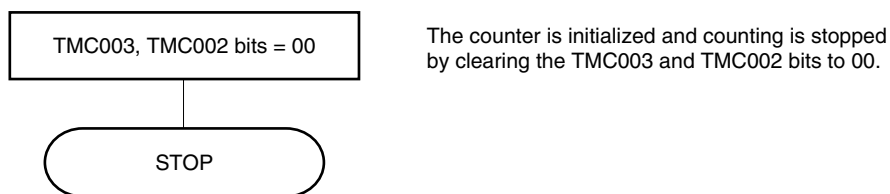
Figure 6-22. Example of Software Processing in External Event Counter Mode



<1> Count operation start flow



<2> Count operation stop flow



**Note** Care must be exercised when setting TOC00. For details, see 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.

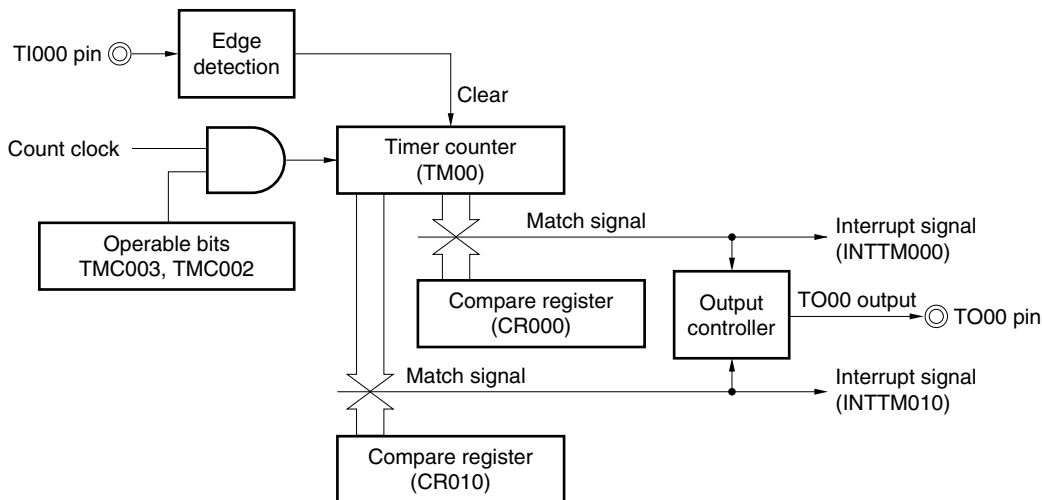
**Remarks 1.** For the setting of the I/O pins, see 6.3 (5) Port mode register 0 (PM0).

**2.** For how to enable the INTTM000 signal interrupt, see CHAPTER 11 INTERRUPT FUNCTIONS.

#### (1) Operation in clear & start mode entered by TI000 pin valid edge input

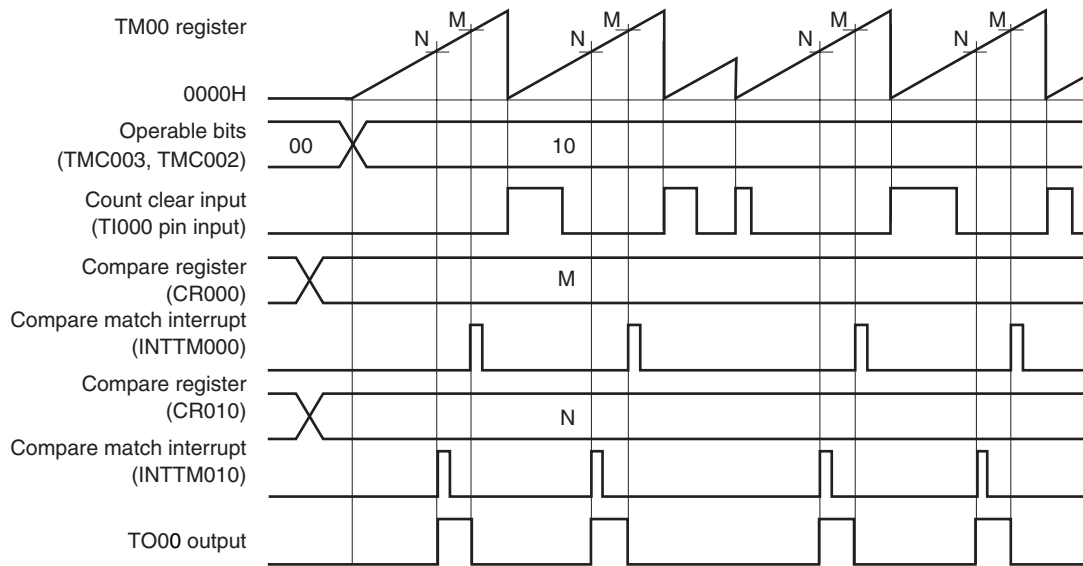
(CR000: compare register, CR010: compare register)

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

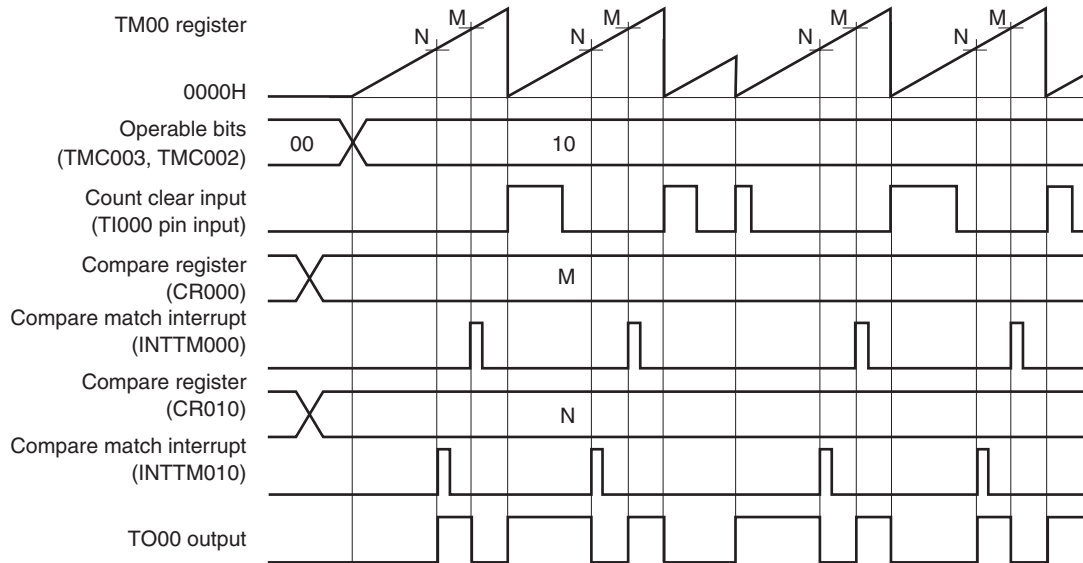


**Figure 6-24. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input (CR000: Compare Register, CR010: Compare Register)**

**(a) TOC00 = 13H, PRM00 = 10H, CRC00, = 00H, TMC00 = 08H**



**(b) TOC00 = 13H, PRM00 = 10H, CRC00, = 00H, TMC00 = 0AH**



(a) and (b) differ as follows depending on the setting of bit 1 (TMC001) of 16-bit timer mode control register 01 (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 6-25. Block Diagram of Clear & Start Mode Entered by TI000 Pin Valid Edge Input  
(CR000: Compare Register, CR010: Capture Register)

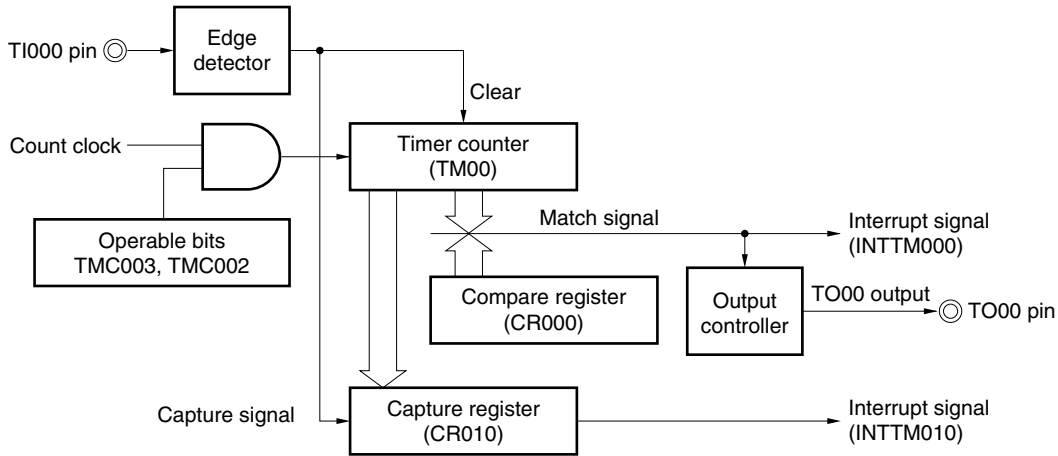
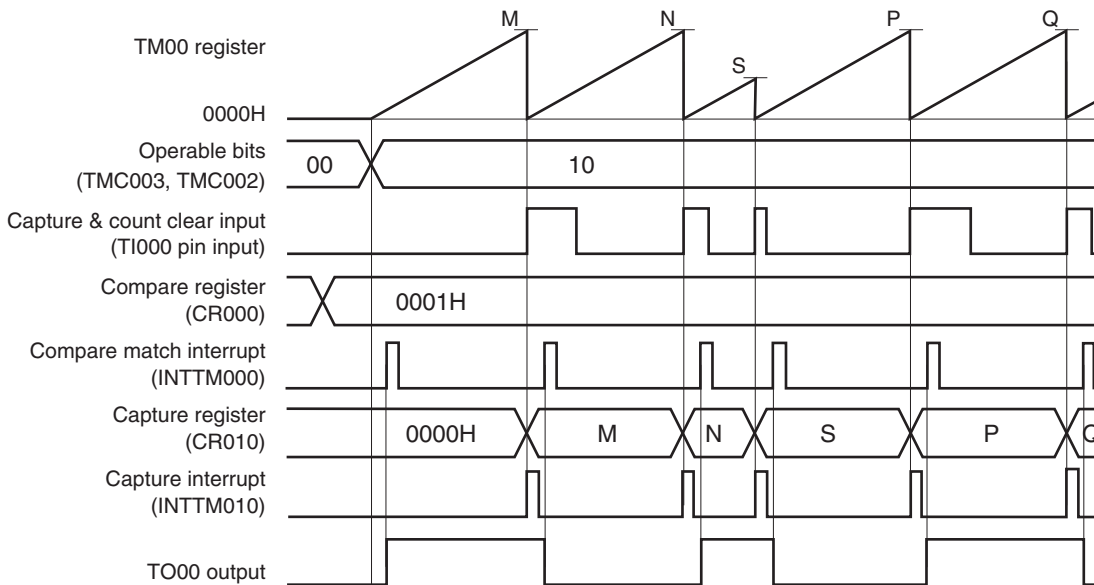


Figure 6-26. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input  
(CR000: Compare Register, CR010: Capture Register) (1/2)

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

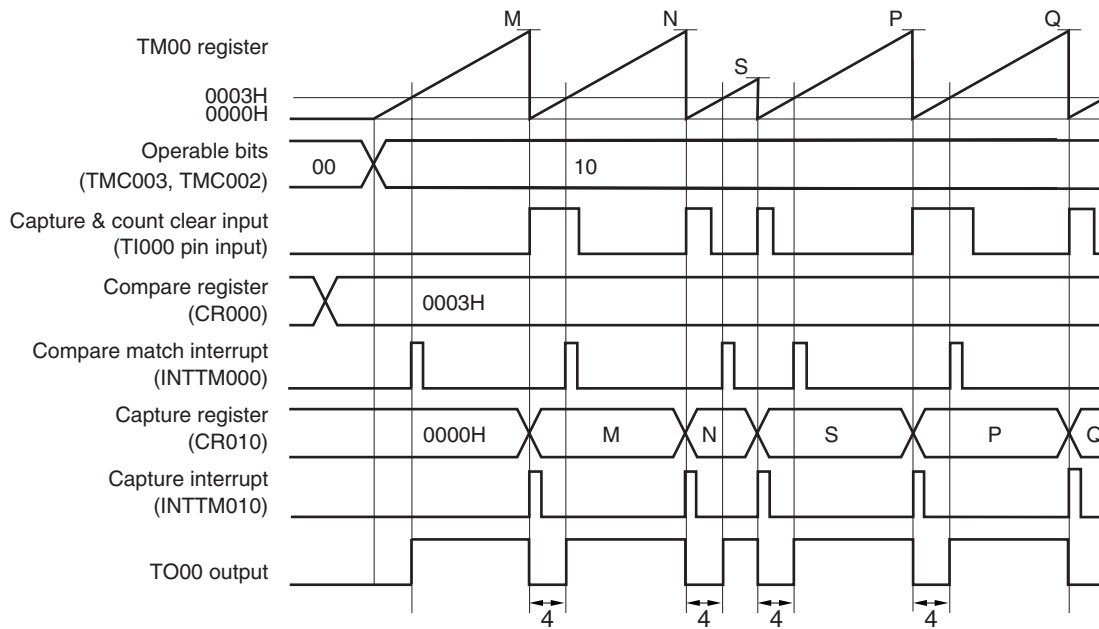


This is an application example where the TO00 output level is inverted when the count value has been captured & cleared.

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 6-26. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input (CR000: Compare Register, CR010: Capture Register) (2/2)**

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

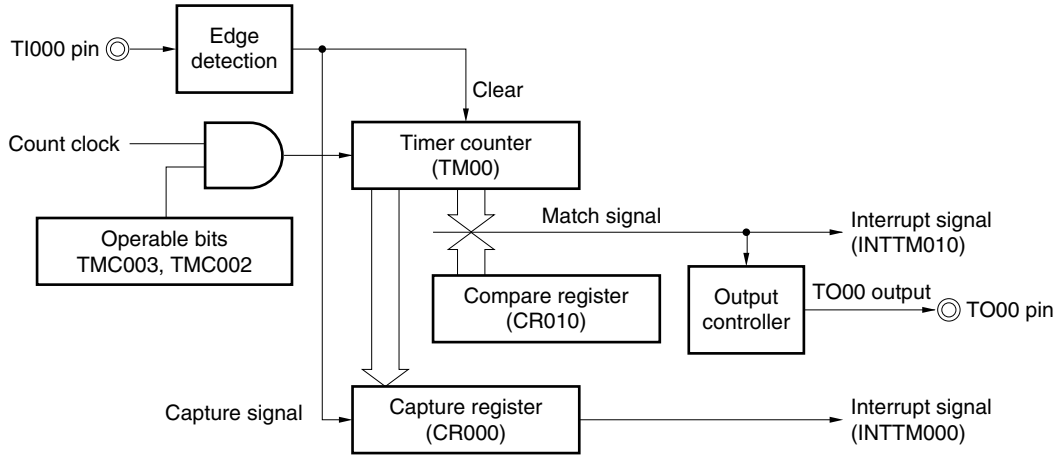


This is an application example where the width set to CR000 (4 clocks in this example) is to be output from the TO00 pin when the count value has been captured & cleared.

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 TI000 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 entered by TI000 pin valid edge input  
(CR000: capture register, CR010: compare register)

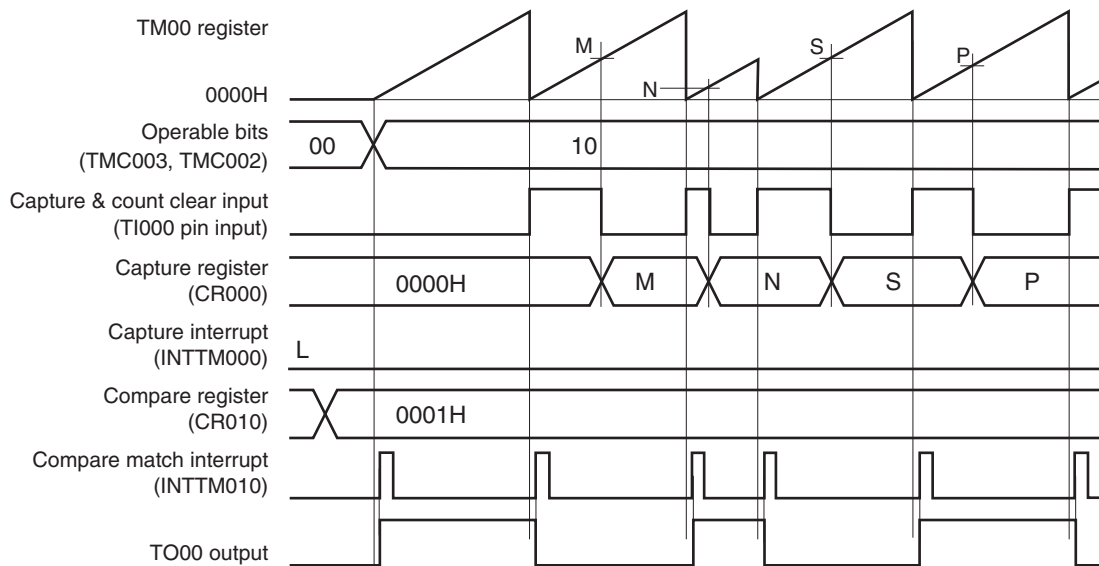
Figure 6-27. Block Diagram of Clear & Start Mode Entered by TI000 Pin Valid Edge Input  
(CR000: Capture Register, CR010: Compare Register)





**Figure 6-28. 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**



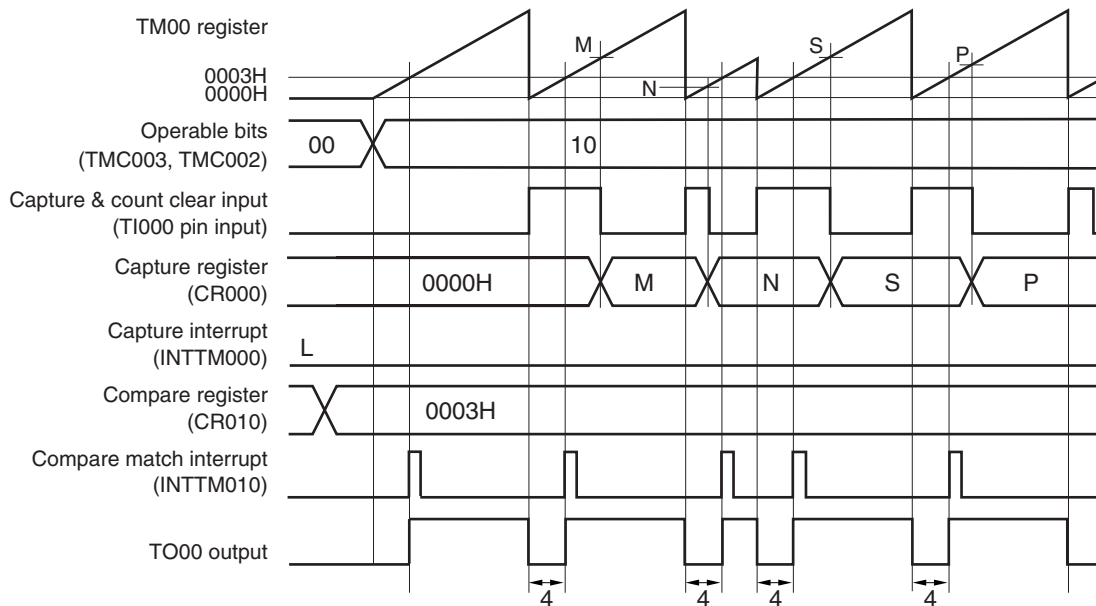
This is an application example where the TO00 output level is to be inverted when the count value has been captured & cleared.

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 6-28. Timing Example of Clear & Start Mode Entered by T1000 Pin Valid Edge Input (CR000: Capture Register, CR010: Compare Register) (2/2)**

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



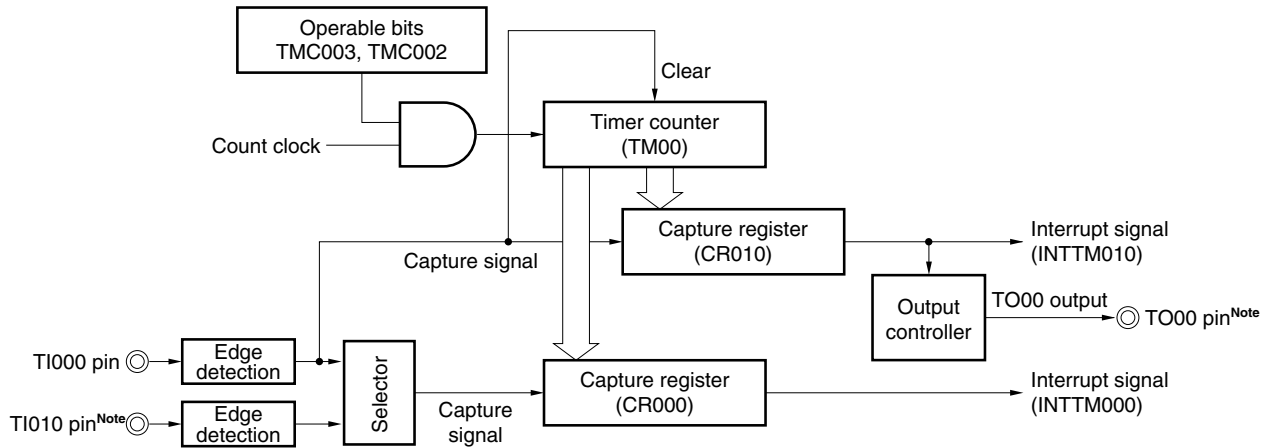
This is an application example where the width set to CR010 (4 clocks in this example) is to be output from the TO00 pin when the count value has been captured & cleared.

TM00 is cleared (to 0000H) at the rising edge detection of the T1000 pin and captured to CR000 at the falling edge detection of the T1000 pin. The TO00 output level is inverted when TM00 is cleared (to 0000H) because the rising edge of the T1000 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 T1000 pin, but the capture interrupt signal (INTTM000) is not generated. However, the INTTM000 interrupt is generated when the valid edge of the T1010 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)

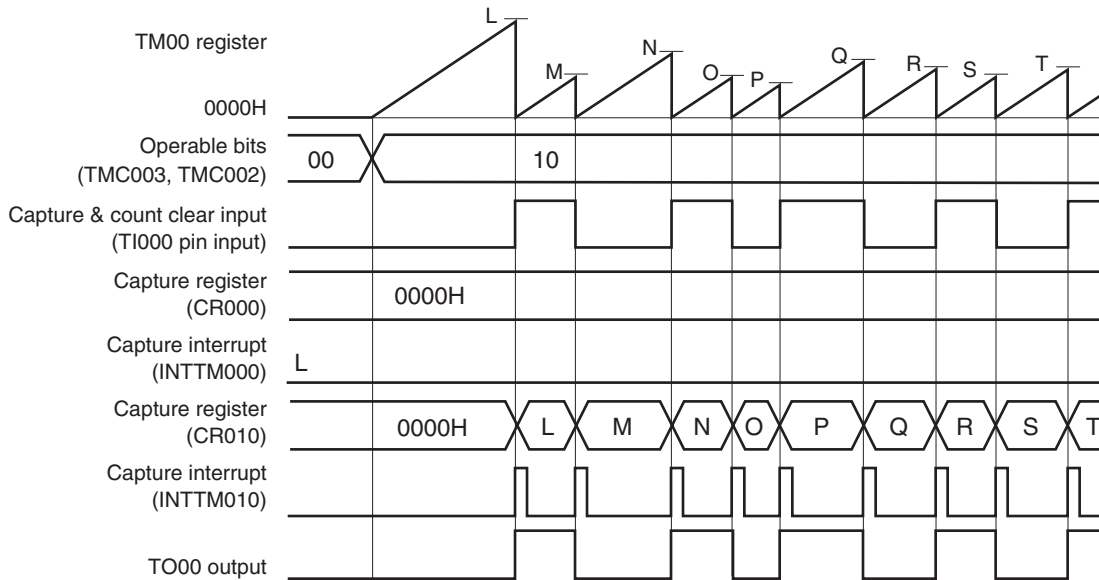
Figure 6-29. Block Diagram of Clear & Start Mode Entered by TI000 Pin Valid Edge Input  
(CR000: Capture Register, CR010: Capture Register)



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

Figure 6-30. 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

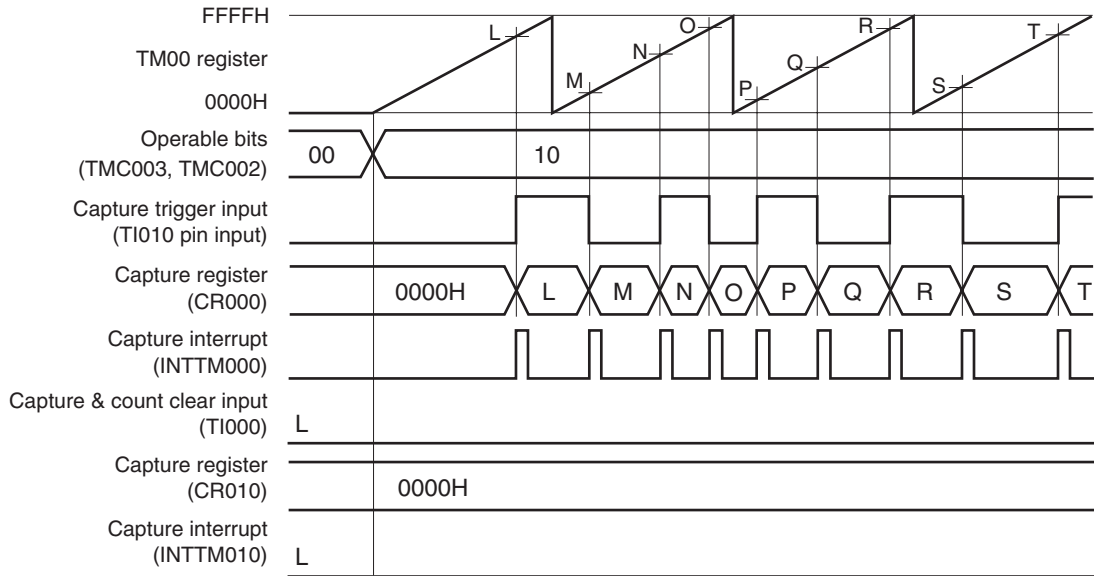


This is an application example where the count value is captured to CR010, TM00 is cleared, and TO00 output is inverted when the rising or falling edge of the TI000 pin is detected.

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 6-30. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input (CR000: Capture Register, CR010: Capture Register) (2/3)**

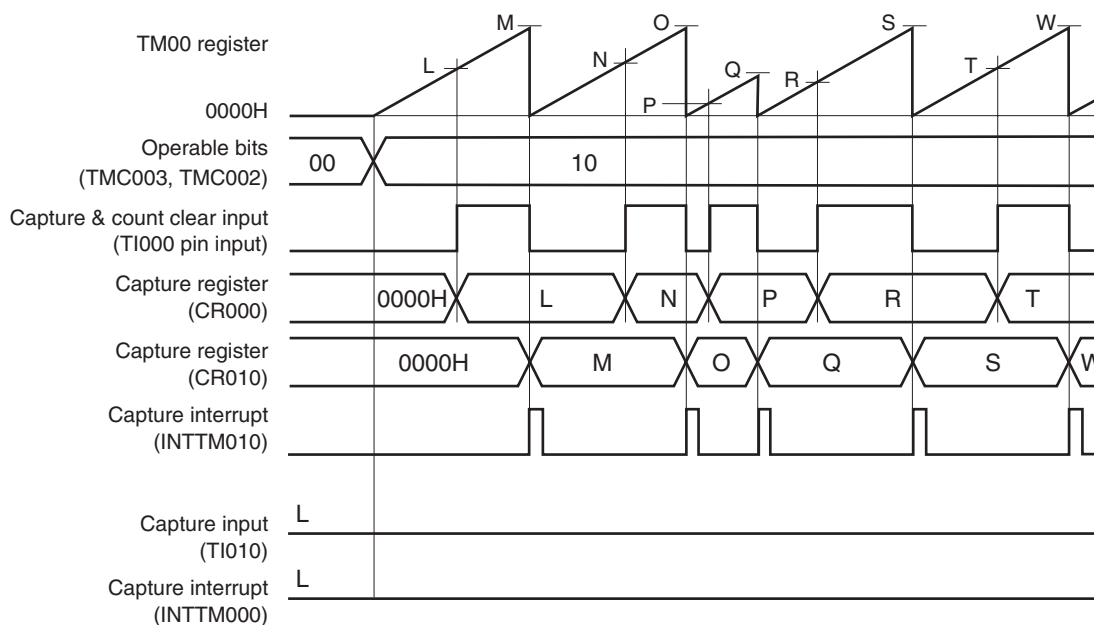
**(b) TOC00 = 13H, PRM00 = C0H, CRC00 = 05H, TMC00 = 0AH**



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 6-30. 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.

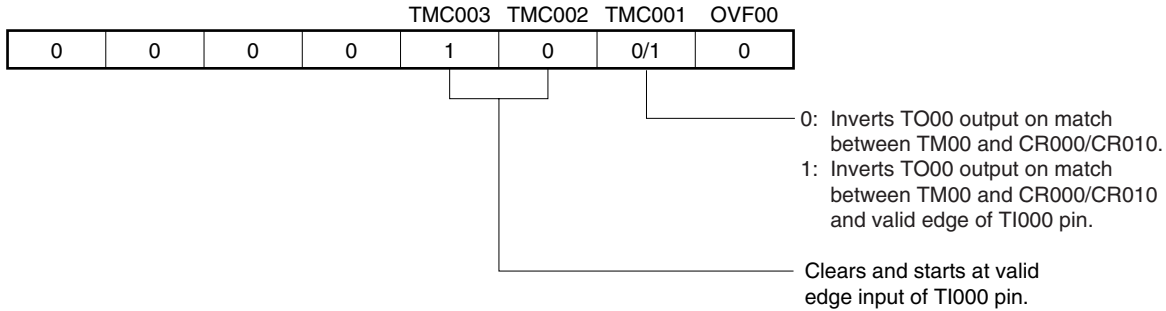
- High-level width = [CR010 value] – [CR000 value] × [Count clock cycle]
- Low-level width = [CR000 value] × [Count clock cycle]

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.

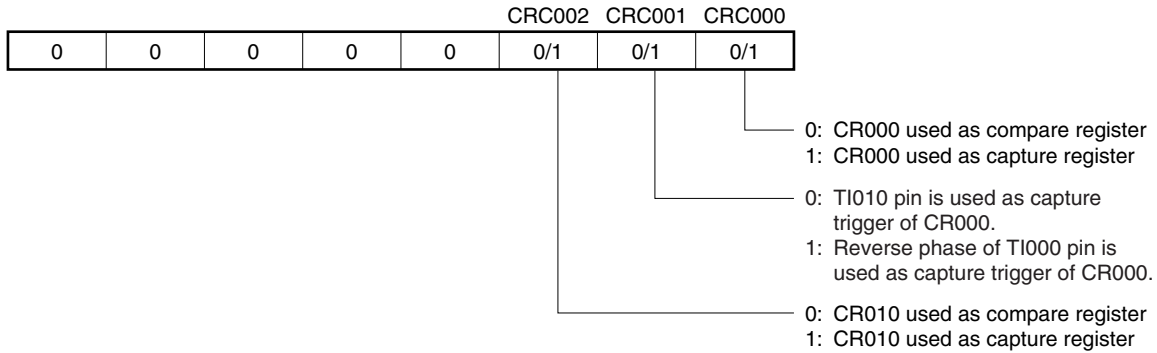
However, if the valid edge specified by bits 6 and 5 (ES101 and ES100) of prescaler mode register 00 (PRM00) is input to the TI010 pin, the count value is not captured but the INTTM000 signal is generated. To measure the pulse width of the TI000 pin, mask the INTTM000 signal when it is not used.

Figure 6-31. Example of Register Settings in Clear & Start Mode Entered by TI000 Pin Valid Edge Input (1/2)

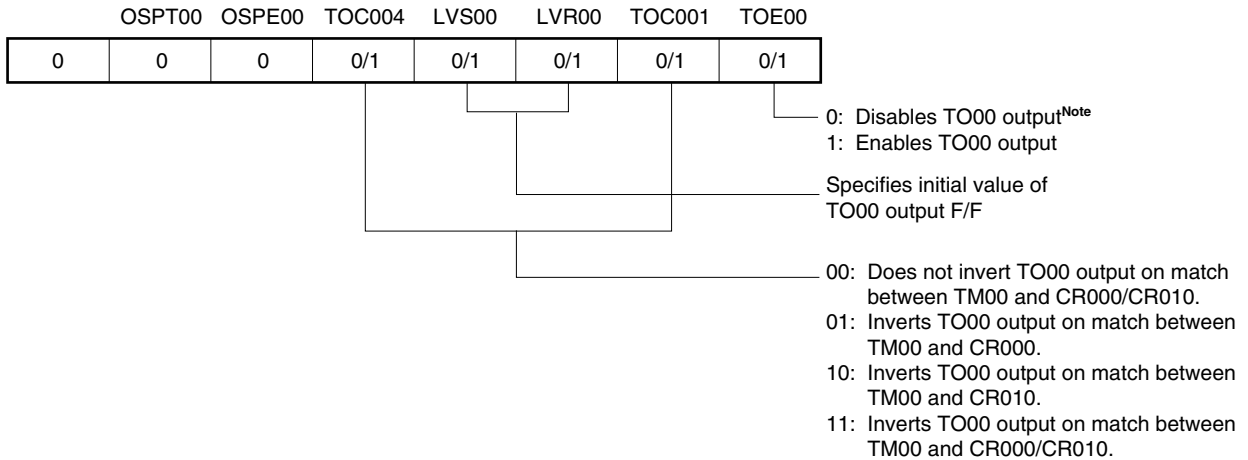
(a) 16-bit timer mode control register 00 (TMC00)



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

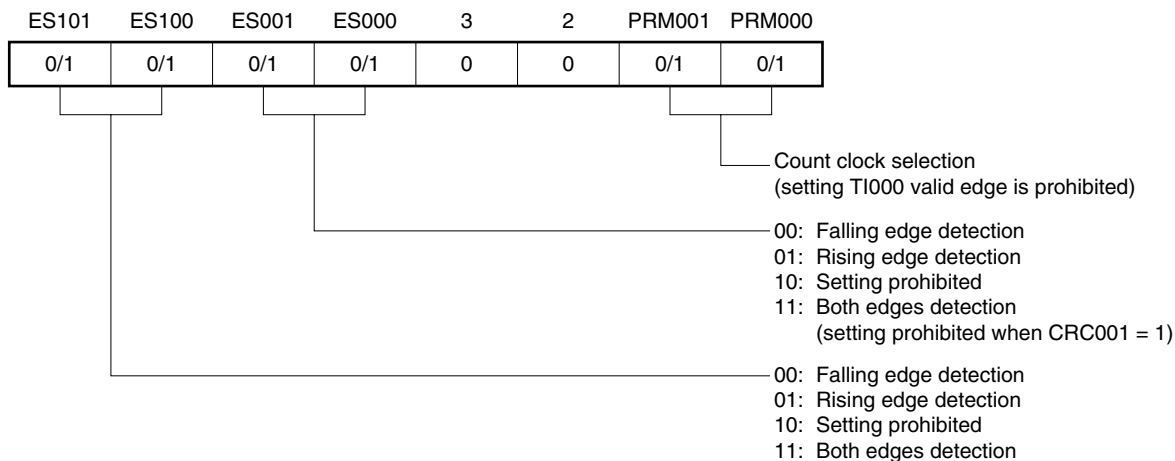


(c) 16-bit timer output control register 00 (TOC00)



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

Figure 6-31. Example of Register Settings in Clear &amp; 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<sup>Note</sup> input as a capture trigger. When the valid edge of the capture trigger is detected, the count value of TM00 is stored in CR000.

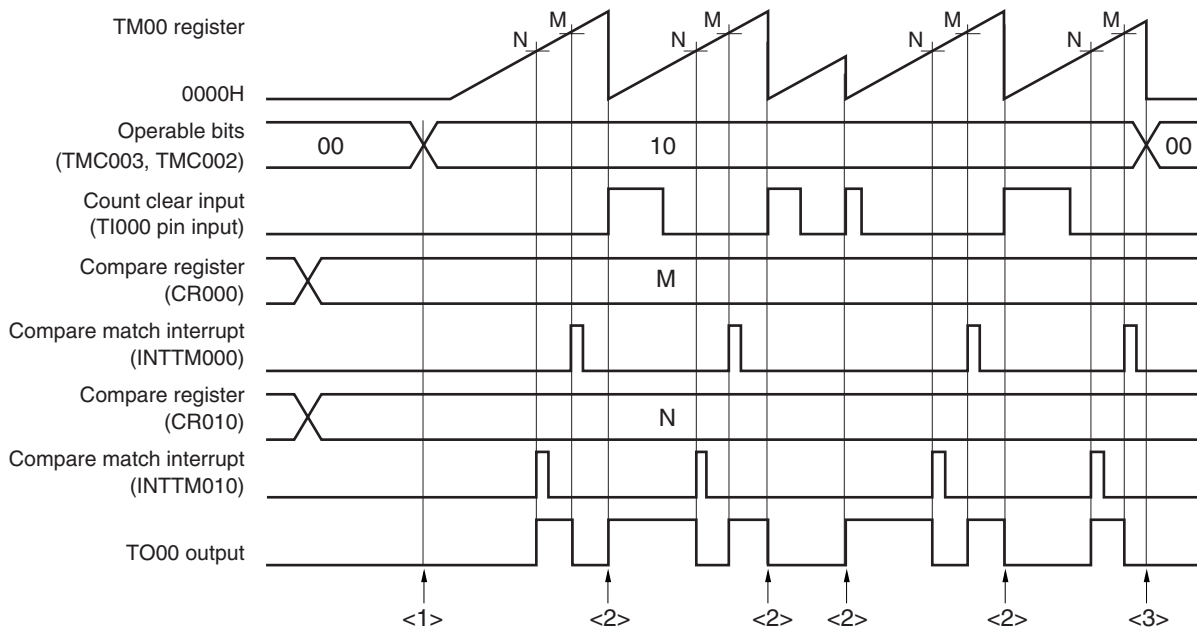
**Note** The timer output (TO00) cannot be used when detection of the valid edge of the TI010 pin is used.

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

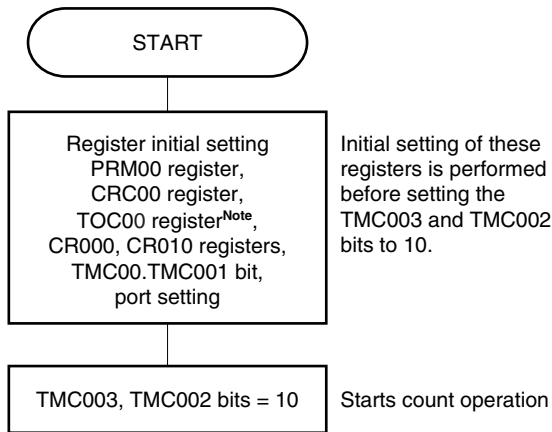
When this register is used as a compare register and when its value matches the count value of TM00, an interrupt signal (INTTM010) is generated. The count value of TM00 is not cleared.

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.

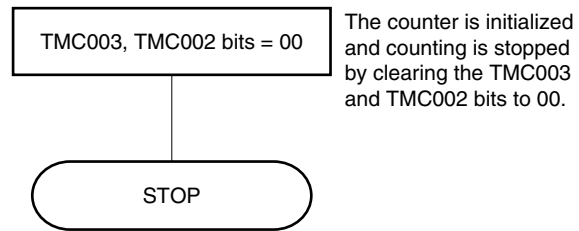
Figure 6-32. Example of Software Processing in Clear & Start Mode Entered by TI000 Pin Valid Edge Input



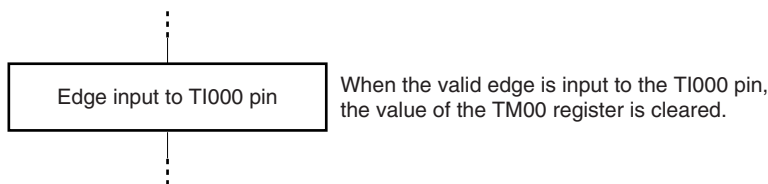
<1> Count operation start flow



<3> Count operation stop flow



<2> TM00 register clear & start flow



**Note** Care must be exercised when setting TOC00. For details, see 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.

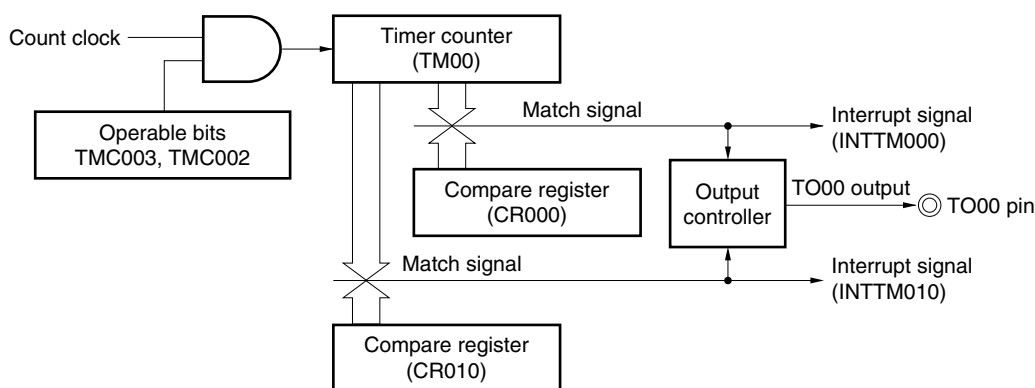
The following three types of free-running timer operations are available.

- 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, see 6.3 (5) **Port mode register 0 (PM0)** and (6) **Port output mode resistors (POM0)**.
- 2.** For how to enable the INTTM000 signal interrupt, see **CHAPTER 11 INTERRUPT FUNCTIONS**.

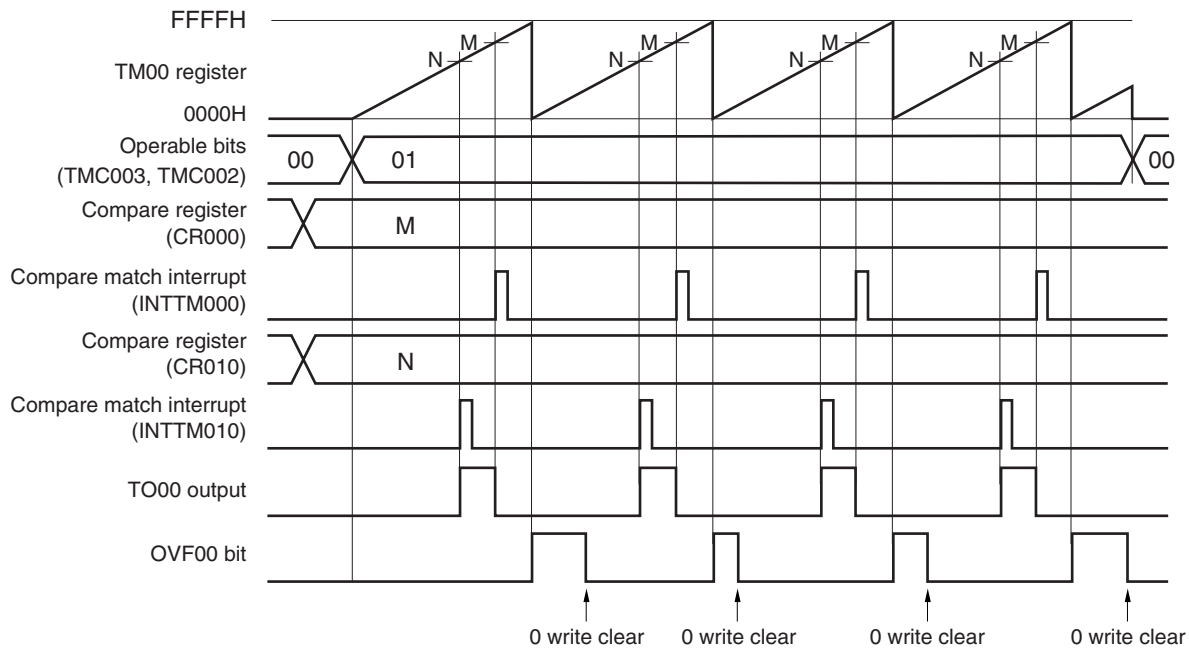
#### (1) Free-running timer mode operation (CR000: compare register, CR010: compare register)

**Figure 6-33. Block Diagram of Free-Running Timer Mode  
(CR000: Compare Register, CR010: Compare Register)**



**Figure 6-34. Timing Example of Free-Running Timer Mode  
(CR000: Compare Register, CR010: Compare Register)**

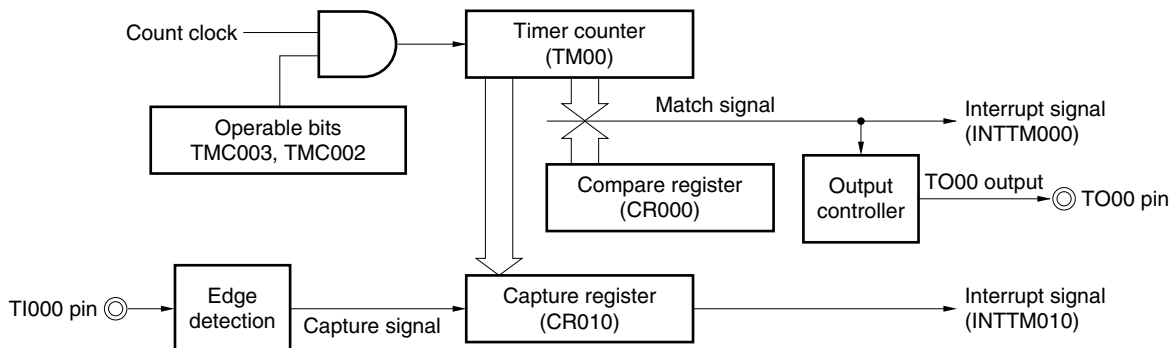
• TOC00 = 13H, PRM00 = 00H, CRC00 = 00H, TMC00 = 04H



This is an application example where two compare registers are used in the free-running timer mode. The TO00 output level is inverted 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.

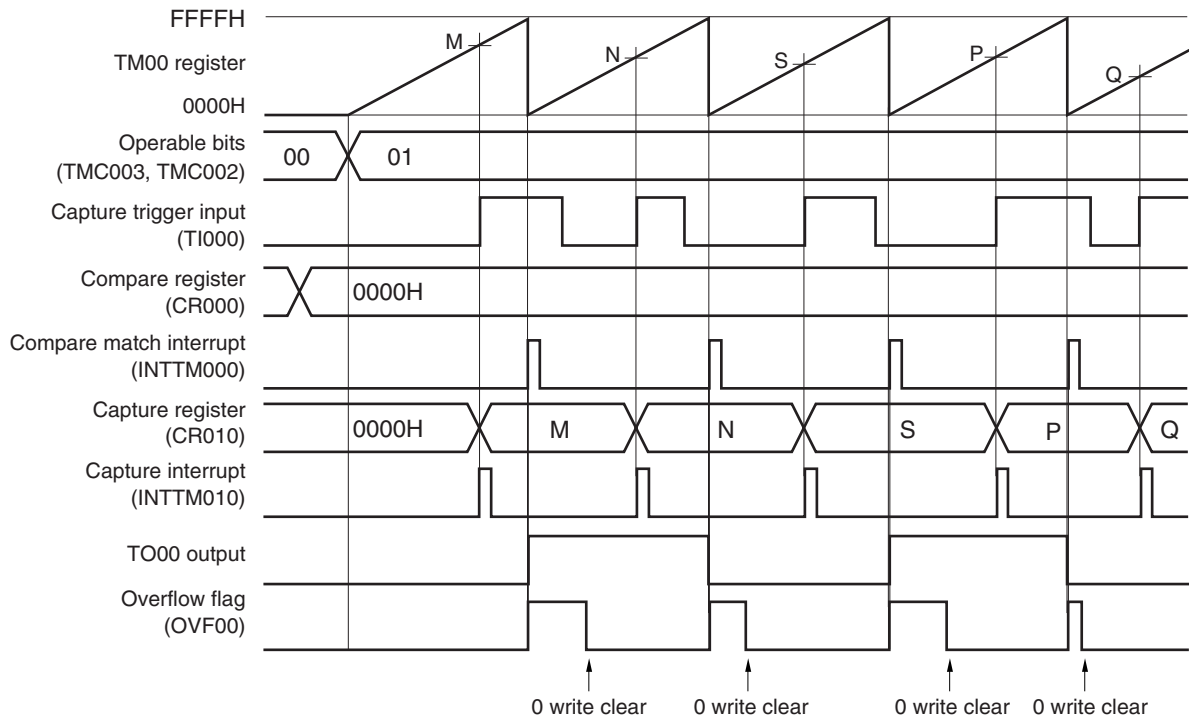
**(2) Free-running timer mode operation  
(CR000: compare register, CR010: capture register)**

**Figure 6-35. Block Diagram of Free-Running Timer Mode  
(CR000: Compare Register, CR010: Capture Register)**



**Figure 6-36. Timing Example of Free-Running Timer Mode  
(CR000: Compare Register, CR010: Capture Register)**

• TOC00 = 13H, PRM00 = 10H, CRC00 = 04H, TMC00 = 04H

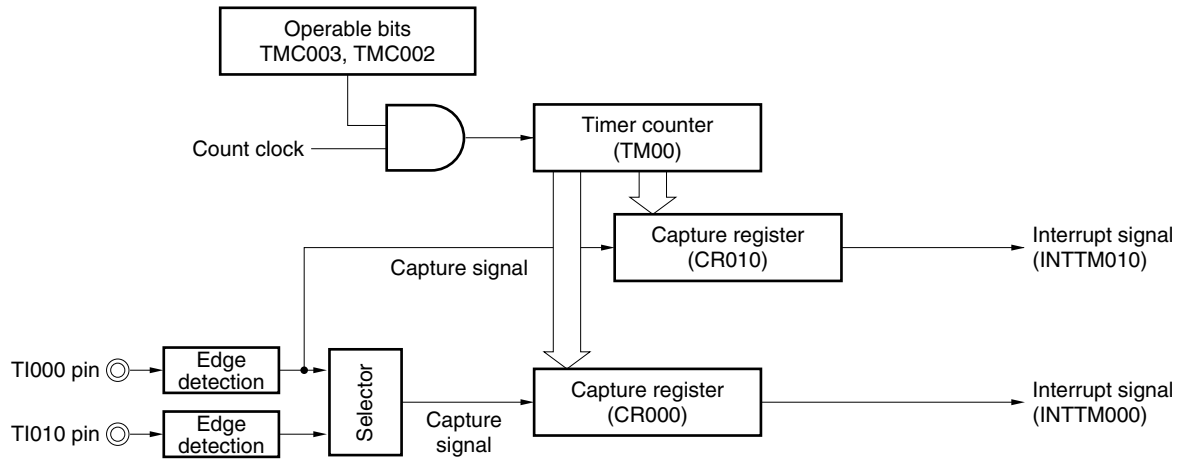


This is an application example where a compare register and a capture register are used at the same time in the free-running timer mode.

In this example, the INTTM000 signal is generated and the TO00 output level is inverted 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)

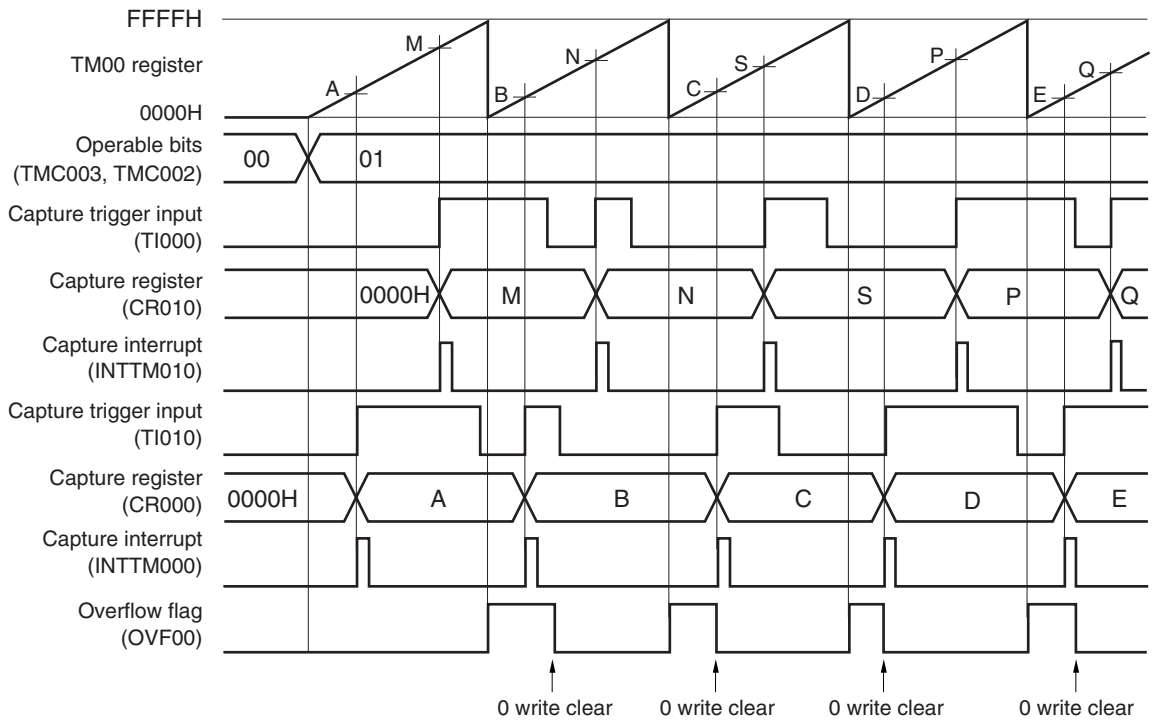
Figure 6-37. Block Diagram of Free-Running Timer Mode  
(CR000: Capture Register, CR010: Capture Register)



**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 6-38. Timing Example of Free-Running Timer Mode  
(CR000: Capture Register, CR010: Capture Register) (1/2)**

**(a) TOC00 = 13H, PRM00 = 50H, CRC00 = 05H, TMC00 = 04H**

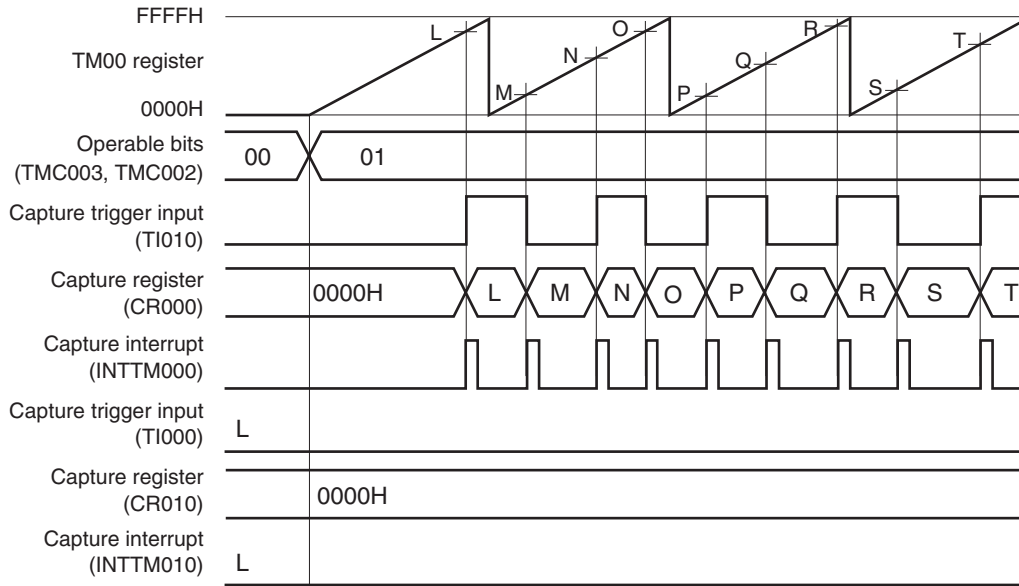


This is an application example where the count values that have been captured at the valid edges of separate capture trigger signals are stored in separate capture registers in the free-running timer mode.

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 6-38. 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 6-39. Example of Register Settings in Free-Running Timer Mode (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: Inverts TO00 output on match between TM00 and CR000/CR010.
- 1: Inverts TO00 output on match between TM00 and CR000/CR010 and valid edge of TI000 pin.
- Free-running timer mode

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

				CRC002	CRC001	CRC000
0	0	0	0	0	0/1	0/1

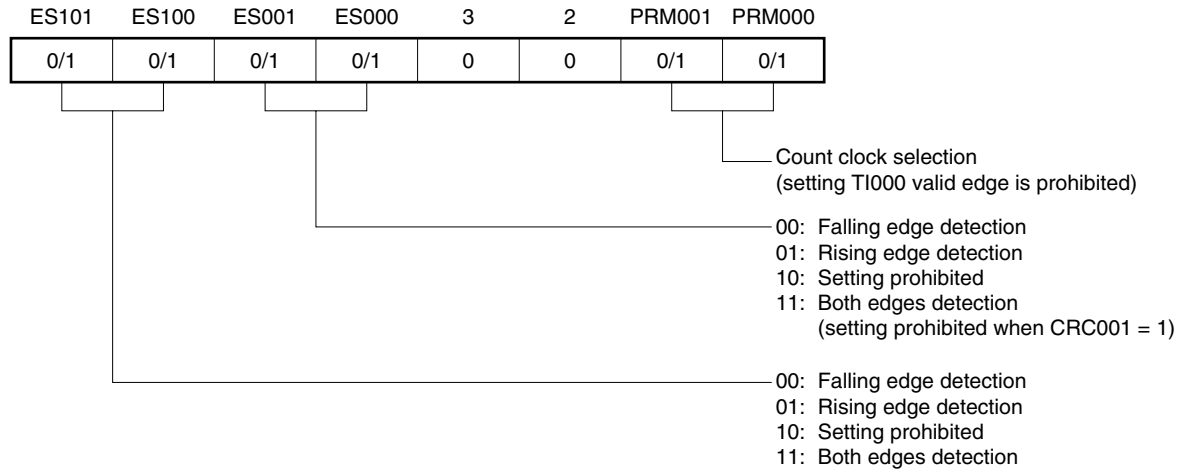
- 0: CR000 used as compare register
- 1: CR000 used as capture register
- 0: TI010 pin is used as capture trigger of CR000.
- 1: Reverse phase of TI000 pin is used as capture trigger of CR000.
- 0: CR010 used as compare register
- 1: CR010 used as capture register

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

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

- 0: Disables TO00 output
- 1: Enables TO00 output
- Specifies initial value of TO00 output F/F
- 00: Does not invert TO00 output on match between TM00 and CR000/CR010.
- 01: Inverts TO00 output on match between TM00 and CR000.
- 10: Inverts TO00 output on match between TM00 and CR010.
- 11: Inverts TO00 output on match between TM00 and CR000/CR010.

Figure 6-39. 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.

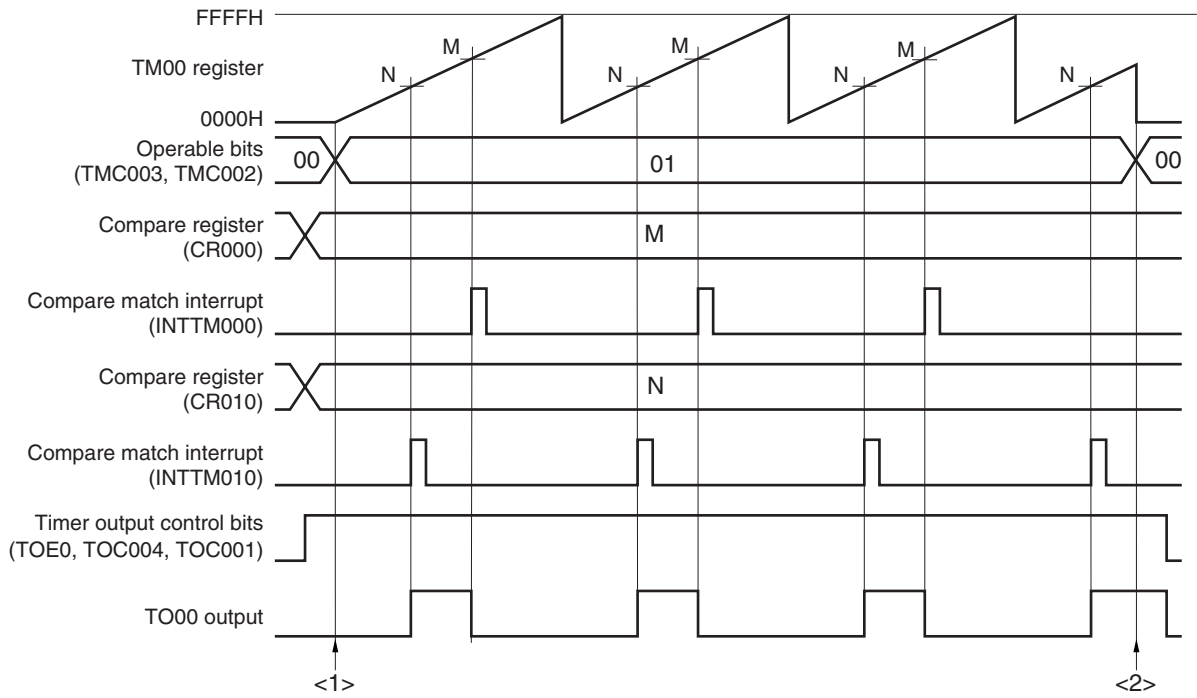
**(g) 16-bit capture/compare register 010 (CR010)**

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.

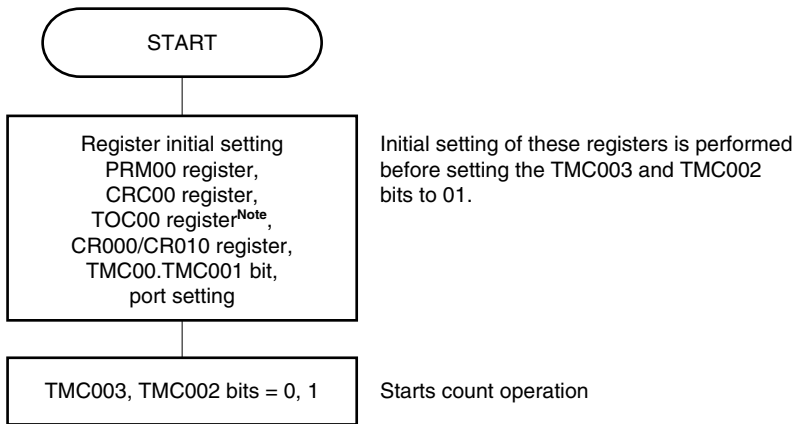
When this register is used as a capture register, the TI000 pin input is used as a capture trigger. When the valid edge of the capture trigger is detected, the count value of TM00 is stored in CR010.



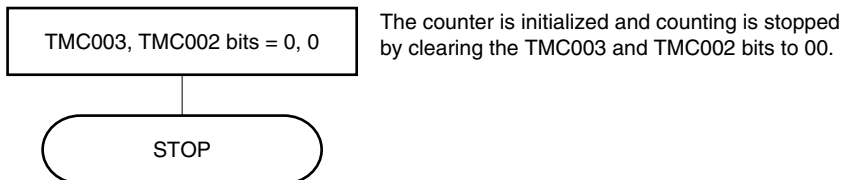
Figure 6-40. Example of Software Processing in Free-Running Timer Mode



<1> Count operation start flow



<2> Count operation stop flow



**Note** Care must be exercised when setting TOC00. For details, see 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, see 6.5.1 Rewriting CR010 during TM00 operation.

- Remarks**
1. For the setting of I/O pins, see 6.3 (5) Port mode register 0 (PM0) and (6) Port output mode resistors (POM0).
  2. For how to enable the INTTM000 signal interrupt, see CHAPTER 11 INTERRUPT FUNCTIONS.

Figure 6-41. Block Diagram of PPG Output Operation

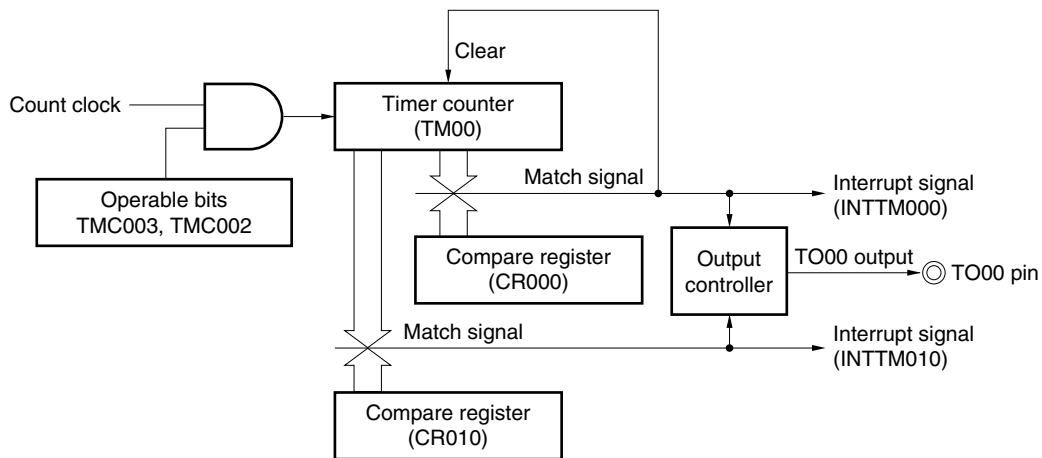
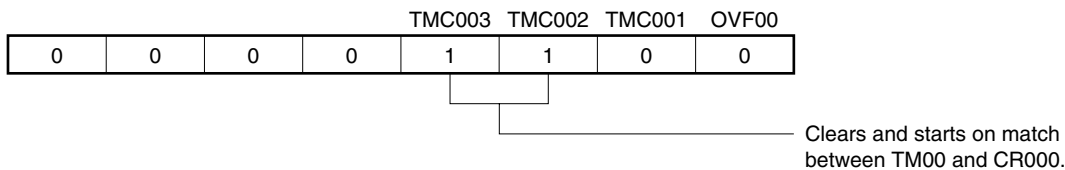
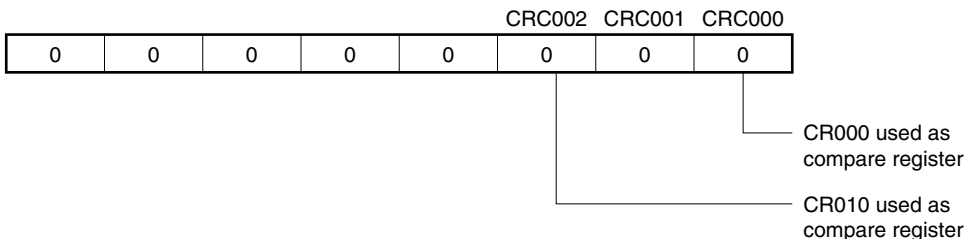


Figure 6-42. Example of Register Settings for PPG Output Operation

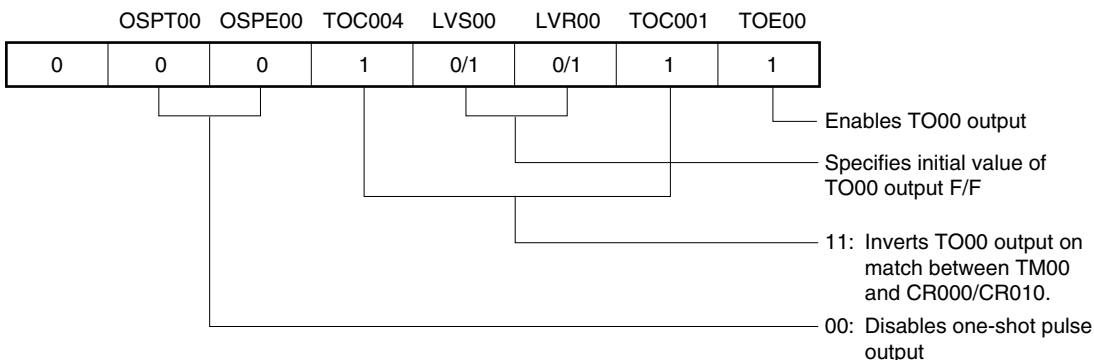
(a) 16-bit timer mode control register 00 (TMC00)



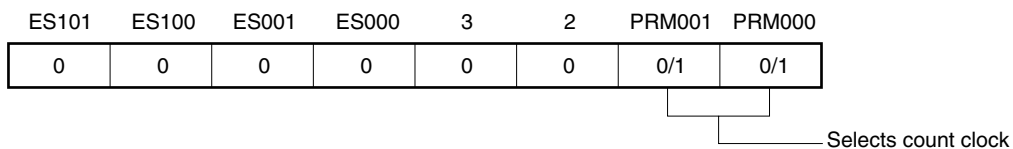
(b) Capture/compare control register 00 (CRC00)



(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)

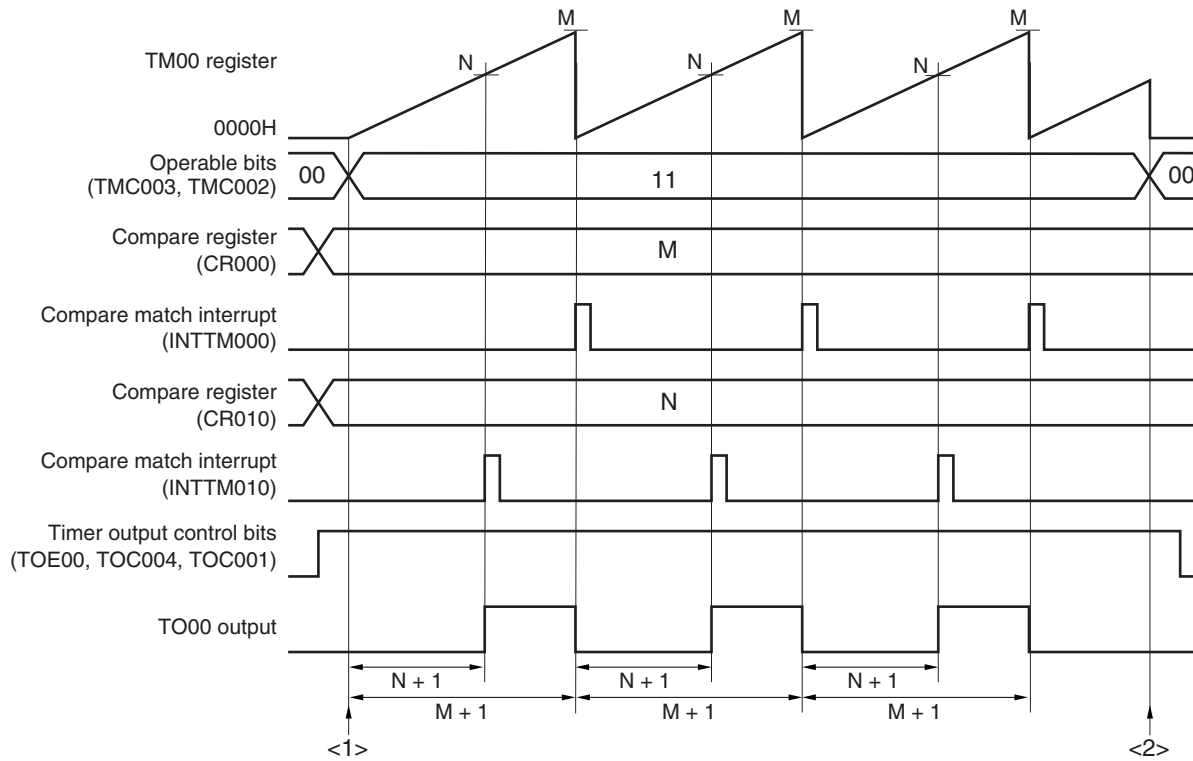
An interrupt signal (INTTM000) is generated when the value of this register matches the count value of TM00. The count value of TM00 is not cleared.

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

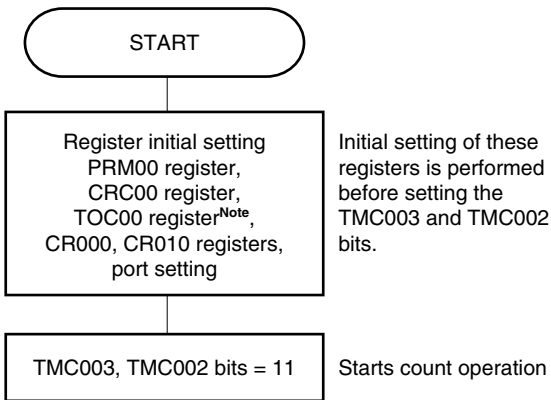
An interrupt signal (INTTM010) is generated when the value of this register matches the count value of TM00. The count value of TM00 is not cleared.

**Caution** Set values to CR000 and CR010 such that the condition  $0000H \leq CR010 < CR000 \leq FFFFH$  is satisfied.

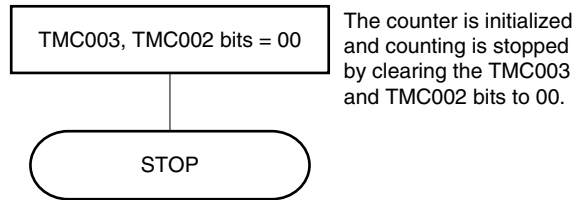
Figure 6-43. Example of Software Processing for PPG Output Operation



<1> Count operation start flow



<2> Count operation stop flow



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

**Remark** PPG pulse cycle =  $(M + 1) \times \text{Count clock cycle}$   
 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, see 6.3 (5) Port mode register 0 (PM0) and (6) Port output mode resistors (POM0).
  2. For how to enable the INTTM000 signal interrupt, see CHAPTER 11 INTERRUPT FUNCTIONS.

Figure 6-44. Block Diagram of One-Shot Pulse Output Operation

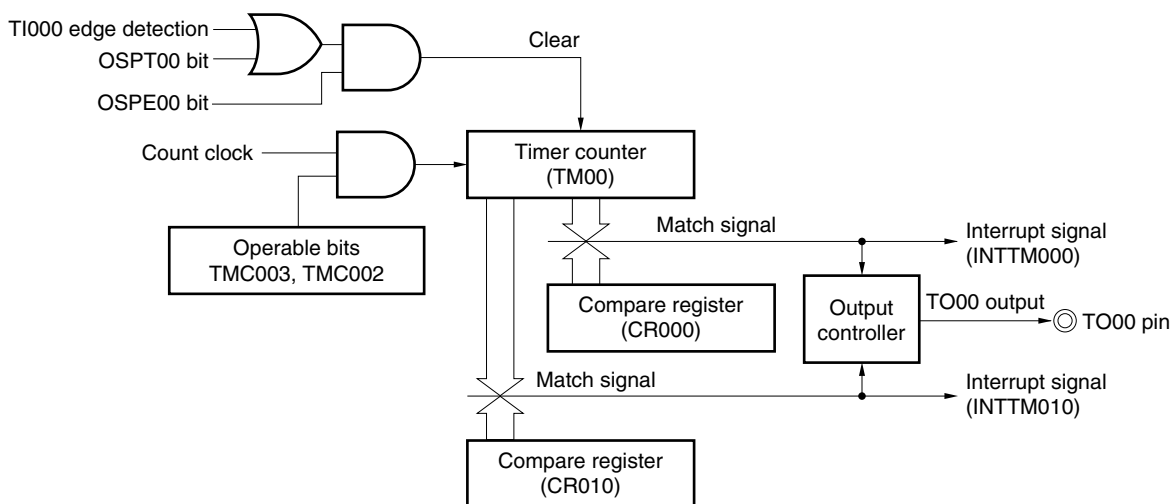
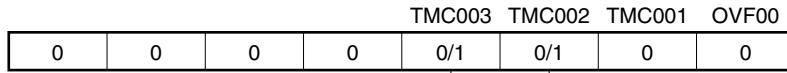


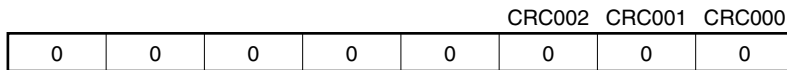
Figure 6-45. Example of Register Settings for One-Shot Pulse Output Operation (1/2)

(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.

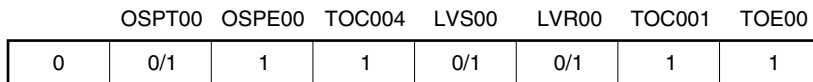
(b) Capture/compare control register 00 (CRC00)



CR000 used as compare register

CR010 used as compare register

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



Enables TO00 output

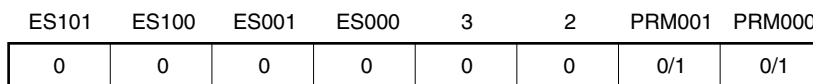
Specifies initial value of TO00 output

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 6-45. 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)**

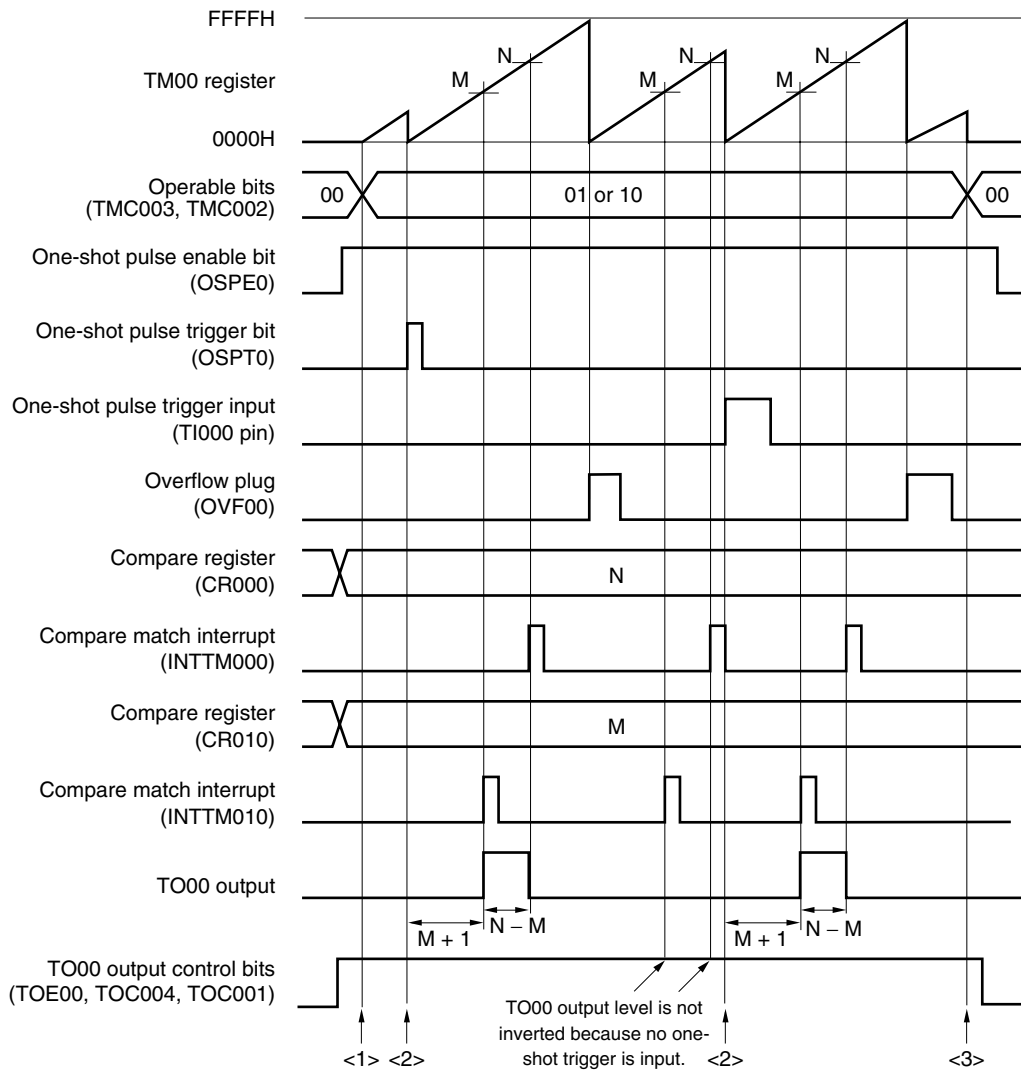
This register is used as a compare register when a one-shot pulse is output. When the value of TM00 matches that of CR000, an interrupt signal (INTTM000) is generated and the TO00 output level is inverted.

**(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 6-46. Example of Software Processing for One-Shot Pulse Output Operation (1/2)

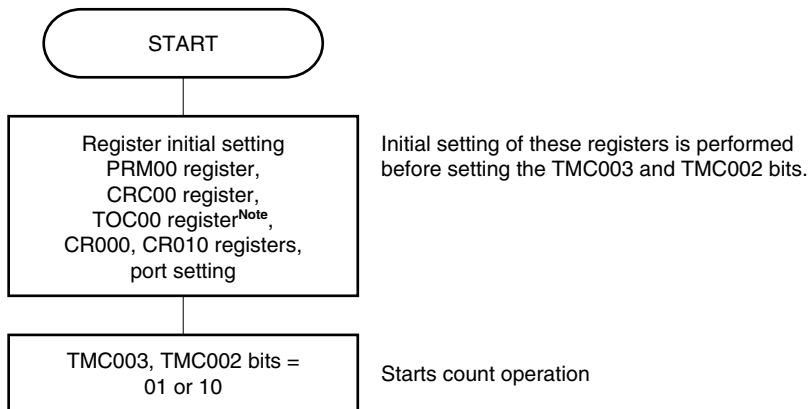


- Time from when the one-shot pulse trigger is input until the one-shot pulse is output  
=  $(M + 1) \times \text{Count clock cycle}$
- One-shot pulse output active level width  
=  $(N - M) \times \text{Count clock cycle}$

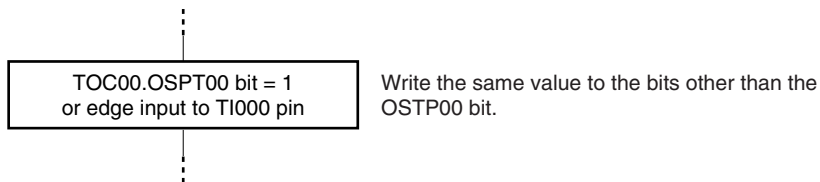


Figure 6-46. Example of Software Processing for One-Shot Pulse Output Operation (2/2)

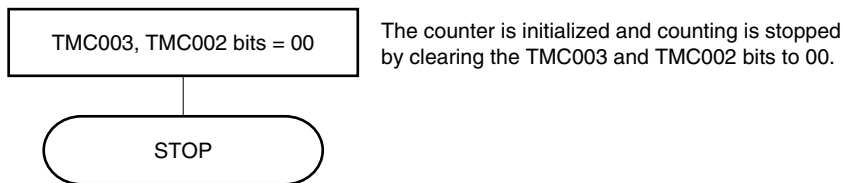
<1> Count operation start flow



<2> One-shot trigger input flow



<3> Count operation stop flow



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

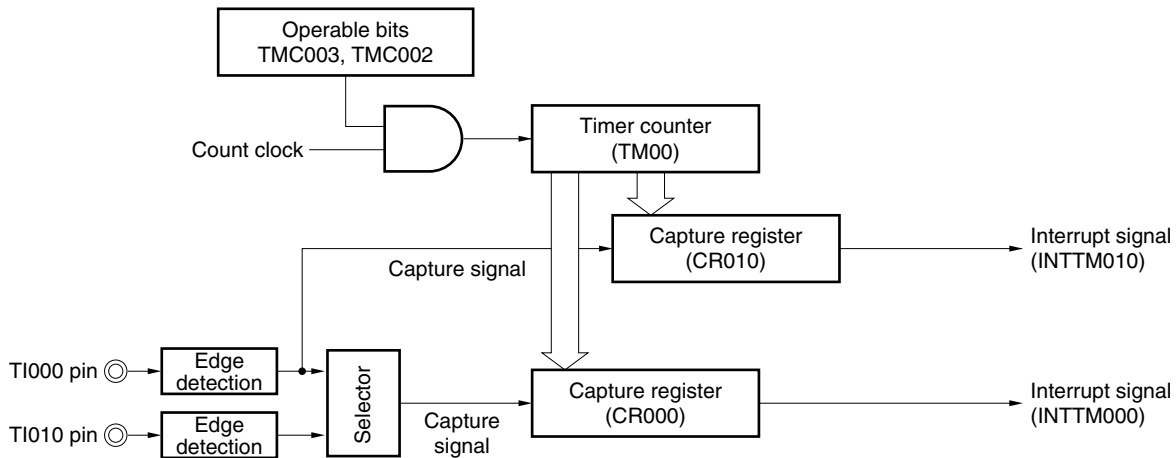
**6.4.8 Pulse width measurement operation**

TM00 can be used to measure the pulse width of the signal input to the TI000 and TI010 pins.

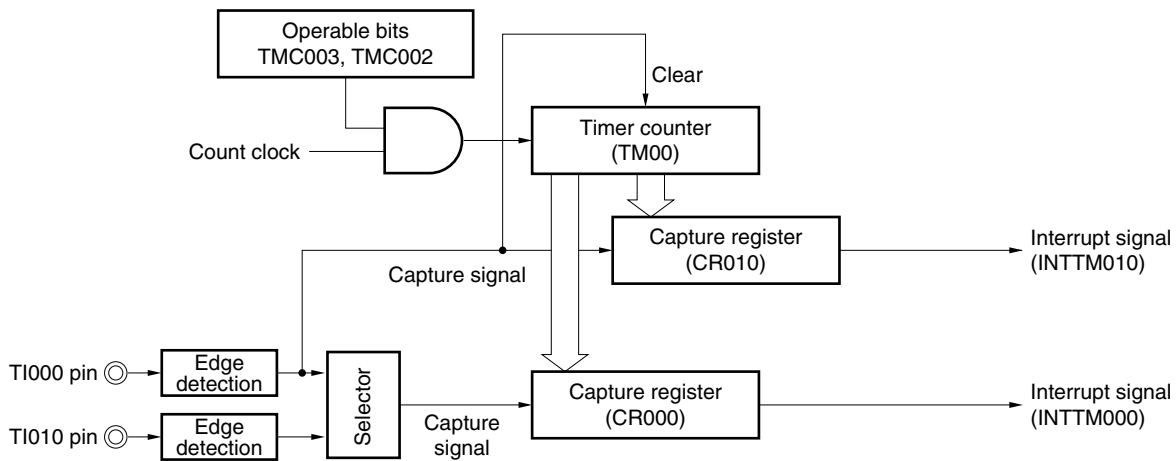
Measurement can be accomplished by operating the 16-bit timer/event counter 00 in the free-running timer mode or by restarting the timer in synchronization with the signal input to the TI000 pin.

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 6-47. Block Diagram of Pulse Width Measurement (Free-Running Timer Mode)**



**Figure 6-48. 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, see **6.3 (5) Port mode register 0 (PM0)**.

**2.** For how to enable the INTTM000 signal interrupt, see **CHAPTER 11 INTERRUPT FUNCTIONS**.

**(1) Measuring the pulse width by using two input signals of the TI000 and TI010 pins (free-running timer mode)**

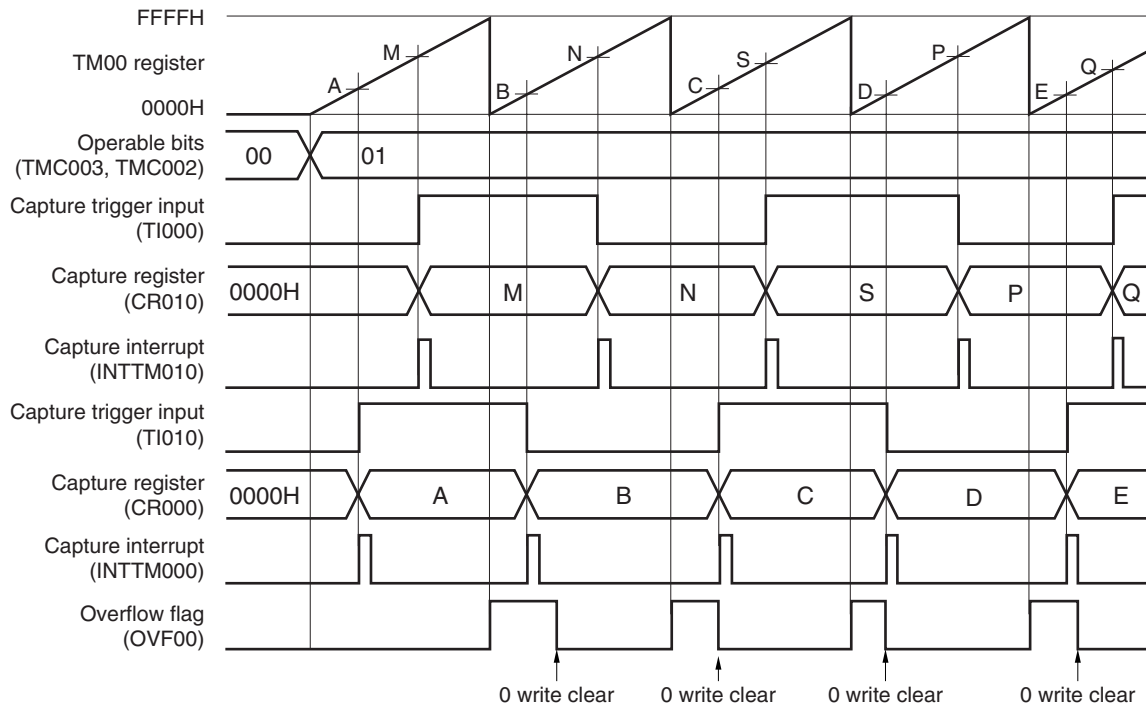
Set the free-running timer mode (TMC003 and TMC002 = 01). When the valid edge of the TI000 pin is detected, the count value of TM00 is captured to CR010. When the valid edge of the TI010 pin is detected, the count value of TM00 is captured to CR000. Specify detection of both the edges of the TI000 and 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 6-49. 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 mode)**

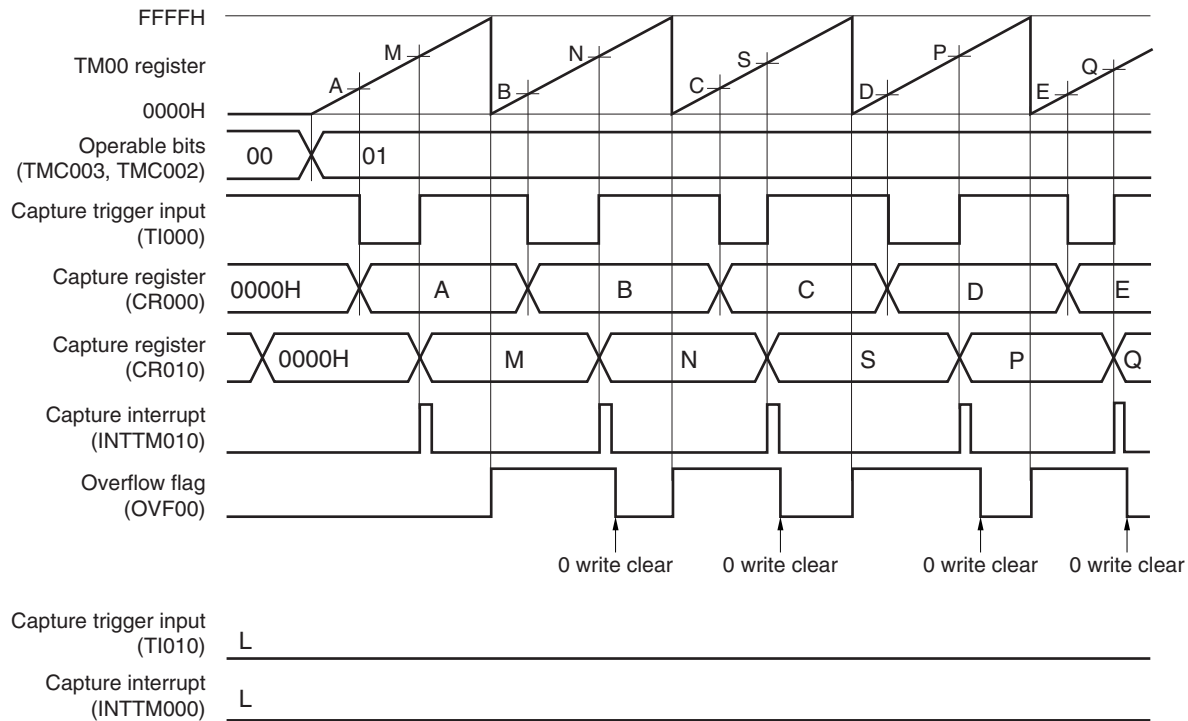
Set the free-running timer mode (TMC003 and TMC002 = 01). The count value of TM00 is captured to CR000 in the phase reverse to the valid edge detected on the TI000 pin. When the valid edge of the TI000 pin is detected, the count value of TM00 is captured to CR010.

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 6-50. Timing Example of Pulse Width Measurement (2)**

• TMC00 = 04H, PRM00 = 10H, CRC00 = 07H



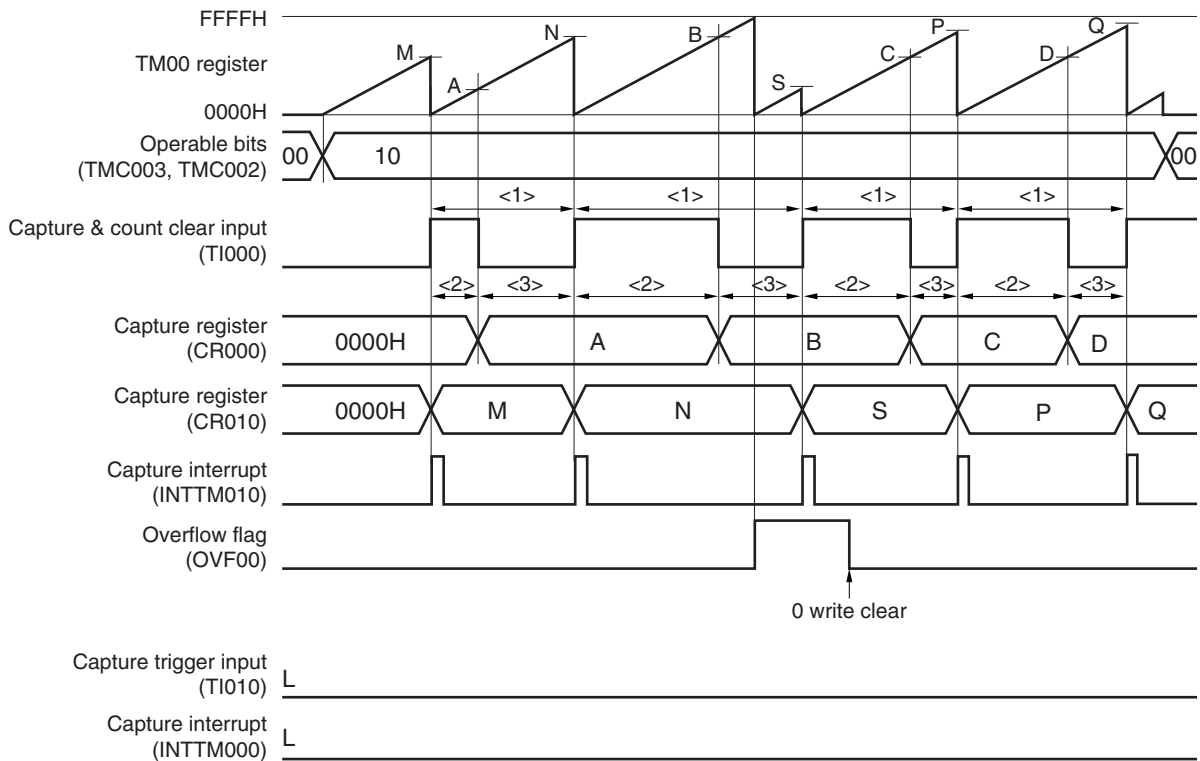
**(3) Measuring the pulse width by using one input signal of the TI000 pin (clear & start mode entered by the TI000 pin valid edge input)**

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 6-51. Timing Example of Pulse Width Measurement (3)**

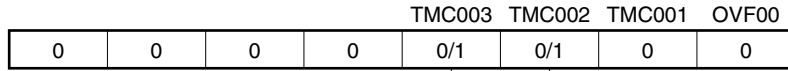
• TMC00 = 08H, PRM00 = 10H, CRC00 = 07H



- <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 =  $(\text{Pulse cycle} - \text{High-level pulse width})$

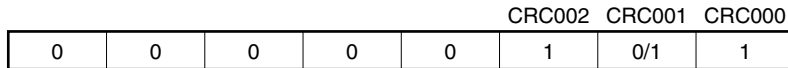
Figure 6-52. Example of Register Settings for Pulse Width Measurement (1/2)

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



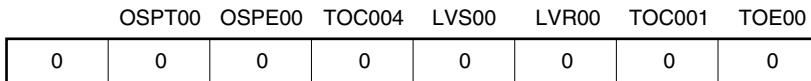
01: Free running timer mode  
 10: Clear and start mode entered by valid edge of TI000 pin.

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

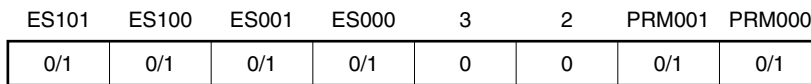


1: CR000 used as capture register  
 0: TI010 pin is used as capture trigger of CR000.  
 1: Reverse phase of TI000 pin is used as capture trigger of CR000.  
 1: CR010 used as capture register

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



(d) Prescaler mode register 00 (PRM00)



Selects count clock (setting valid edge of TI000 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 6-52. 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)**

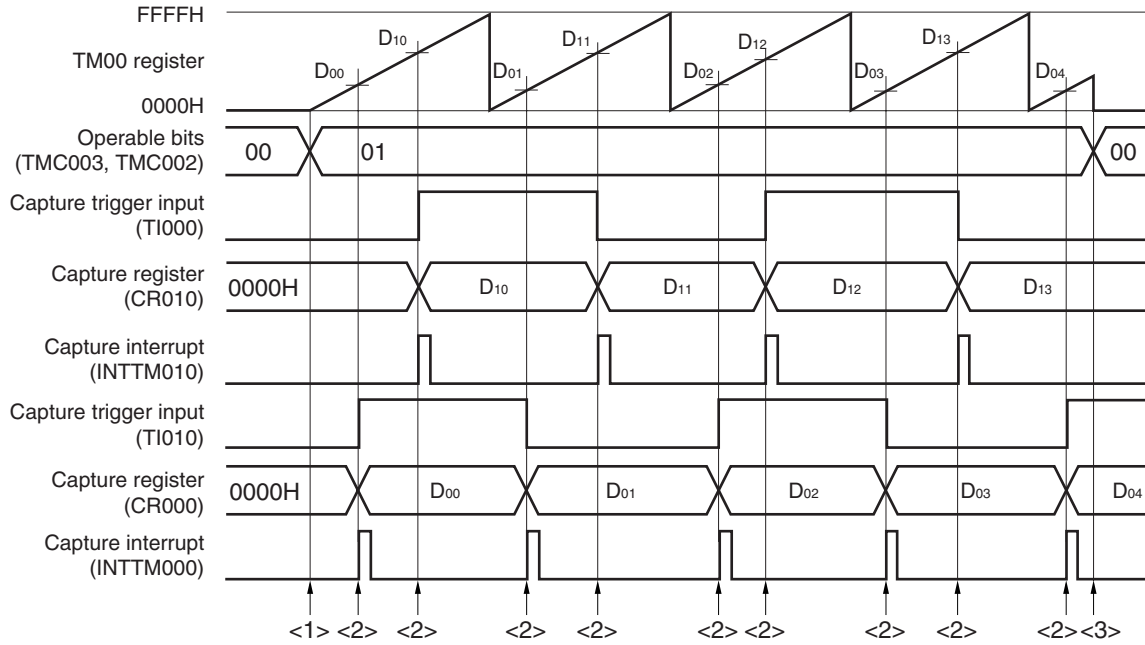
This register is used as a capture register. Either the TI000 or TI010 pin is selected as a capture trigger. When a specified edge of the capture trigger is detected, the count value of TM00 is stored in CR000.

**(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 6-53. 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

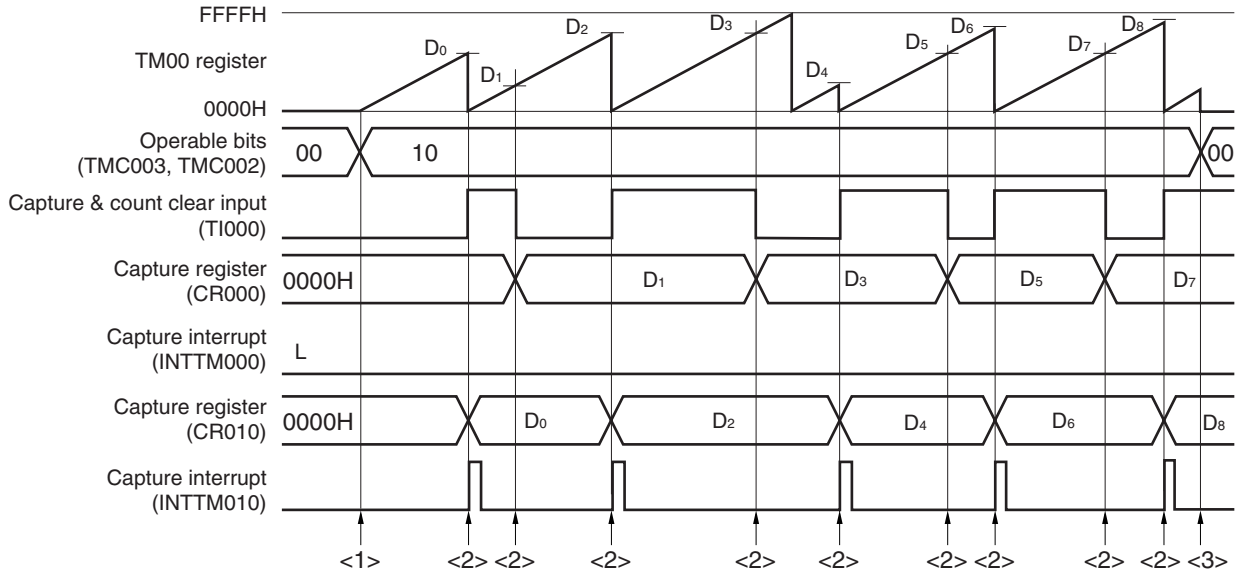
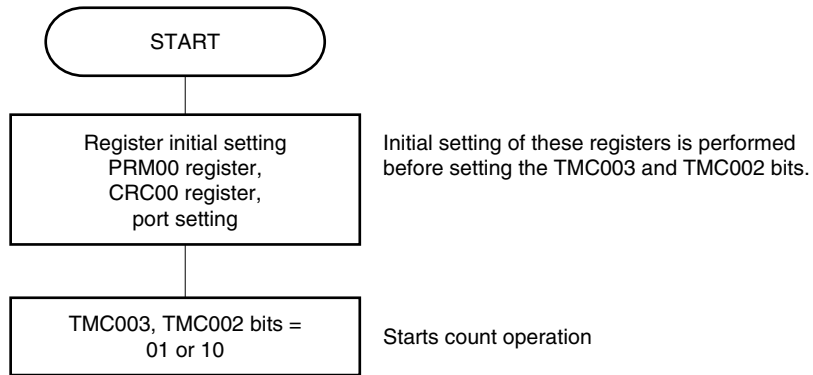


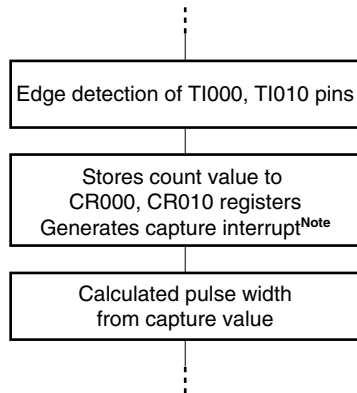


Figure 6-53. Example of Software Processing for Pulse Width Measurement (2/2)

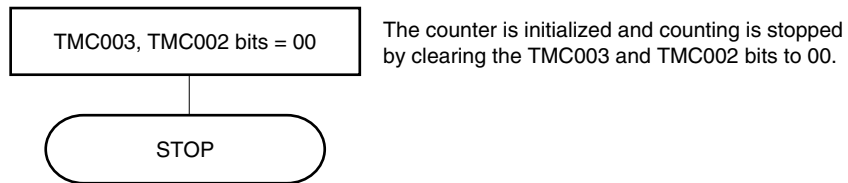
<1> Count operation start flow



<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  $\mu$ PD179F11x, 179F12x microcontrollers 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, see **CHAPTER 11 INTERRUPT FUNCTIONS**.

### 6.5.2 Setting LVS00 and LVR00

#### (1) Usage of LVS00 and LVR00

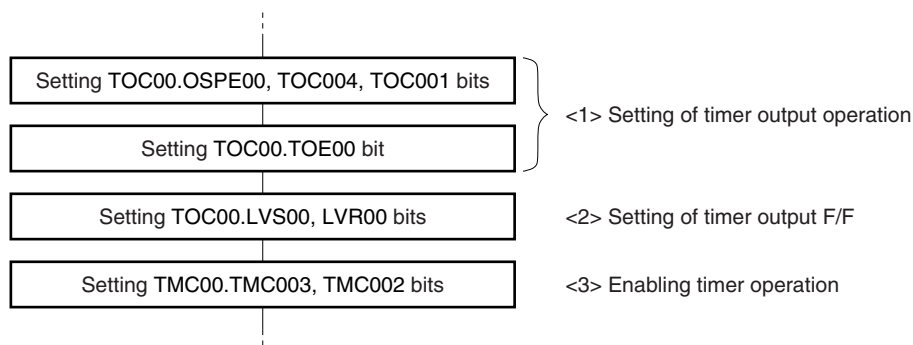
LVS00 and LVR00 are used to set the default value of TO00 output and to invert the timer output without enabling the timer operation (TMC003 and TMC002 = 00). Clear LVS00 and LVR00 to 00 (default value: low-level output) when software control is unnecessary.

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**

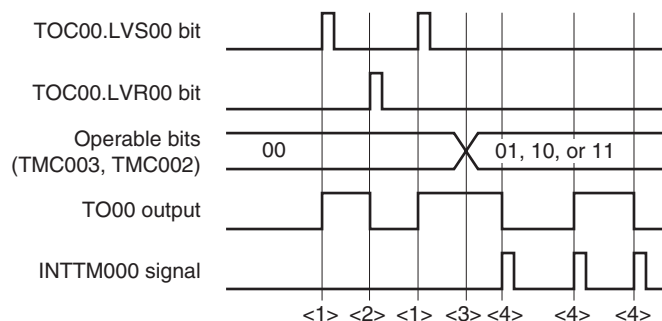
Set LVS00 and LVR00 using the following procedure.

**Figure 6-54. 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 6-55. Timing Example of LVR00 and LVS00**



- <1> TO00 output goes high when LVS00 and LVR00 = 10.
- <2> 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, 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.

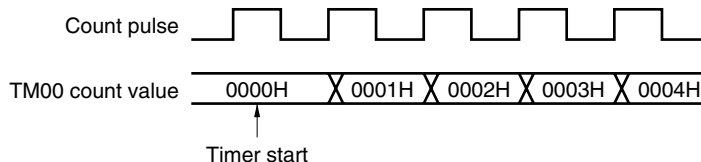
**Table 6-3. Restrictions for Each Channel of 16-bit Timer/Event Counter 00**

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	$0000H \leq CR010 < CR000 \leq FFFFH$
As one-shot pulse output	Setting the same value to CR000 and CR010 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 6-56. Start Timing of TM00 Count**



### (3) Setting of CR000 and CR010 (clear & start mode entered upon a match between TM00 and CR000)

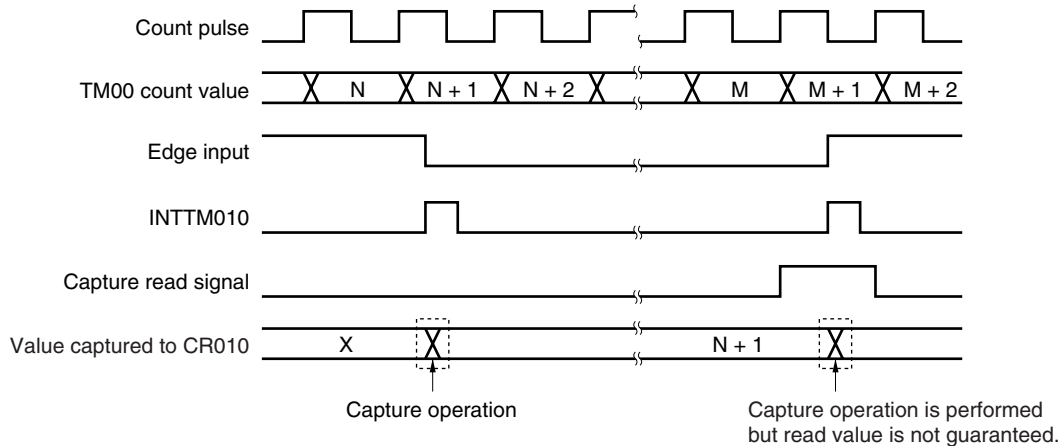
Set a value other than 0000H to CR000 and CR010 (TM00 cannot count one pulse when it is used as an external event counter).

**(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 6-57. 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 ES001.

**(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.

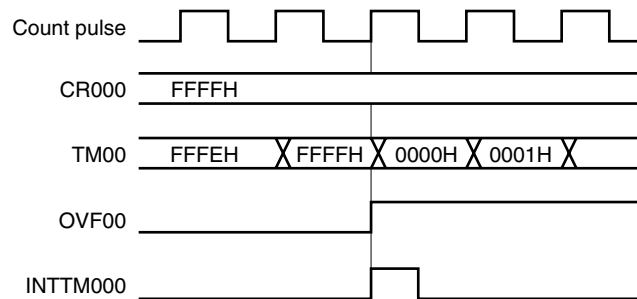
↓

Set CR000 to FFFFH.

↓

When TM00 matches CR000 and TM00 is cleared from FFFFH to 0000H

**Figure 6-58. 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 (see **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 (see **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 (see **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

## 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 0 (PM0) Port register 0 (P0) Port output mode register 0 (POM0)

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

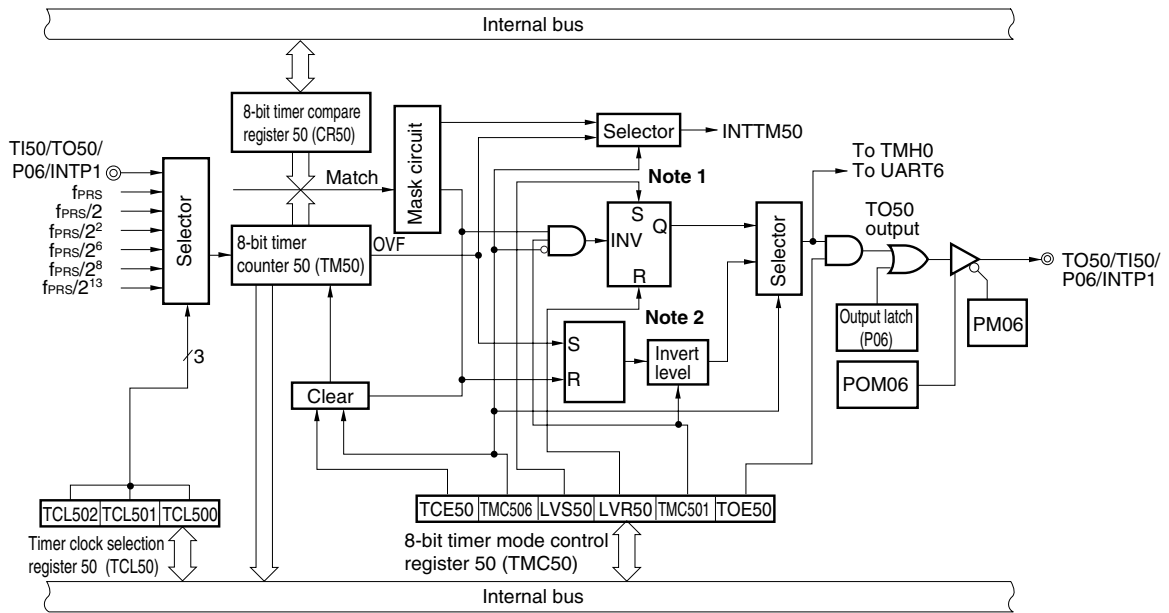
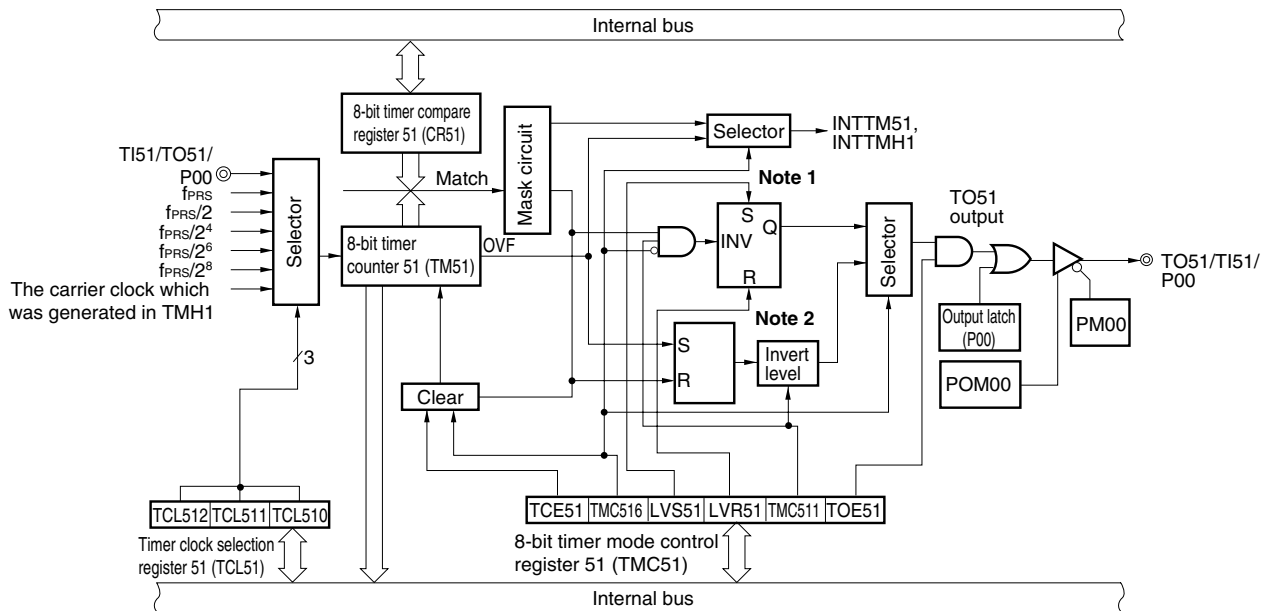


Figure 7-2. Block Diagram of 8-bit Timer/Event Counter 51



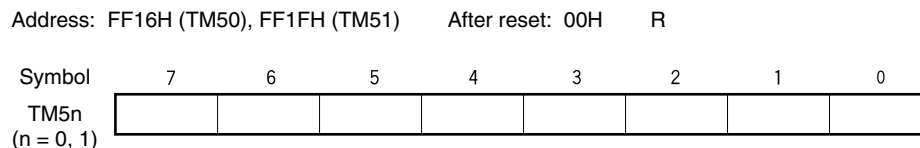
- Notes**
1. Timer output F/F
  2. PWM output F/F

**(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.

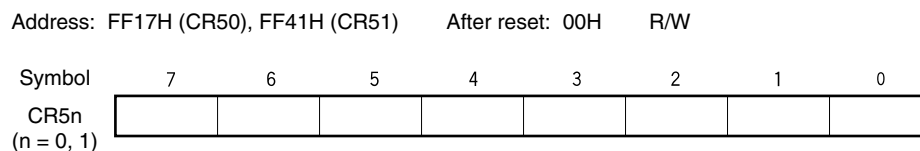
Except in PWM mode, the value set in CR5n is constantly compared with the 8-bit timer counter 5n (TM5n) count value, and an interrupt request (INTTM5n) is generated if they match.

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 0 (PM0)
- Port register 0 (P0)
- Port output mode register 0 (POM0)

#### (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 <sub>PRS</sub> = 2 MHz	f <sub>PRS</sub> = 4 MHz
0	0	0	TI50 pin falling edge		
0	0	1	TI50 pin rising edge		
0	1	0	f <sub>PRS</sub>	2 MHz	4 MHz
0	1	1	f <sub>PRS</sub> /2	1 MHz	2 MHz
1	0	0	f <sub>PRS</sub> /2 <sup>2</sup>	500 kHz	1 MHz
1	0	1	f <sub>PRS</sub> /2 <sup>6</sup>	31.25 kHz	62.25 kHz
1	1	0	f <sub>PRS</sub> /2 <sup>8</sup>	7.81 kHz	15.6 kHz
1	1	1	f <sub>PRS</sub> /2 <sup>13</sup>	0.24 kHz	0.488 kHz

- 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<sub>PRS</sub>: 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 <sub>PRS</sub> = 2 MHz	f <sub>PRS</sub> = 4 MHz
0	0	0	TI51 pin falling edge		
0	0	1	TI51 pin rising edge		
0	1	0	f <sub>PRS</sub>	2 MHz	4MHz
0	1	1	f <sub>PRS</sub> /2	1 MHz	2MHz
1	0	0	f <sub>PRS</sub> /2 <sup>4</sup>	125 kHz	250 kHz
1	0	1	f <sub>PRS</sub> /2 <sup>6</sup>	31.25 kHz	62.5 kHz
1	1	0	f <sub>PRS</sub> /2 <sup>8</sup>	7.81 kHz	15.6 kHz
1	1	1	The carrier clock which was generated in TMH1	–	–

- 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<sub>PRS</sub>: 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<sup>Note</sup>

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<sup>Note</sup>

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 level)							
1	0	Timer output F/F set (1) (default value of TO51 output: high level)							
1	1	Setting prohibited							
TMC511	In other modes (TMC516 = 0)				In PWM mode (TMC516 = 1)				
	Timer F/F control				Active level selection				
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/P06/INTP1 and TO51/TI51/P00 pin outputs are determined depending on PM06 and P06, and PM00 and P00, 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 in TO5n output regardless of the value of TCE5n.
  - n = 0, 1

**(3) Port mode registers 0 (PM0)**

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

When using the P06/TO50/TI50/INTP1 and P00/TO51/TI51 pins for timer output, clear PM06 and PM00 and the output latches of P06 and P00 to 0.

When using the P06/TO50/TI50/INTP1 and P00/TO51/TI51 pins for timer input, set PM06 and PM00 to 1. The output latches of P06 and P00 at this time may be 0 or 1.

PM0 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 0 (PM0)**

Address: FF20H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00

PM0n	P0n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

**(4) Port output mode resistors (POM0)**

This register set the output mode of port 0.

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

Reset signal generation sets these registers to 00H.

**Figure 7-10. Format of Pull-up Resistor Option Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POM0	POM07	POM06	POM05	POM04	POM03	POM02	POM01	POM00	FF38H	00H	R/W

POM0n	P0n pin output mode selection (n = 0 to 7)
0	CMOS output
1	N-ch open-drain output (P07:P-ch open-drain output)

## 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.

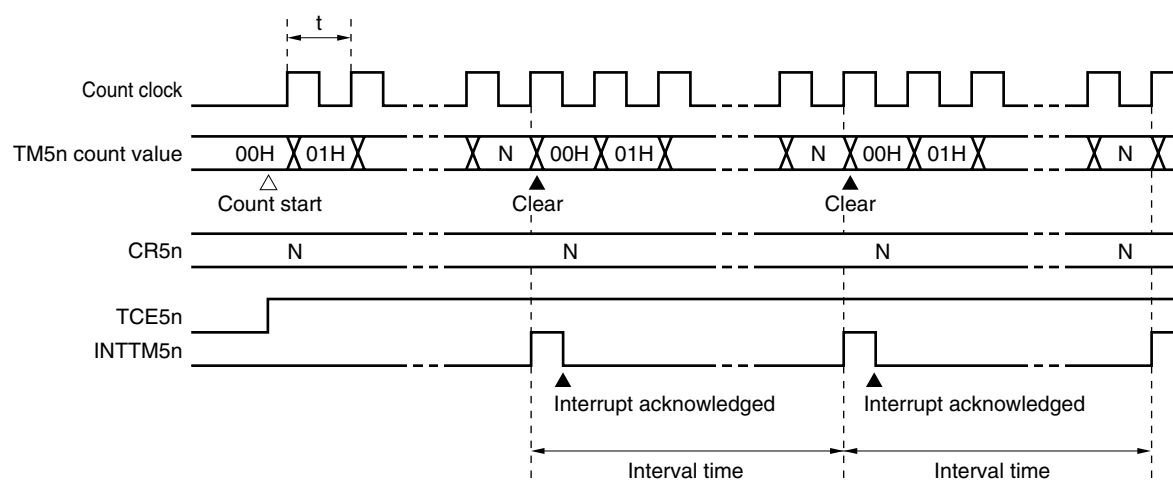
**Caution** Do not write other values to CR5n during operation.

**Remarks** 1. For how to enable the INTTM5n signal interrupt, see **CHAPTER 11 INTERRUPT FUNCTIONS**.

2. n = 0, 1

**Figure 7-11. Interval Timer Operation Timing (1/2)**

#### (a) Basic operation



**Remark** Interval time =  $(N + 1) \times t$

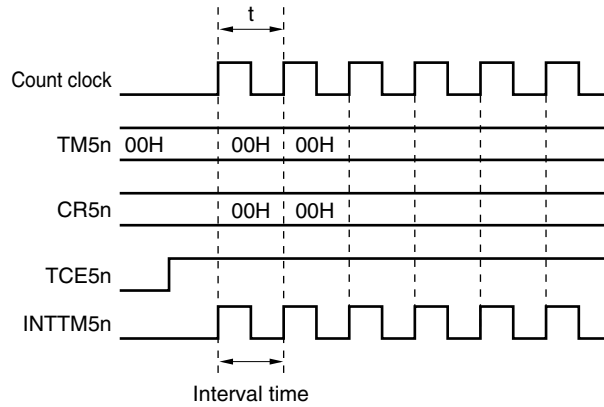
N = 01H to FFH

n = 0, 1

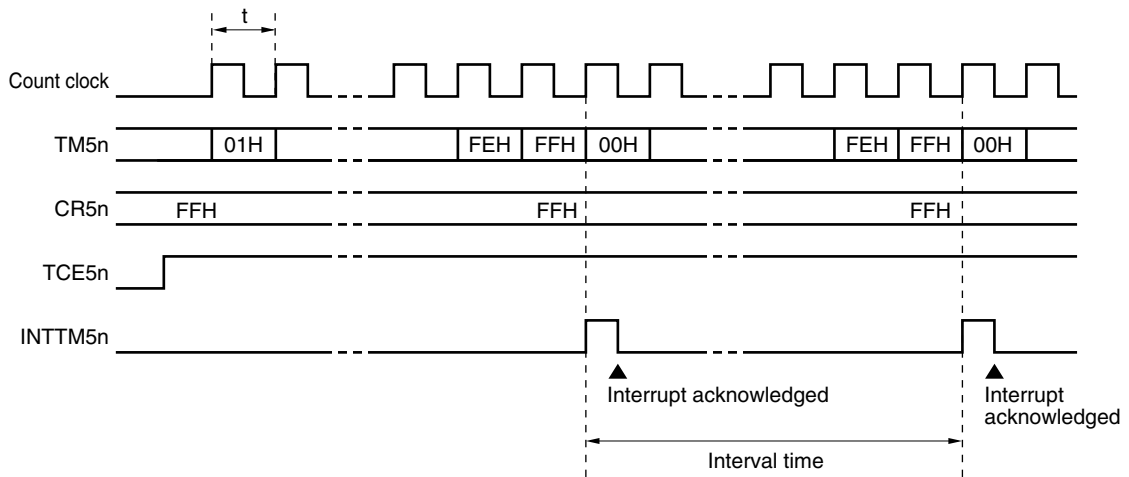


Figure 7-11. Interval Timer Operation Timing (2/2)

(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 (PM06 or PM00)<sup>Note</sup> 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).

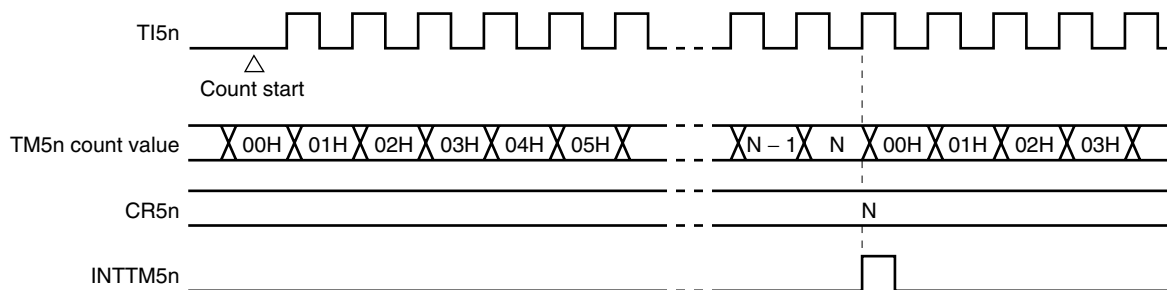
<4> After these settings, INTTM5n is generated each time the values of TM5n and CR5n match.

**Note** 8-bit timer/event counter 50: PM06

8-bit timer/event counter 51: PM00

**Remark** For how to enable the INTTM5n signal interrupt, see **CHAPTER 11 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 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 (P06 or P00)<sup>Note</sup> and port mode register (PM06 or PM00)<sup>Note</sup> 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: P06, PM06

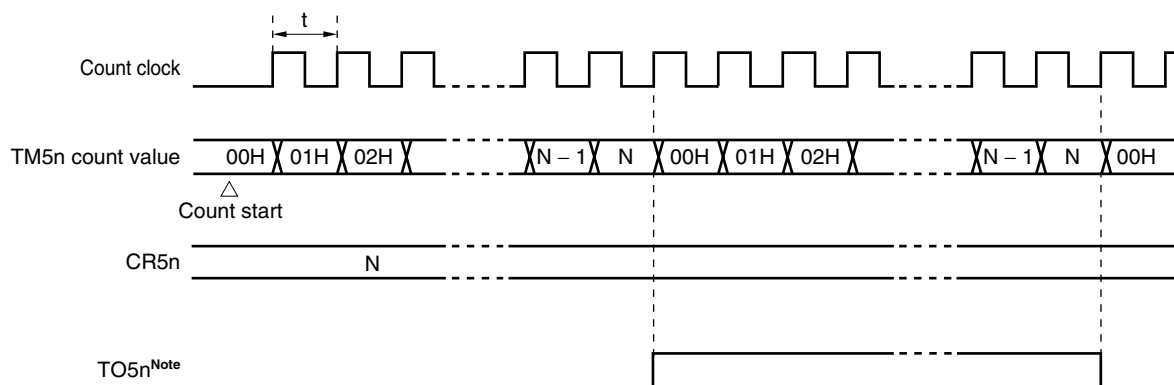
8-bit timer/event counter 51: P00, PM00

**Caution** Do not write other values to CR5n during operation.

**Remarks** 1. For how to enable the INTTM5n signal interrupt, see **CHAPTER 11 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 (P06 or P00)<sup>Note</sup> and port mode register (PM06 or PM00)<sup>Note</sup> 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 = 01000001B or 01000011B)

<2> The count operation starts when TCE5n = 1.  
Clear TCE5n to 0 to stop the count operation.

**Note** 8-bit timer/event counter 50: P06, PM06  
8-bit timer/event counter 51: P00, PM00

**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**.

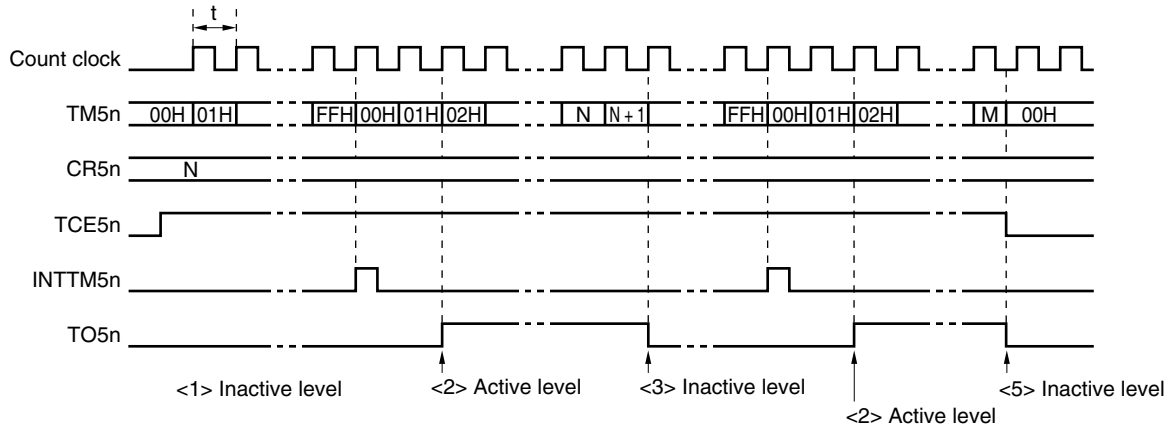
The cycle, active-level width, and duty are as follows.

- Cycle =  $2^8 t$
- Active-level width =  $Nt$
- Duty =  $N/2^8$   
(N = 00H to FFH)

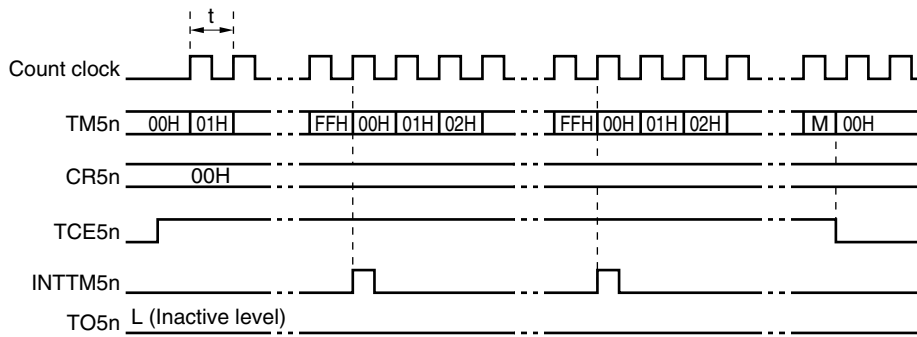
**Remark** n = 0, 1

Figure 7-14. PWM Output Operation Timing

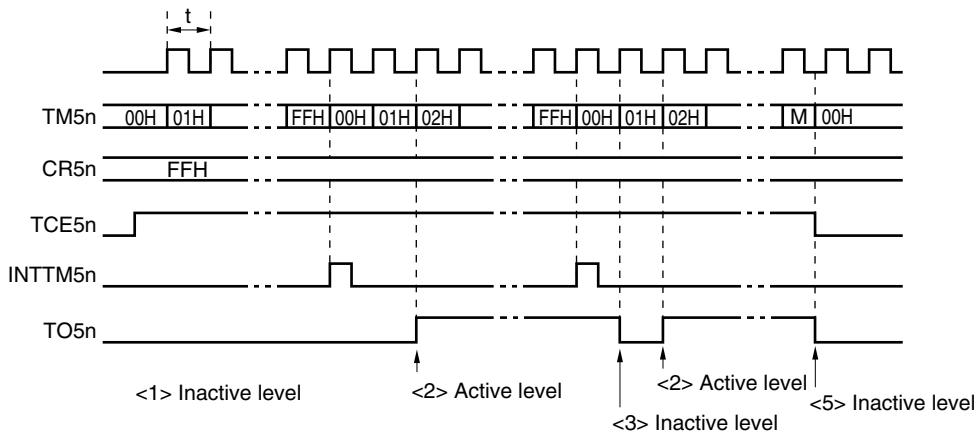
(a) Basic operation (active level = H)



(b) CR5n = 00H



(c) CR5n = FFH



Remarks 1. <1> to <3> and <5> in Figure 7-14 (a) correspond to <1> to <3> and <5> in PWM output operation in

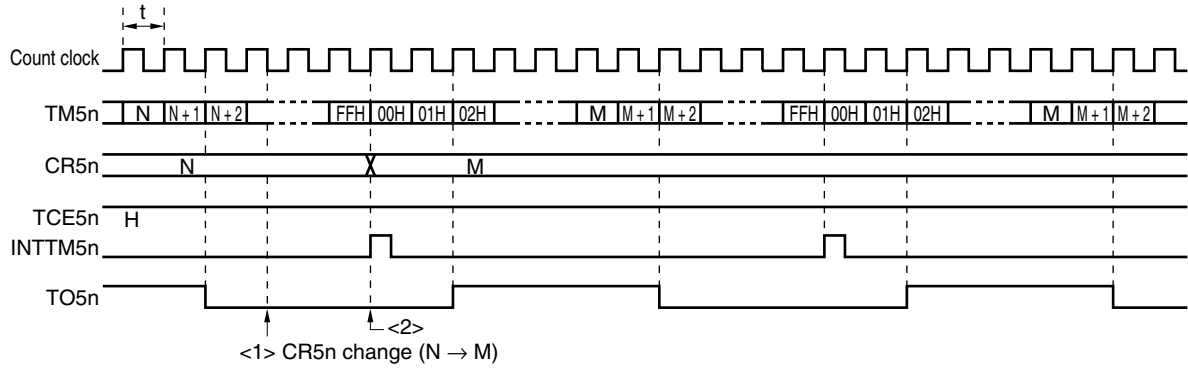
7.4.4 (1) PWM output basic operation.

2. n = 0, 1

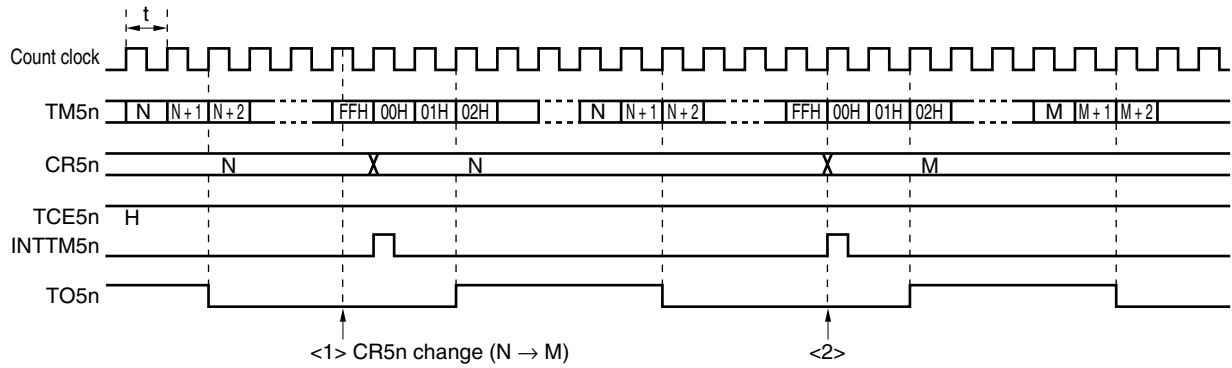
## (2) Operation with CR5n changed

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.



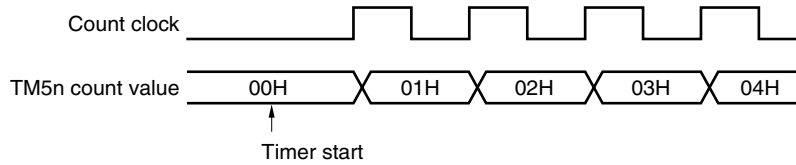
**Caution** When reading from CR5n between <1> and <2> in Figure 7-15, the value read differs from the actual value (read value: M, actual value of CR5n: N).

## 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 Start 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 output for remote control (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	TOH0, TOH1/REM, output controller
Control registers	8-bit timer H mode register n (TMHMDn) 8-bit timer H carrier control register 1 (TMCYC1) <sup>Note</sup> Port mode register 0 (PM0) Port register 0 (P0) Port output mode register 0 (PM0)

**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

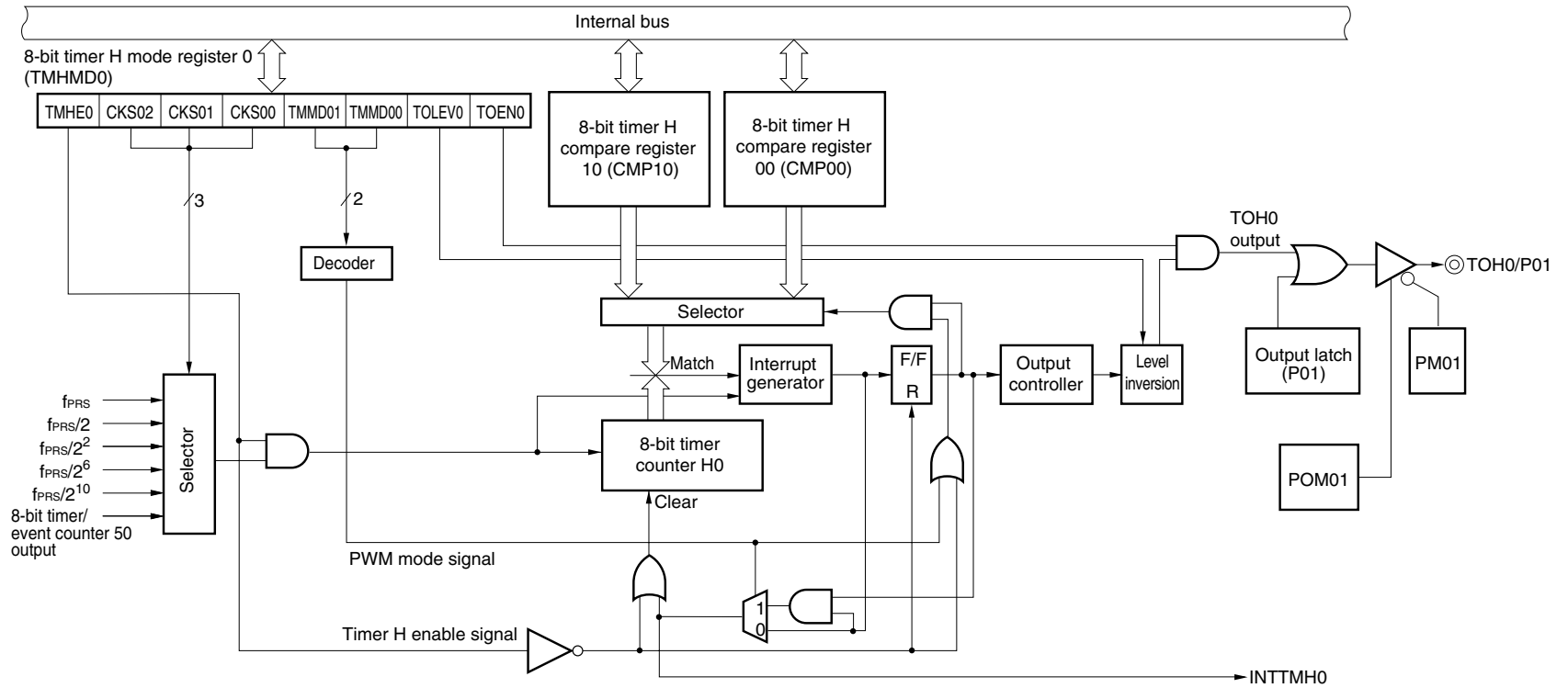
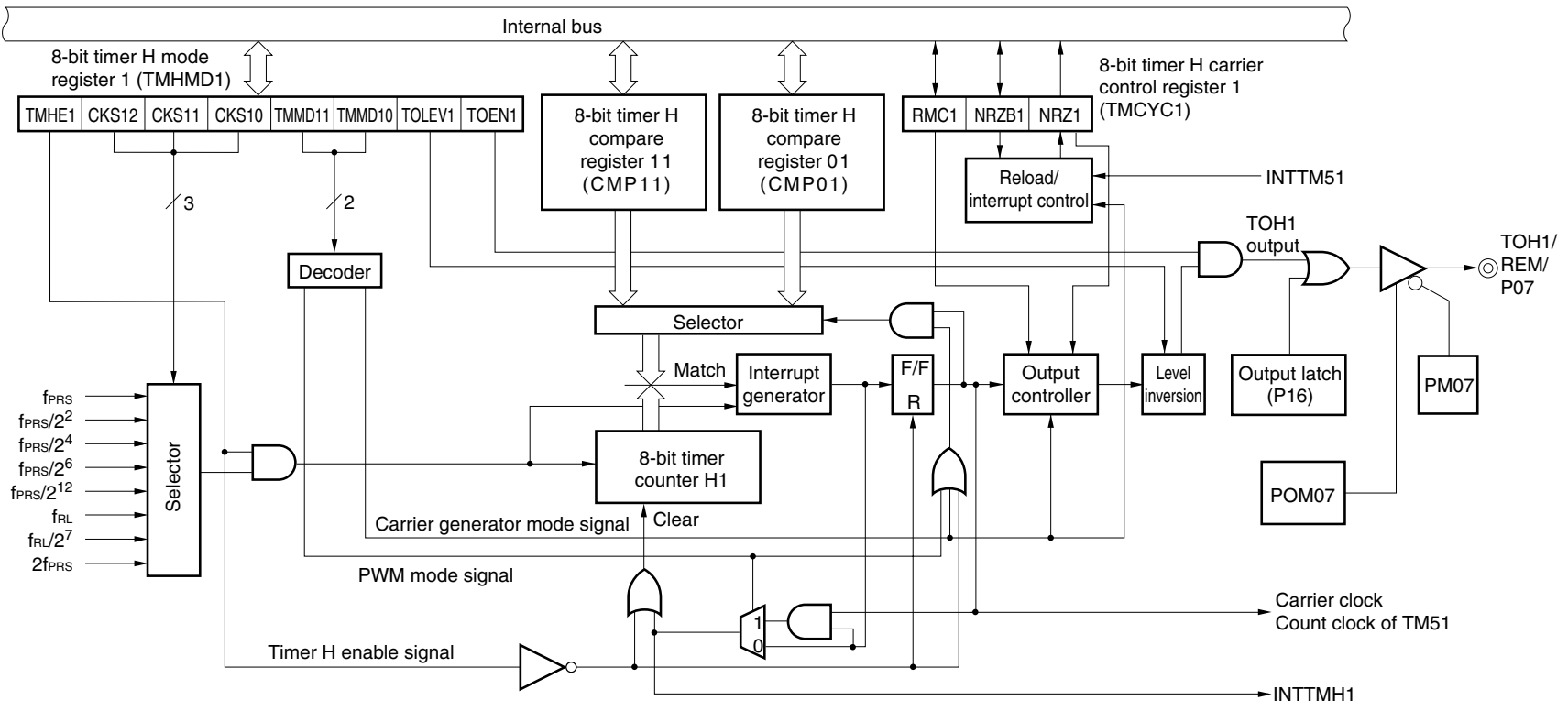


Figure 8-2. Block Diagram of 8-bit Timer H1



**(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 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)**

Address: FF18H (CMP00), FF1AH (CMP01) After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CMP0n (n = 0, 1)								

**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 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 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)**

Address: FF19H (CMP10), FF1BH (CMP11) After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CMP1n (n = 0, 1)								

**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)<sup>Note</sup>
- Port mode register 0 (PM0)
- Port register 0 (P0)
- Port mode register 0 (PM0)

**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

Symbol	<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		
				$f_{PRS} = 2 \text{ MHz}$	$f_{PRS} = 4 \text{ MHz}$
0	0	0	$f_{PRS}$	2 MHz	4 MHz
0	0	1	$f_{PRS}/2$	1 MHz	2 MHz
0	1	0	$f_{PRS}/2^2$	500 kHz	1 MHz
0	1	1	$f_{PRS}/2^6$	31.25 kHz	62.5 kHz
1	0	0	$f_{PRS}/2^{10}$	1.95 kHz	3.91 kHz
1	0	1	TM50 output <sup>Note</sup>		
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).

- Cautions**
- In the PWM output mode, be sure to set 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).
  - The actual TOH0/P01 pin output is determined depending on PM01 and P01, besides TOH0 output.

- Remarks**
- f<sub>PRS</sub>: Peripheral hardware clock frequency
  - 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

Symbol	<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 <sub>PRS</sub> = 2 MHz	f <sub>PRS</sub> = 4 MHz
0	0	0	f <sub>PRS</sub>	2 MHz	4 MHz
0	0	1	f <sub>PRS</sub> /2 <sup>2</sup>	500 kHz	1 MHz
0	1	0	f <sub>PRS</sub> /2 <sup>4</sup>	125 kHz	250 kHz
0	1	1	f <sub>PRS</sub> /2 <sup>6</sup>	31.25 kHz	62.5 kHz
1	0	0	f <sub>PRS</sub> /2 <sup>12</sup>	0.49 kHz	0.98 kHz
1	0	1	f <sub>RL</sub> /2 <sup>7</sup>	1.80 kHz (TYP.)	
1	1	0	2 <sub>PRS</sub>	4 kHz	8 MHz
1	1	1	f <sub>RL</sub>	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

<R>

- 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  $TOH1/REM/P07$  pin output is determined depending on  $PM07$  and  $P07$ , 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<sup>Note</sup>

Symbol	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 0 (PM0)**

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

When using the P01/TOH0 and P07/TOH1/REM pins for timer output, and carrier generator output for remote control, clear PM01 and PM07 and the output latches of P01 and P07 to 0.

PM0 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 (PM0)**

Address: FF20H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00

PM0n	P0n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

**(4) Port output mode resistors (POM0)**

This register set the output mode of port 0. Set POM07 to 1 when using the P07/TOH1/REM pin for carrier generator output for remote control.

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

Reset signal generation sets these registers to 00H.

**Figure 8-9. Format of Pull-up Resistor Option Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POM0	POM07	POM06	POM05	POM04	POM03	POM02	POM01	POM00	FF38H	00H	R/W

POM0n	P0n pin output mode selection (n = 0 to 7)
0	CMOS output
1	N-ch open-drain output (P07:P-ch open-drain output)

## 8.4 Operation of 8-bit Timers H0 and H1

### 8.4.1 Operation as interval timer/square-wave output

When 8-bit timer counter  $H_n$  and compare register  $0_n$  (CMP $0_n$ ) match, an interrupt request signal (INTTMH $n$ ) is generated and 8-bit timer counter  $H_n$  is cleared to 00H.

Compare register  $1_n$  (CMP $1_n$ ) is not used in interval timer mode. Since a match of 8-bit timer counter  $H_n$  and the CMP $1_n$  register is not detected even if the CMP $1_n$  register is set, timer output is not affected.

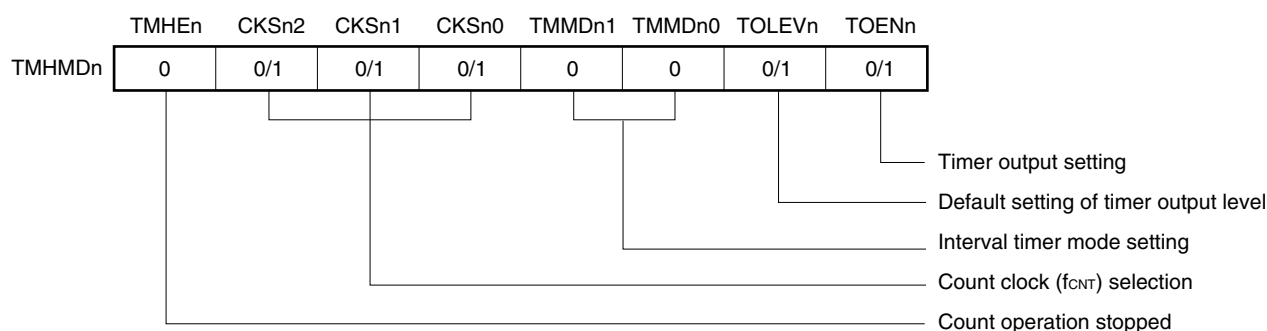
By setting bit 0 (TOEN $n$ ) of timer H mode register  $n$  (TMHMD $n$ ) to 1, a square wave of any frequency (duty = 50%) is output from TOH $n$ .

#### Setting

<1> Set each register.

Figure 8-10. Register Setting During Interval Timer/Square-Wave Output Operation

#### (i) Setting timer H mode register $n$ (TMHMD $n$ )



#### (ii) CMP $0_n$ 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 TMHE $n$  = 1.

<3> When the values of 8-bit timer counter  $H_n$  and the CMP $0_n$  register match, the INTTMH $n$  signal is generated and 8-bit timer counter  $H_n$  is cleared to 00H.

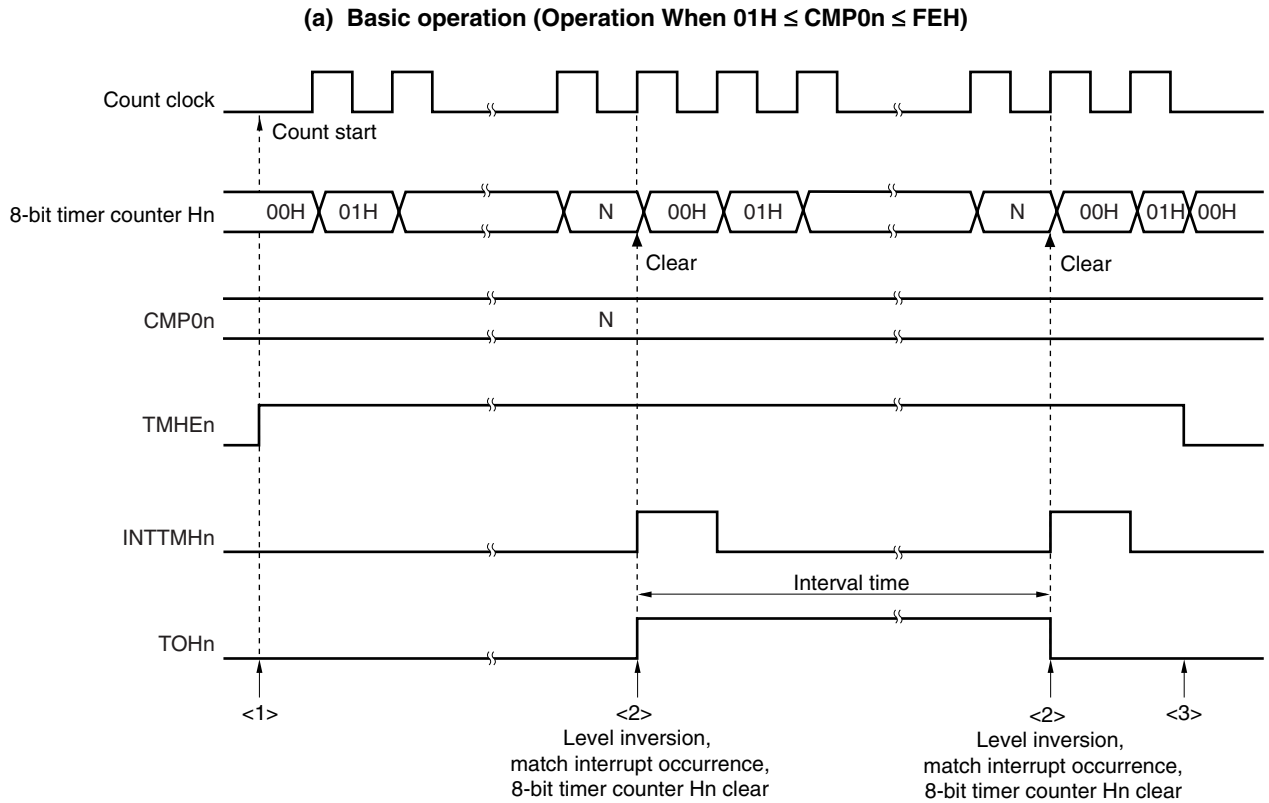
<4> Subsequently, the INTTMH $n$  signal is generated at the same interval. To stop the count operation, clear TMHE $n$  to 0.

**Remarks 1.** For the setting of the output pin, see **8.3 (3) Port mode register 0 (PM0)** and **(4) Port output mode resistors (POM0)**.

**2.** For how to enable the INTTMH $n$  signal interrupt, see **CHAPTER 11 INTERRUPT FUNCTIONS**.

**3.**  $n = 0, 1$

Figure 8-11. Timing of Interval Timer/Square-Wave Output Operation (1/2)

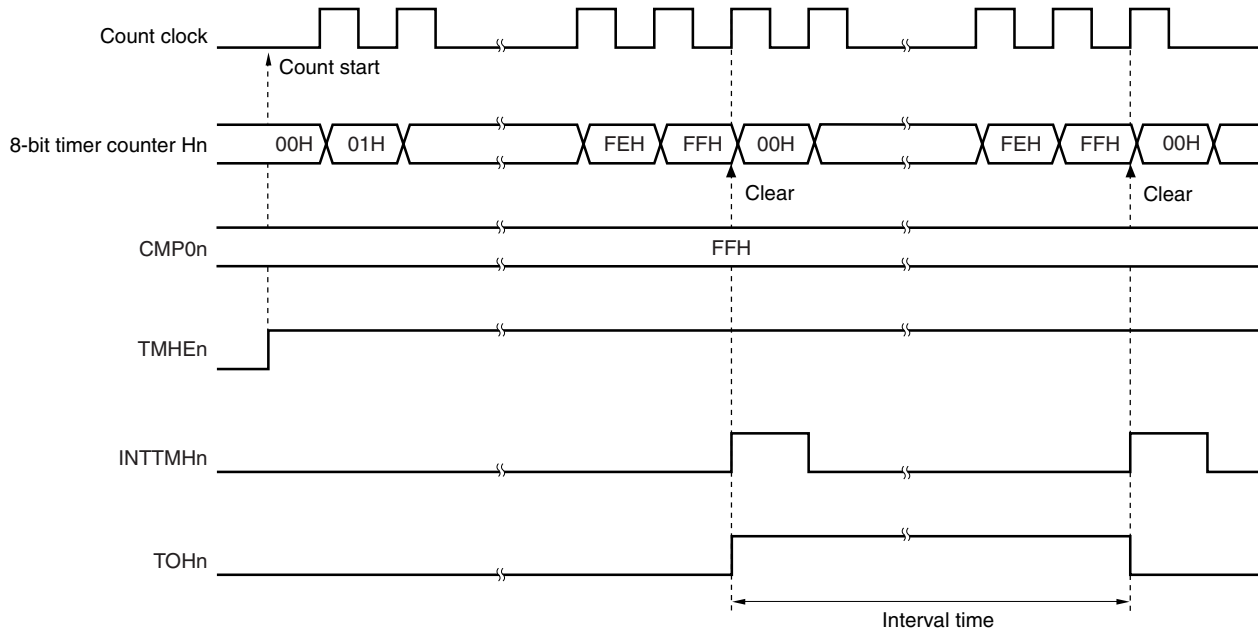


- <1> The count operation is enabled by setting the TMHEn bit to 1. The count clock starts counting no more than 1 clock after the operation is enabled.
- <2> When the value of the 8-bit timer counter Hn matches the value of the CMP0n register, the value of the timer counter is cleared, and the level 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.

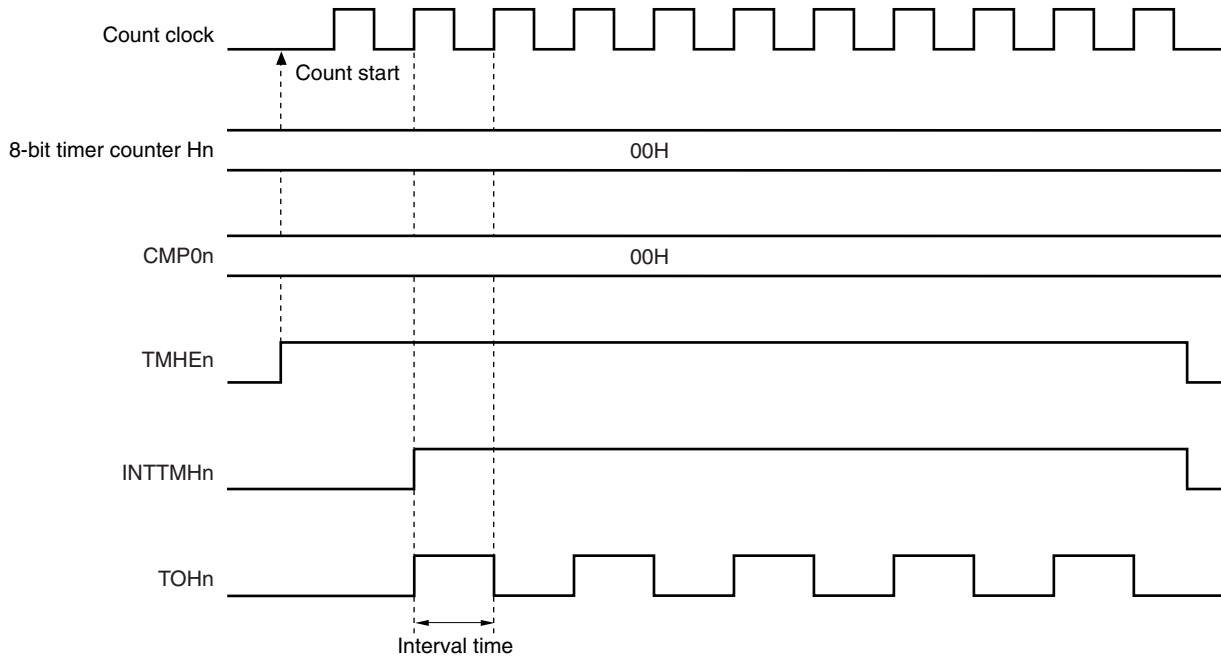
**Remark**  $n = 0, 1$   
 $01H \leq N \leq FEH$

Figure 8-11. Timing of Interval Timer/Square-Wave Output Operation (2/2)

(b) Operation when CMP0n = FFH



(c) Operation when CMP0n = 00H



**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.

8-bit timer compare register 0n (CMP0n) controls the cycle of timer output (TOHn). Rewriting the CMP0n register during timer operation is prohibited.

8-bit timer compare register 1n (CMP1n) controls the duty of timer output (TOHn). Rewriting the CMP1n register during timer operation is possible.

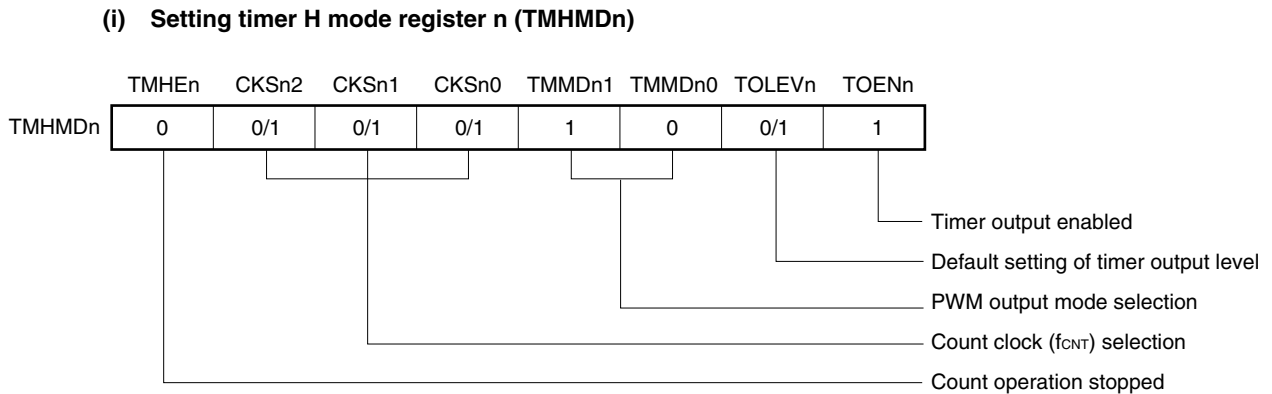
The operation in PWM output mode is as follows.

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 8-12. 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 8-bit timer counter Hn and the CMP0n register match, 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 8-bit timer counter Hn is changed from the CMP0n register to the CMP1n register.

<4> When 8-bit timer counter Hn and the CMP1n register match, an inactive level is output and the compare register to be compared with 8-bit timer counter Hn is changed from the CMP1n register to the CMP0n register. At this time, 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.

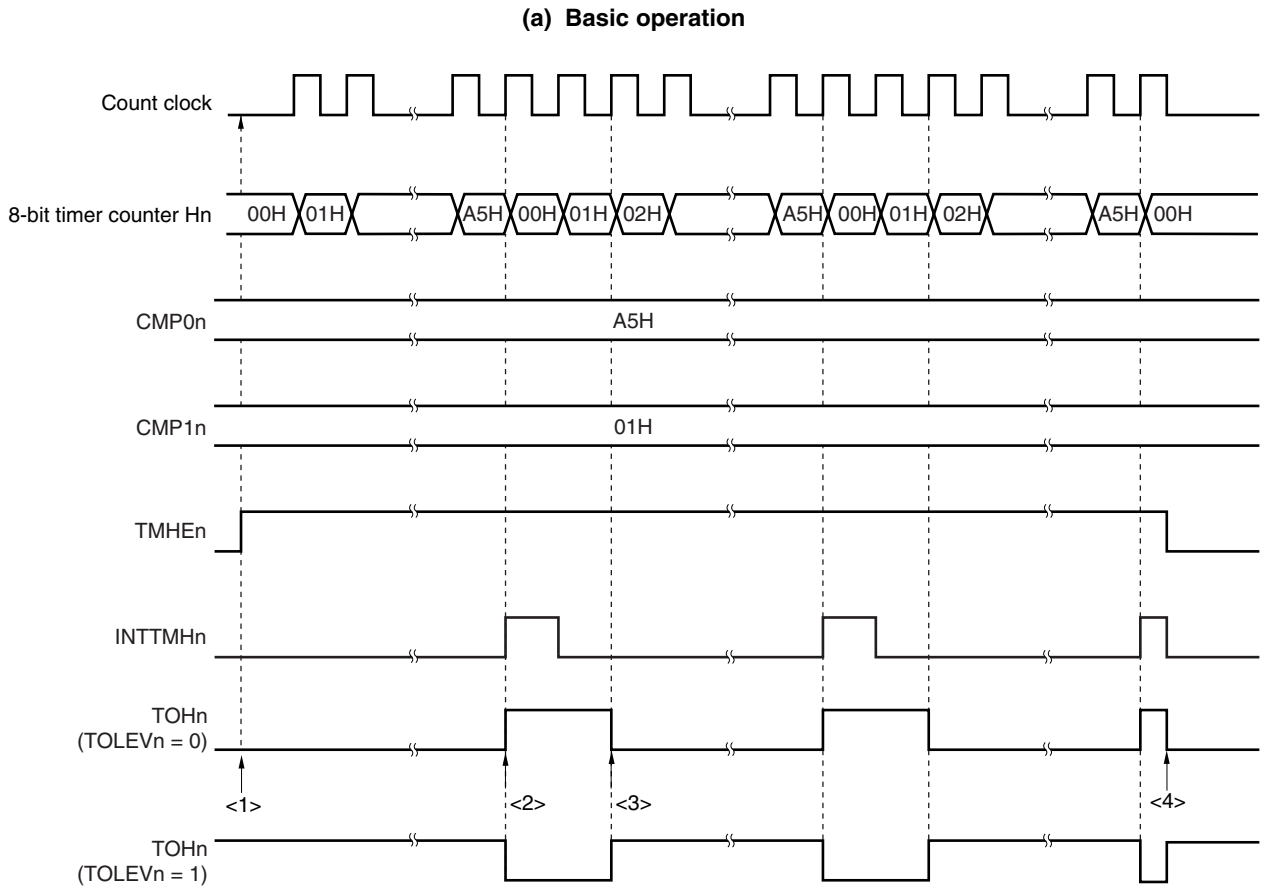
$$00H \leq \text{CMP1n (M)} < \text{CMP0n (N)} \leq \text{FFH}$$

**Remarks** 1. For the setting of the output pin, see 8.3 (3) Port mode register 0 (PM0) and (4) Port output mode resistors (POM0).

2. For details on how to enable the INTTMHn signal interrupt, see CHAPTER 11 INTERRUPT FUNCTIONS.

3. n = 0, 1

Figure 8-13. Operation Timing in PWM Output Mode (1/4)

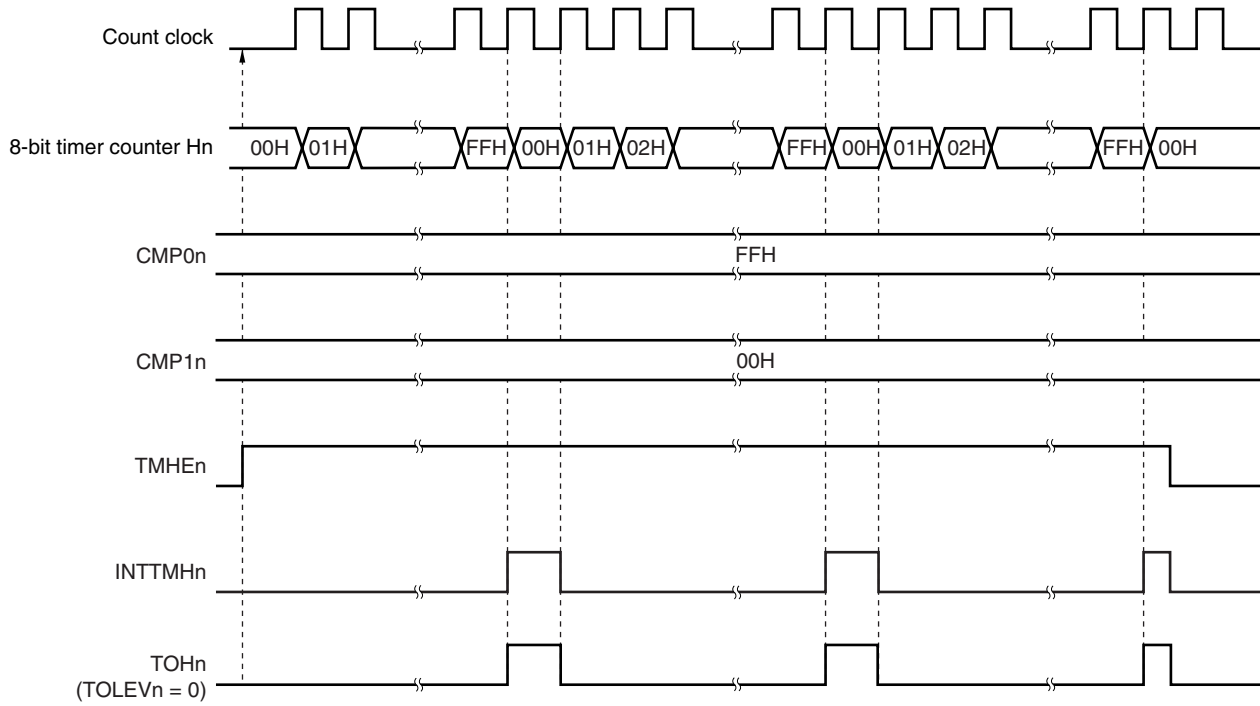


- <1> The count operation is enabled by setting the TMHEn bit to 1. Start 8-bit timer counter Hn by masking one count clock to count up. At this time, PWM output outputs an inactive level.
- <2> When the values of 8-bit timer counter Hn and the CMP0n register match, an active level is output. At this time, the value of 8-bit timer counter Hn is cleared, and the INTTMHn signal is output.
- <3> When the values of 8-bit timer counter Hn and the CMP1n register match, an inactive level is output. At this time, the 8-bit 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.

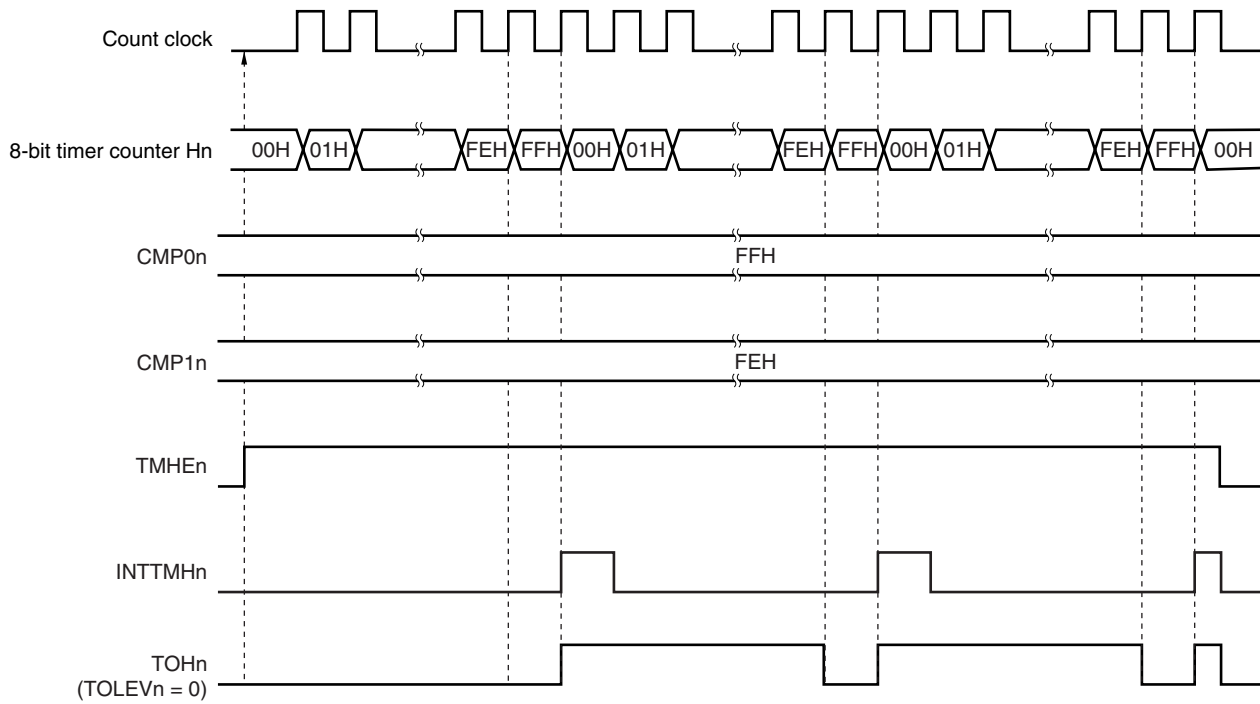
**Remark** n = 0, 1

Figure 8-13. Operation Timing in PWM Output Mode (2/4)

(b) Operation when  $CMP0n = FFH$ ,  $CMP1n = 00H$



(c) Operation when  $CMP0n = FFH$ ,  $CMP1n = FEH$

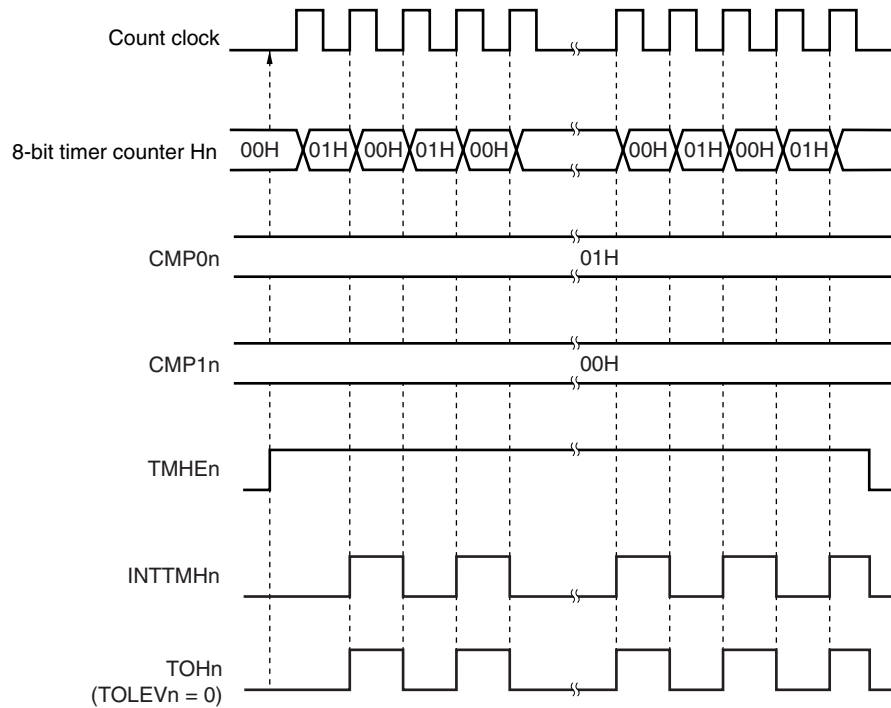


**Remark** n = 0, 1



Figure 8-13. Operation Timing in PWM Output Mode (3/4)

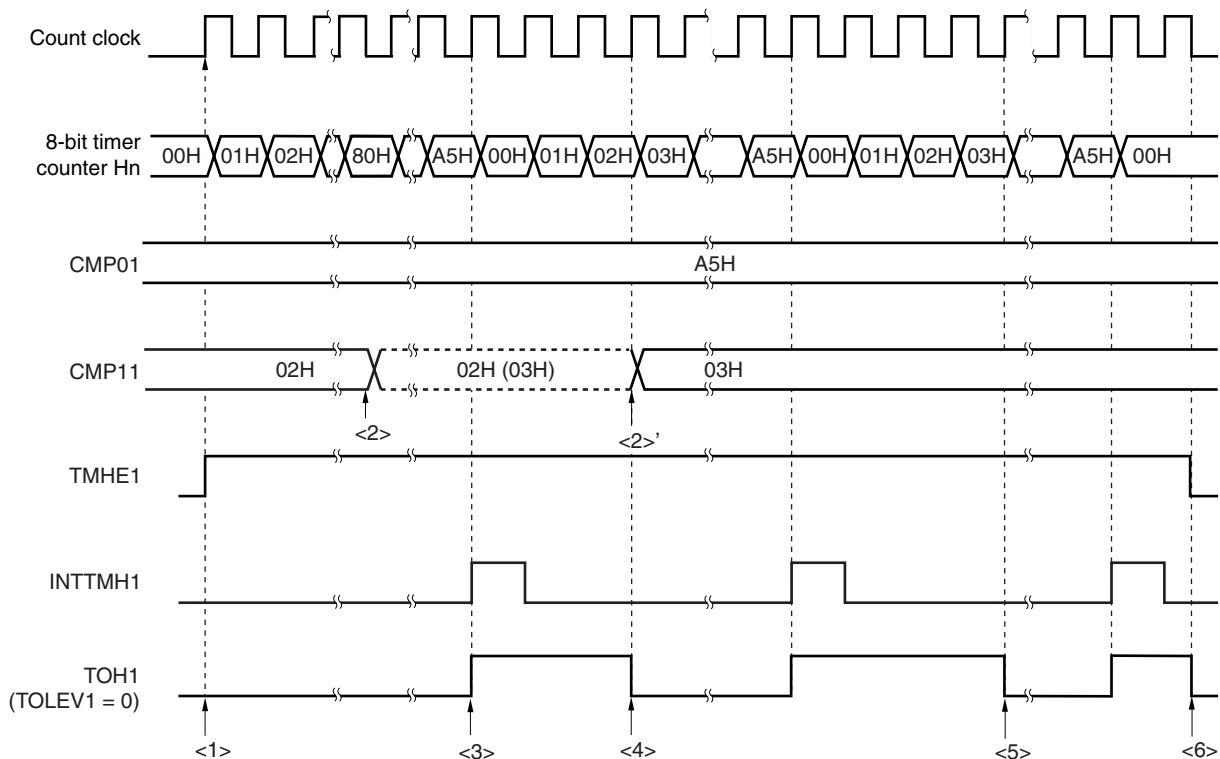
(d) Operation when  $CMP0n = 01H$ ,  $CMP1n = 00H$



**Remark** n = 0, 1

Figure 8-13. 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 TMHE<sub>n</sub> = 1. Start 8-bit timer counter H<sub>n</sub> by masking one count clock to count up. At this time, PWM output outputs an inactive level.
- <2> The CMP1<sub>n</sub> register value can be changed during timer counter operation. This operation is asynchronous to the count clock.
- <3> When the values of 8-bit timer counter H<sub>n</sub> and the CMP0<sub>n</sub> register match, the value of 8-bit timer counter H<sub>n</sub> is cleared, an active level is output, and the INTTMH<sub>n</sub> signal is output.
- <4> If the CMP1<sub>n</sub> register value is changed, the value is latched and not transferred to the register. When the values of 8-bit timer counter H<sub>n</sub> and the CMP1<sub>n</sub> register before the change match, the value is transferred to the CMP1<sub>n</sub> register and the CMP1<sub>n</sub> register value is changed (<2>'). However, three count clocks or more are required from when the CMP1<sub>n</sub> 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 8-bit timer counter H<sub>n</sub> and the CMP1<sub>n</sub> register after the change match, an inactive level is output. 8-bit timer counter H<sub>n</sub> is not cleared and the INTTMH<sub>n</sub> signal is not generated.
- <6> Clearing the TMHE<sub>n</sub> bit to 0 during timer H<sub>n</sub> operation sets the INTTMH<sub>n</sub> 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, 8-bit timer H1 is used to generate the carrier signal of an infrared remote controller, and 8-bit timer/event counter 51 is used to generate an infrared remote control signal (time count).

The carrier clock generated by 8-bit timer H1 is output in the cycle set by 8-bit timer/event counter 51.

In carrier generator mode, the output of the 8-bit timer H1 carrier pulse is controlled by 8-bit timer/event counter 51, and the carrier pulse is output from the REM output.

#### (1) Carrier generation

In carrier generator mode, 8-bit timer H compare register 01 (CMP01) generates a low-level width carrier pulse waveform and 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.

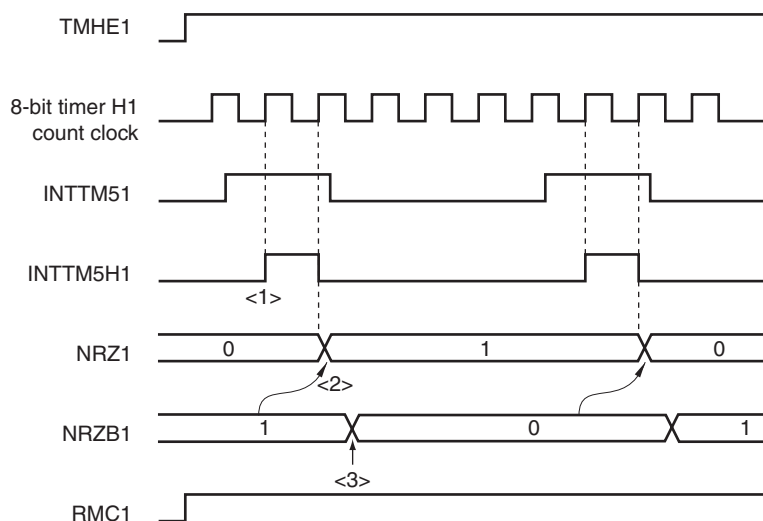
#### (2) Carrier output control

Carrier output is controlled by the interrupt request signal (INTTM51) of 8-bit timer/event counter 51 and the NRZB1 and RMC1 bits of the 8-bit timer H carrier control register 1 (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 8-14. Transfer Timing



- <1> The INTTM51 signal is synchronized with the count clock of 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.

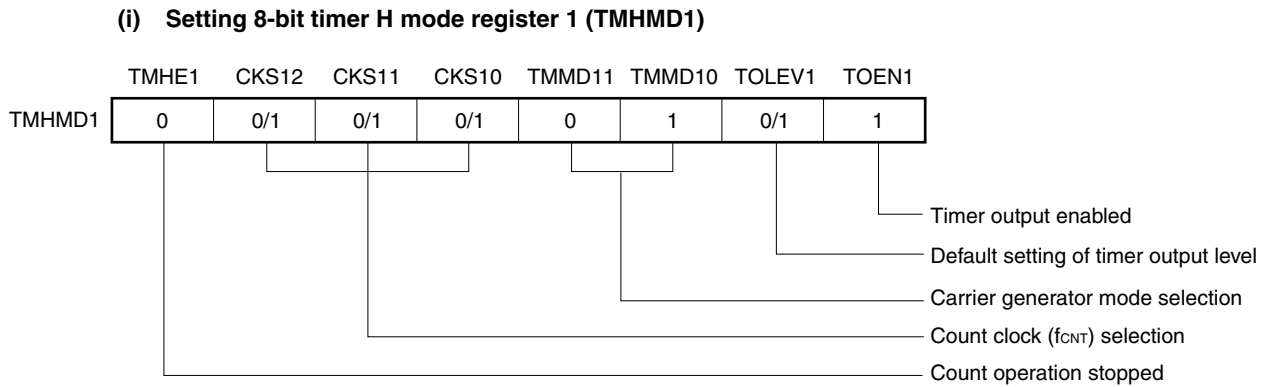
- Cautions**
1. Do not rewrite the NRZB1 bit again until at least the second clock after it has been rewritten, or else the transfer from the NRZB1 bit to the NRZ1 bit is not guaranteed.
  2. When 8-bit timer/event counter 51 is used in the carrier generator mode, an interrupt is generated at the timing of <1>. When 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 8-15. 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, 8-bit timer H1 starts counting.

<3> When TCE51 of 8-bit timer mode control register 51 (TMC51) is set to 1, 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 8-bit timer counter H1 and the CMP01 register value match, the INTTMH1 signal is generated, 8-bit timer counter H1 is cleared. At the same time, the compare register to be compared with 8-bit timer counter H1 is switched from the CMP01 register to the CMP11 register.

<5> When the count value of 8-bit timer counter H1 and the CMP11 register value match, the INTTMH1 signal is generated, 8-bit timer counter H1 is cleared. At the same time, the compare register to be compared with 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 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 REM output.

<10> By performing the procedures above, an arbitrary carrier clock is obtained. To stop the count operation, clear TMHE1 to 0.

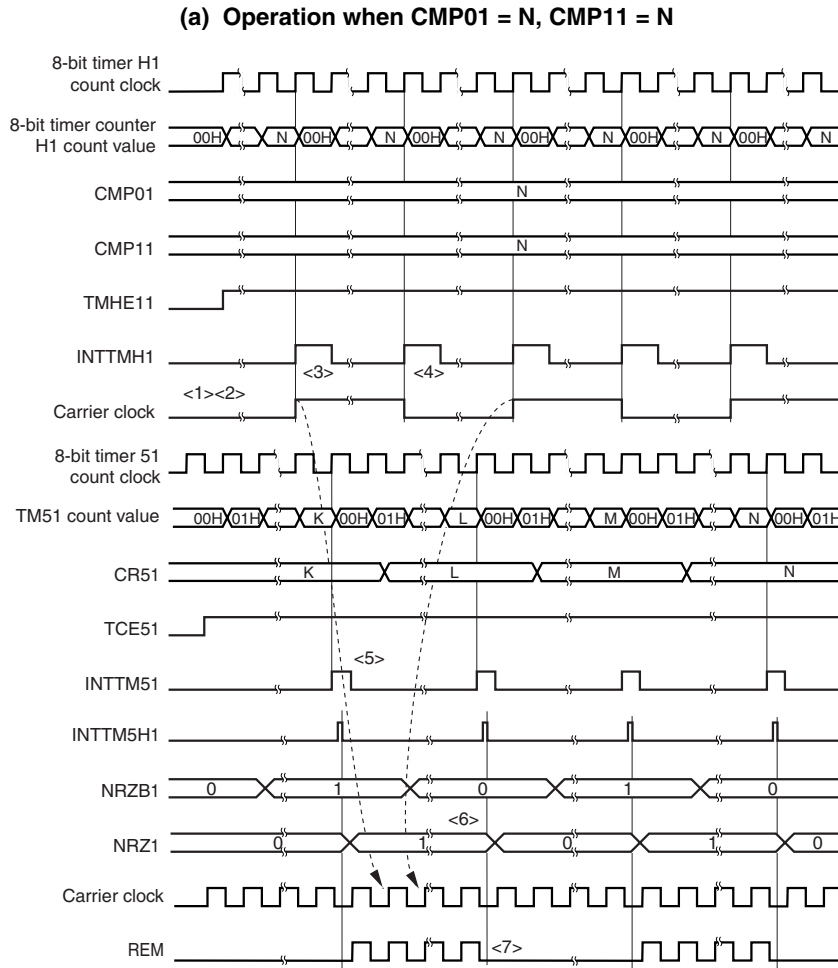
If the setting value of the CMP01 register is N, the setting value of the CMP11 register is M, and the count clock frequency is  $f_{CNT}$ , the carrier clock output cycle and duty are as follows.

- 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 0 (PM0) and (4) Port output mode resistors (POM0).
  2. For how to enable the INTTMH1 signal interrupt, see CHAPTER 11 INTERRUPT FUNCTIONS.

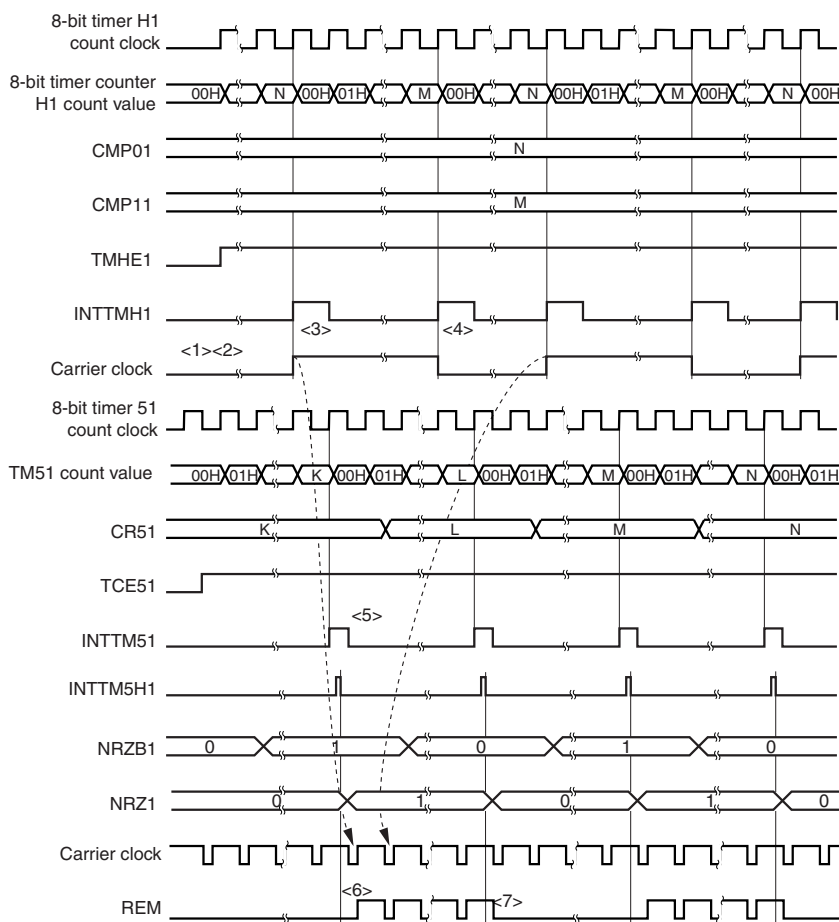
Figure 8-16. Carrier Generator Mode Operation Timing (1/3)



- <1> When TMHE1 = 0 and TCE51 = 0, 8-bit timer counter H1 operation is stopped.
- <2> When TMHE1 = 1 is set, 8-bit timer counter H1 starts a count operation. At that time, the carrier clock remains default.
- <3> When the count value of 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 8-bit timer counter H1 is switched from the CMP01 register to the CMP11 register. 8-bit timer counter H1 is cleared to 00H.
- <4> When the count value of 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 8-bit timer counter H1 is switched from the CMP11 register to the CMP01 register. 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 REM output becomes low level.

**Remark** INTTM5H1 is an internal signal and not an interrupt source.

Figure 8-16. Carrier Generator Mode Operation Timing (2/3)

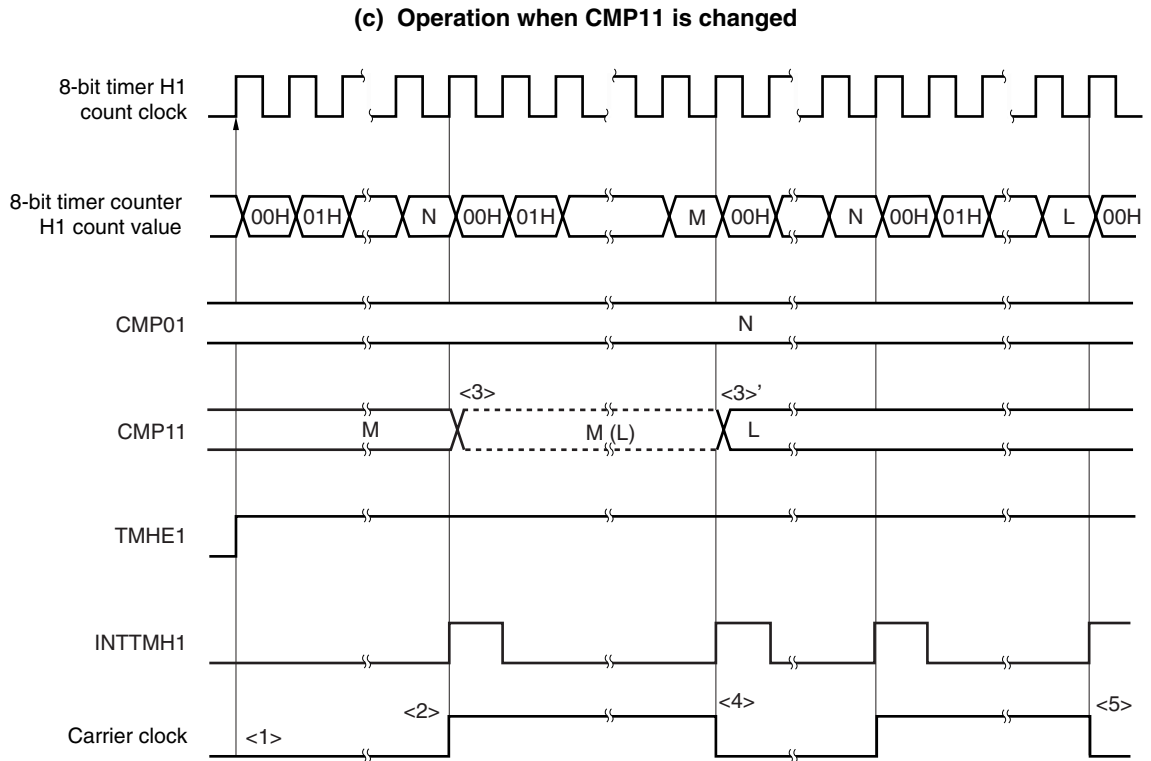
(b) Operation when  $CMP01 = N$ ,  $CMP11 = M$ 

- <1> When  $TMHE1 = 0$  and  $TCE51 = 0$ , 8-bit timer counter H1 operation is stopped.
- <2> When  $TMHE1 = 1$  is set, 8-bit timer counter H1 starts a count operation. At that time, the carrier clock remains default.
- <3> When the count value of 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 8-bit timer counter H1 is switched from the  $CMP01$  register to the  $CMP11$  register. 8-bit timer counter H1 is cleared to 00H.
- <4> When the count value of 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 8-bit timer counter H1 is switched from the  $CMP11$  register to the  $CMP01$  register. 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  $REM$  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 8-16. Carrier Generator Mode Operation Timing (3/3)



- <1> When  $TMHE1 = 1$  is set, 8-bit timer H1 starts a count operation. At that time, the carrier clock remains default.
- <2> When the count value of 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 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 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 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 8-bit timer counter H1 is changed from the CMP11 register to the CMP01 register.
- <5> The timing at which the count value of 8-bit timer counter H1 and the CMP11 register value match again is indicated by the value after the change (L).

## CHAPTER 9 WATCHDOG TIMER

### 9.1 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 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 14 RESET FUNCTION**.

## 9.2 Configuration of Watchdog Timer

The watchdog timer includes the following hardware.

**Table 9-1. Configuration of Watchdog Timer**

Item	Configuration
Control register	Watchdog timer enable register (WDTE)

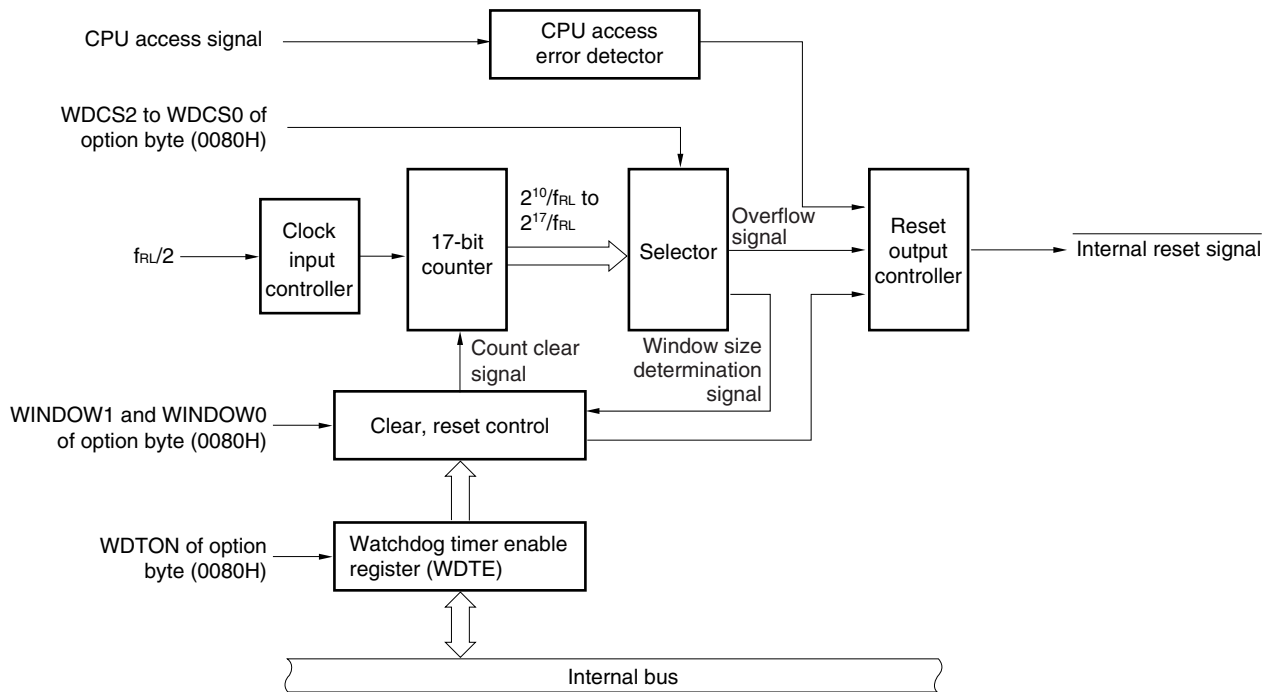
How the counter operation is controlled, overflow time, and window open period are set by the option byte.

**Table 9-2. 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 17 OPTION BYTE**.

**Figure 9-1. Block Diagram of Watchdog Timer**



### 9.3 Register Controlling Watchdog Timer

The watchdog timer is controlled by the watchdog timer enable register (WDTE).

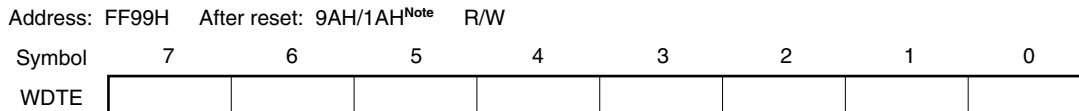
#### (1) Watchdog timer enable register (WDTE)

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<sup>Note</sup>.

**Figure 9-2. Format of Watchdog Timer Enable Register (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)).

## 9.4 Operation of Watchdog Timer

### 9.4.1 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 17**).

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 **9.4.2** and **CHAPTER 17**).
  - Set a window open period by using bits 6 and 5 (WINDOW1 and WINDOW0) of the option byte (0080H) (for details, see **9.4.3** and **CHAPTER 17**).
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 register (detection of an invalid check during a CPU program loop)
    - If the CPU accesses an area not set by the IMS register (excluding FB00H 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.

<R> **5.** The watchdog timer continues its operation during self programming of the flash memory. During processing, the interrupt acknowledge time is delayed. Set the overflow time and window size taking this delay into consideration.

**9.4.2 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.

<R>

**Table 9-3. Setting of Overflow Time of Watchdog Timer (when 2.1 V ≤ V<sub>DD</sub> ≤ 3.6 V)**

WDCS2	WDCS1	WDCS0	Overflow Time of Watchdog Timer
0	0	0	2 <sup>10</sup> /f <sub>RL</sub> (3.71 ms)
0	0	1	2 <sup>11</sup> /f <sub>RL</sub> (7.42 ms)
0	1	0	2 <sup>12</sup> /f <sub>RL</sub> (14.84 ms)
0	1	1	2 <sup>13</sup> /f <sub>RL</sub> (29.68 ms)
1	0	0	2 <sup>14</sup> /f <sub>RL</sub> (59.36 ms)
1	0	1	2 <sup>15</sup> /f <sub>RL</sub> (118.72 ms)
1	1	0	2 <sup>16</sup> /f <sub>RL</sub> (237.45 ms)
1	1	1	2 <sup>17</sup> /f <sub>RL</sub> (474.90 ms)

**Cautions 1.** The combination of WDCS2 = WDCS1 = WDCS0 = 0 and WINDOW1 = WINDOW0 = 0 is prohibited.

**2.** The watchdog timer continues its operation during self programming of the flash memory. During processing, the interrupt acknowledge time is delayed. Set the overflow time and window size taking this delay into consideration.

**Remarks 1.** f<sub>RL</sub>: Internal low-speed oscillation clock frequency  
**2.** ( ): f<sub>RL</sub> = 276 kHz (MAX.)

&lt;R&gt;

**Table 9-4. Setting of Overflow Time of Watchdog Timer (when  $1.8\text{ V} \leq V_{DD} < 2.1\text{ V}$ )**

WDCS2	WDCS1	WDCS0	Overflow Time of Watchdog Timer
0	0	0	$2^{10}/f_{RL}$ (3.41ms)
0	0	1	$2^{11}/f_{RL}$ (6.83 ms)
0	1	0	$2^{12}/f_{RL}$ (13.65 ms)
0	1	1	$2^{13}/f_{RL}$ (27.31 ms)
1	0	0	$2^{14}/f_{RL}$ (54.61 ms)
1	0	1	$2^{15}/f_{RL}$ (109.23 ms)
1	1	0	$2^{16}/f_{RL}$ (218.45 ms)
1	1	1	$2^{17}/f_{RL}$ (436.91 ms)

**Cautions** 1. The combination of  $WDCS2 = WDCS1 = WDCS0 = 0$  and  $WINDOW1 = WINDOW0 = 0$  is prohibited.

2. The watchdog timer continues its operation during self programming of the flash memory. During processing, the interrupt acknowledge time is delayed. Set the overflow time and window size taking this delay into consideration.

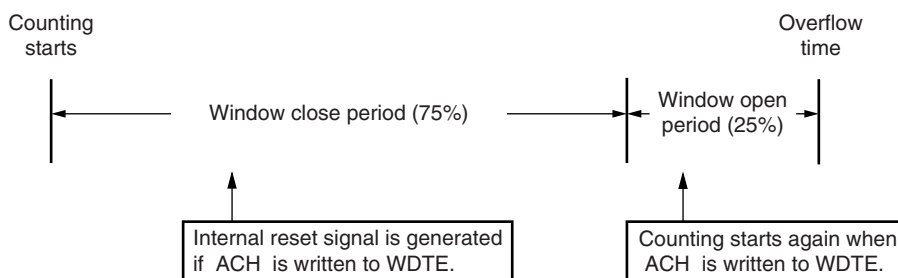
**Remarks** 1.  $f_{RL}$ : Internal low-speed oscillation clock frequency  
 2. (.) :  $f_{RL} = 300\text{ kHz (MAX.)}$

### 9.4.3 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 9-5. 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%

**Cautions 1.** The combination of WDCS2 = WDCS1 = WDCS0 = 0 and WINDOW1 = WINDOW0 = 0 is prohibited.

<R> **2.** The watchdog timer continues its operation during self programming of the flash memory. During processing, the interrupt acknowledge time is delayed. Set the overflow time and window size taking this delay into consideration.

**Remark** If the overflow time is set to  $2^{10}/f_{RL}$ , the window close time and open time are as follows.

<R> (when  $2.1\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ )

	Setting of Window Open Period			
	25%	50%	75%	100%
Window close time	0 to 3.56 ms	0 to 2.37 ms	0 to 1.19 ms	None
Window open time	3.56 to 3.71 ms	2.37 to 3.71 ms	1.19 to 3.71 ms	0 to 3.71 ms

<When window open period is 25%>

- Overflow time:  
 $2^{10}/f_{RL} \text{ (MAX.)} = 2^{10}/276\text{ kHz (MAX.)} = 3.71\text{ ms}$
- Window close time:  
 $0\text{ to }2^{10}/f_{RL} \text{ (MIN.)} \times (1 - 0.25) = 0\text{ to }2^{10}/216\text{ kHz (MIN.)} \times 0.75 = 0\text{ to }3.56\text{ ms}$
- Window open time:  
 $2^{10}/f_{RL} \text{ (MIN.)} \times (1 - 0.25)\text{ to }2^{10}/f_{RL} \text{ (MAX.)} = 2^{10}/216\text{ kHz (MIN.)} \times 0.75\text{ to }2^{10}/276\text{ kHz (MAX.)}$   
 $= 3.56\text{ to }3.71\text{ ms}$

<R> (when  $1.8\text{ V} \leq V_{DD} < 2.1\text{ V}$ )

	Setting of Window Open Period			
	25%	50%	75%	100%
Window close time	Setting prohibited	0 to 2.84 ms	0 to 1.42 ms	None
Window open time	Setting prohibited	2.84 to 3.41 ms	1.42 to 3.41 ms	0 to 3.41 ms

<When window open period is 25%>

- Overflow time:  
 $2^{10}/f_{RL} \text{ (MAX.)} = 2^{10}/300\text{ kHz (MAX.)} = 3.41\text{ ms}$
- Window close time:  
 $0\text{ to }2^{10}/f_{RL} \text{ (MIN.)} \times (1 - 0.5) = 0\text{ to }2^{10}/180\text{ kHz (MIN.)} \times 0.5 = 0\text{ to }2.84\text{ ms}$
- Window open time:  
 $2^{10}/f_{RL} \text{ (MIN.)} \times (1 - 0.5)\text{ to }2^{10}/f_{RL} \text{ (MAX.)} = 2^{10}/180\text{ kHz (MIN.)} \times 0.5\text{ to }2^{10}/300\text{ kHz (MAX.)}$   
 $= 2.84\text{ to }3.41\text{ ms}$



## CHAPTER 10 SERIAL INTERFACE UART6

### 10.1 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 **10.4.1 Operation stop mode**.

#### (2) Asynchronous serial interface (UART) mode

The functions of this mode are outlined below.

For details, see **10.4.2 Asynchronous serial interface (UART) mode** and **10.4.3 Dedicated baud rate generator**.

- Maximum transfer rate: 312.5 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

- 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 ( $f_{CLK6}$ ) 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 ( $f_{CLK6}$ ) 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.

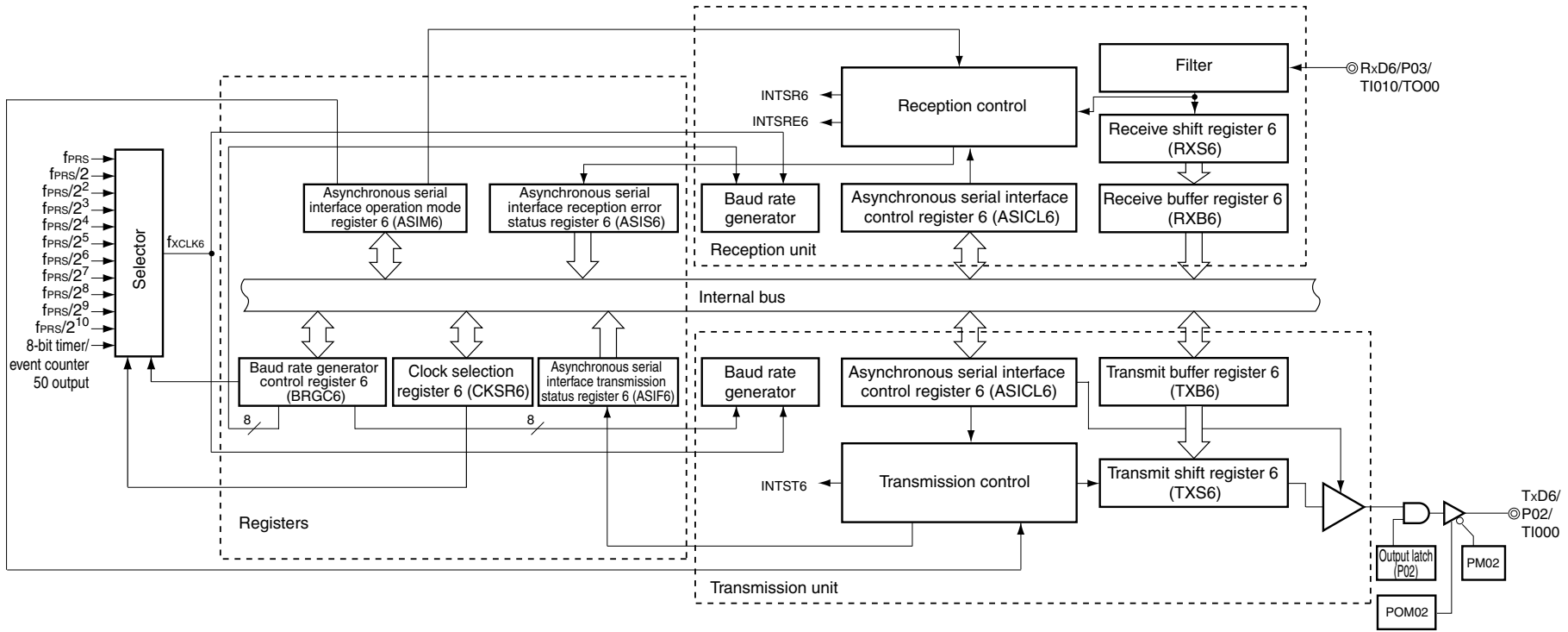
## 10.2 Configuration of Serial Interface UART6

Serial interface UART6 includes the following hardware.

**Table 10-1. 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) Port mode register 0 (PM0) Port register 0 (P0) Port output mode register 0 (POM0)

Figure 10-1. Block Diagram of Serial Interface UART6



**(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.

### 10.3 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)
- Port mode register 0 (PM0)
- Port register 0 (P0)
- Port output mode register 0 (POM0)

#### (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 10-2. 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 <sup>Note 1</sup>	Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit <sup>Note 2</sup> .							
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. The output of the TxD6 pin goes high level and the input from the RxD6 pin is fixed to the high level when POWER6 = 0 during transmission.
  2. Asynchronous serial interface reception error status register 6 (ASIS6), asynchronous serial interface transmission status register 6 (ASIF6), and receive buffer register 6 (RXB6) are reset.

Figure 10-2. 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 <sup>Note</sup>
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<sub>CLK6</sub>) 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<sub>CLK6</sub>) after setting TXE6 = 1.
  6. Clear the TXE6 and RXE6 bits to 0 before rewriting the PS61, PS60, and CL6 bits.
  7. 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.
  8. 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 10-3. 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.

**(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 10-4. 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 clears 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 10-5. 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 <sub>CLK6</sub> ) selection	
					f <sub>PRS</sub> = 2 MHz	f <sub>PRS</sub> = 4 MHz
0	0	0	0	f <sub>PRS</sub>	2 MHz	4 MHz
0	0	0	1	f <sub>PRS</sub> /2	1 MHz	2 MHz
0	0	1	0	f <sub>PRS</sub> /2 <sup>2</sup>	500 kHz	1 MHz
0	0	1	1	f <sub>PRS</sub> /2 <sup>3</sup>	250 kHz	500 kHz
0	1	0	0	f <sub>PRS</sub> /2 <sup>4</sup>	125 kHz	250 kHz
0	1	0	1	f <sub>PRS</sub> /2 <sup>5</sup>	62.5 kHz	125 kHz
0	1	1	0	f <sub>PRS</sub> /2 <sup>6</sup>	31.25 kHz	62.5 kHz
0	1	1	1	f <sub>PRS</sub> /2 <sup>7</sup>	15.625 kHz	31.25 kHz
1	0	0	0	f <sub>PRS</sub> /2 <sup>8</sup>	7.813 kHz	15.625 kHz
1	0	0	1	f <sub>PRS</sub> /2 <sup>9</sup>	3.906 kHz	7.813 kHz
1	0	1	0	f <sub>PRS</sub> /2 <sup>10</sup>	1.953 kHz	3.906 kHz
1	0	1	1	TM50 output <sup>Note</sup>		
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<sub>PRS</sub>: 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 10-6. 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).

**Figure 10-7. Format of Asynchronous Serial Interface Control Register 6 (ASICL6)**

Address: FF58H After reset: 16H R/W

Symbol	7	6	5	4	3	2	1	0
ASICL6	0	0	0	0	0	0	DIR6	TXDLV6

DIR6	First-bit specification
0	MSB
1	LSB

TXDLV6	Enables/disables inverting TxD6 output
0	Normal output of TxD6
1	Inverted output of TxD6

**Caution** Before rewriting the DIR6 and TXDLV6 bits, clear the TXE6 and RXE6 bits to 0.

**(7) Port mode register 1 (PM1)**

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

When using the P02/TxD6/TI000 pin for serial interface data output, clear PM02 to 0 and set the output latch of P02 to 1.

When using the P03/RxD6/TI010/TO00 pin for serial interface data input, set PM03 to 1. The output latch of P03 at this time may be 0 or 1.

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

Reset signal generation sets this register to FFH.

**Figure 10-8. Format of Port Mode Register 0 (PM0)**

Address: FF20H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00

PM0n	P0n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

**(8) Port output mode resistors (POM0)**

This register set the output mode of port 0.

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

Reset signal generation sets these registers to 00H.

**Figure 10-9. Format of Pull-up Resistor Option Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POM0	POM07	POM06	POM05	POM04	POM03	POM02	POM01	POM00	FF38H	00H	R/W

POM0n	P0n pin output mode selection (n = 0 to 7)
0	CMOS output
1	N-ch open-drain output (P07:P-ch open-drain output)

## 10.4 Operation of Serial Interface UART6

Serial interface UART6 has the following two modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode

### 10.4.1 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 <sup>Note 1</sup>	Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit <sup>Note 2</sup> .							
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. The output of the TxD6 pin goes high and the input from the RxD6 pin is fixed to high level when POWER6 = 0 during transmission.
  2. Asynchronous serial interface reception error status register 6 (ASIS6), asynchronous serial interface transmission status register 6 (ASIF6), 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/P03/TI010/TO00 and TxD6/P02/TI000 pins as general-purpose port pins, see CHAPTER 4 PORT FUNCTIONS.

**10.4.2 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)
- Port mode register 0 (PM0)
- Port register 0 (P0)
- Port output mode register 0 (POM0)

The basic procedure of setting an operation in the UART mode is as follows.

- <1> Set the CKSR6 register (see **Figure 10-5**).
- <2> Set the BRGC6 register (see **Figure 10-6**).
- <3> Set bits 0 to 4 (ISRM6, SL6, CL6, PS60, PS61) of the ASIM6 register (see **Figure 10-2**).
- <4> Set bits 0 and 1 (TXDLV6, DIR6) of the ASICL6 register (see **Figure 10-7**).
- <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 the 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 10-2. Relationship Between Register Settings and Pins**

POWER6	TXE6	RXE6	PM02	P02	PM03	P03	UART6 Operation	Pin Function	
								TxD6/P02/TI000	RxD6/P03/TI010/TO00
0	0	0	×	×	×	×	Stop	P02/TI000	P03/ TI010/TO00
1	0	1	×	×	1	×	Reception	P02/TI000	RxD6
	1	0	0	1	×	×	Transmission	TxD6	P03/TI010/TO00
	1	1	0	1	1	×	Transmission/reception	TxD6	RxD6

**Note** Can be set as port function or 16-bit timer/event counter 00.

<b>Remark</b>	×	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	
PM0×	Port mode register	
P0×	Port output latch	

## (2) Communication operation

### (a) Format and waveform example of normal transmit/receive data

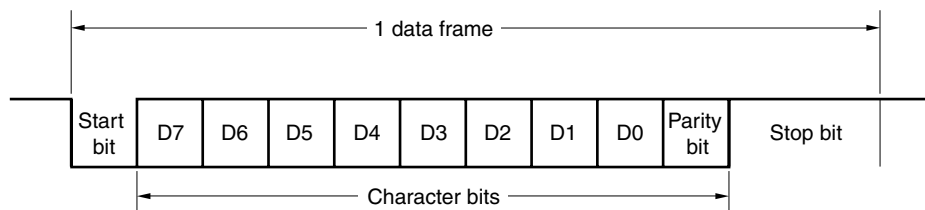
Figures 10-10 and 10-11 show the format and waveform example of the normal transmit/receive data.

**Figure 10-10. 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).

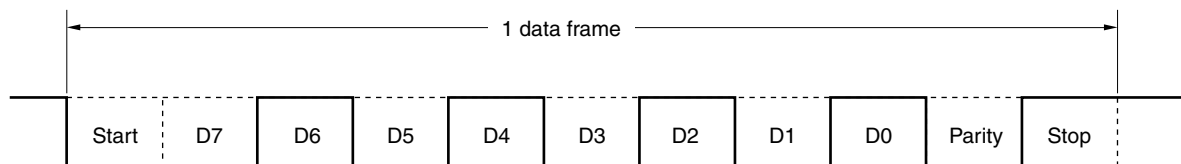
Whether the Tx/D6 pin outputs normal or inverted data is specified by bit 0 (TXDLV6) of ASICL6.

Figure 10-11. Example of Normal UART Transmit/Receive Data Waveform

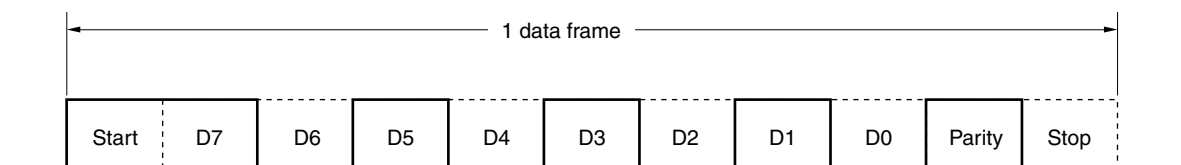
1. Data length: 8 bits, LSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H



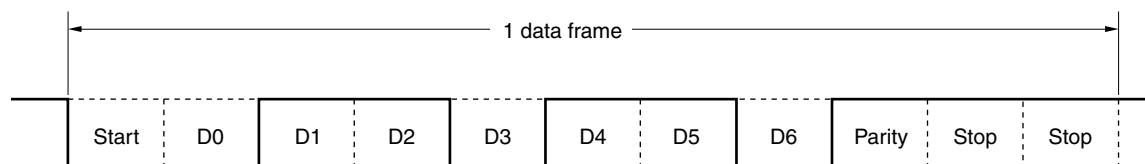
2. Data length: 8 bits, MSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H



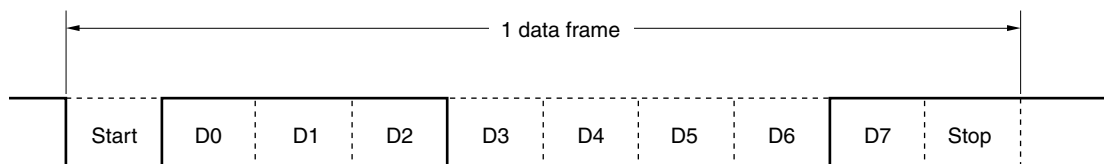
3. Data length: 8 bits, MSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H, Tx/D6 pin inverted output



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.

**(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**

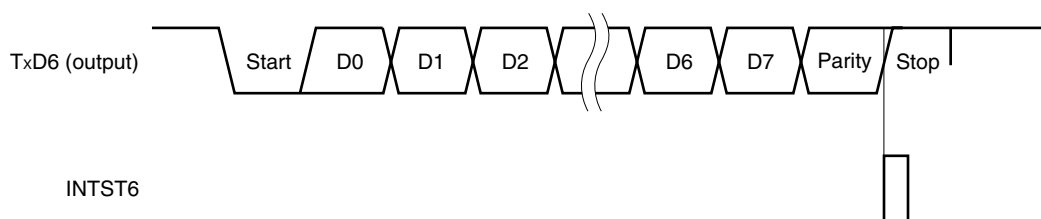
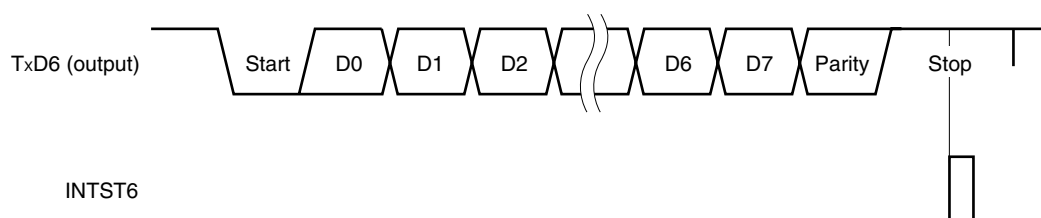
When bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1 and bit 6 (TXE6) of ASIM6 is then set to 1, transmission is enabled. Transmission can be started by writing transmit data to transmit buffer register 6 (TXB6). The start bit, parity bit, and stop bit are 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 10-12 shows the timing of the transmission completion interrupt request (INTST6). This interrupt occurs as soon as the last stop bit has been output.

**Figure 10-12. 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.

**Caution** 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.

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.

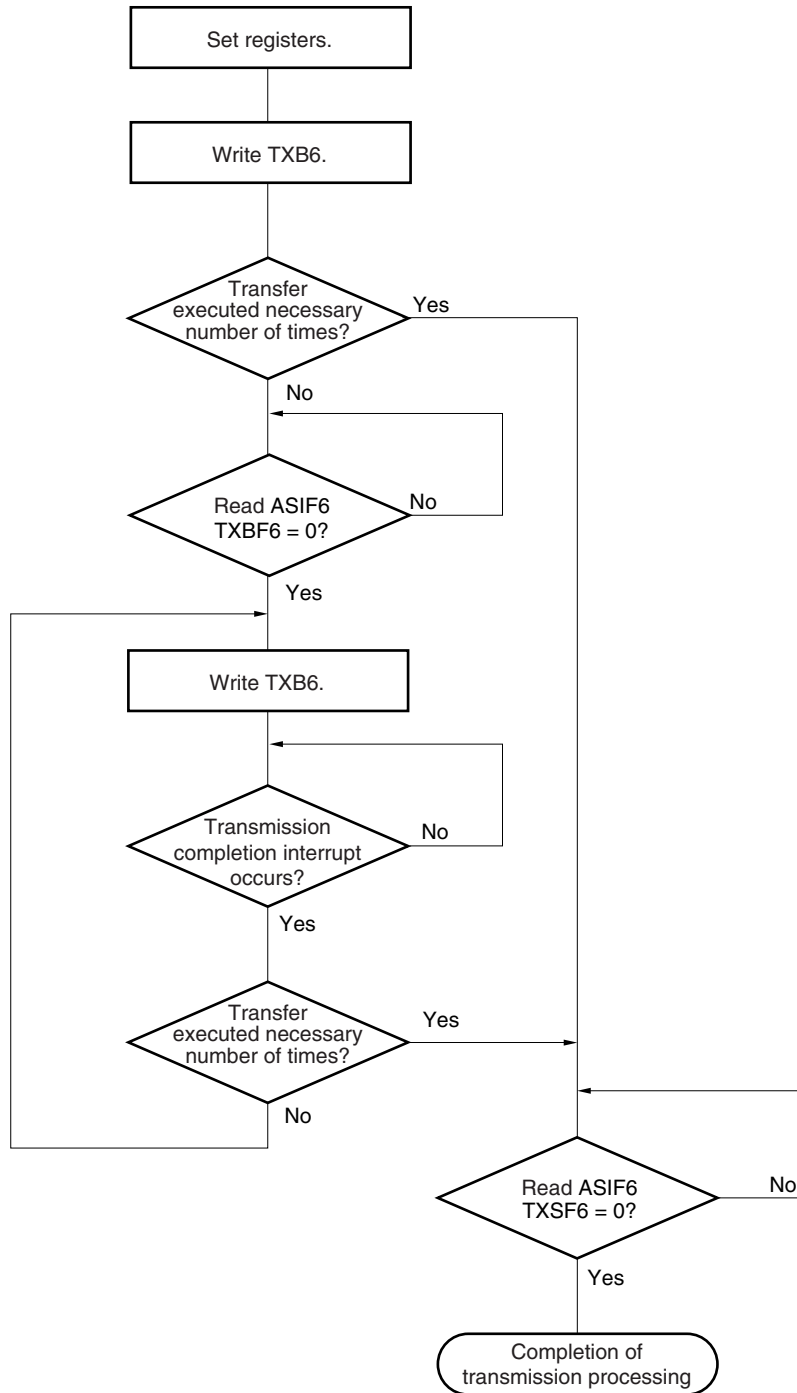
The communication status can be checked using the TXSF6 flag.

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 10-13 shows an example of the continuous transmission processing flow.

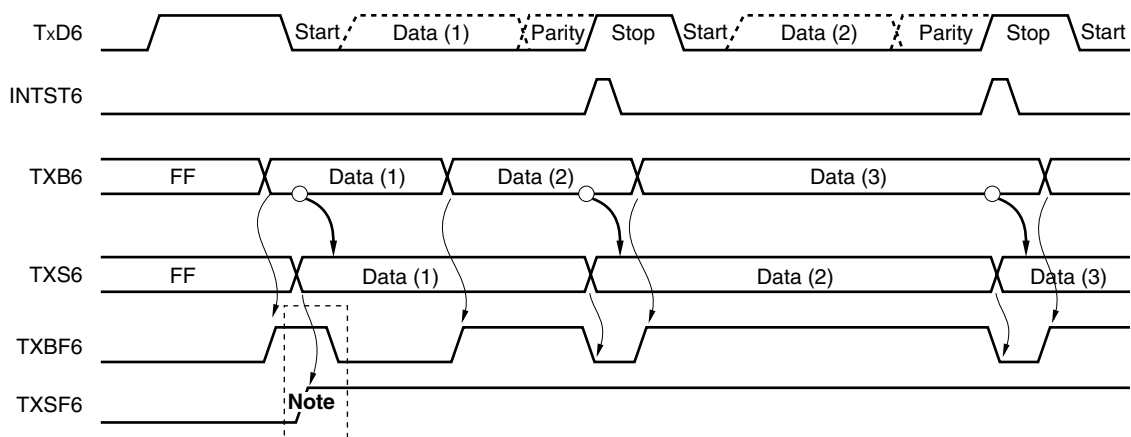
**Figure 10-13. Example of Continuous Transmission Processing Flow**



**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 10-14 shows the timing of starting continuous transmission, and Figure 10-15 shows the timing of ending continuous transmission.

**Figure 10-14. Timing of Starting Continuous Transmission**

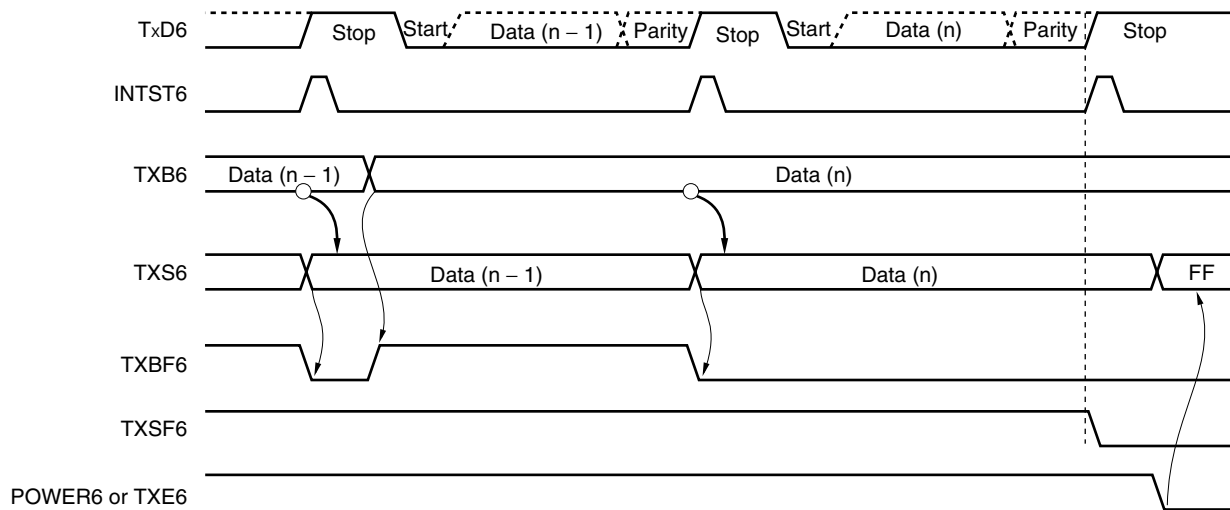


**Note** When ASIF6 is read, there is a period in which TXBF6 and TXSF6 = 1, 1. Therefore, judge whether writing is enabled using only the TXBF6 bit.

**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 10-15. Timing of Ending Continuous Transmission



<b>Remark</b>	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**

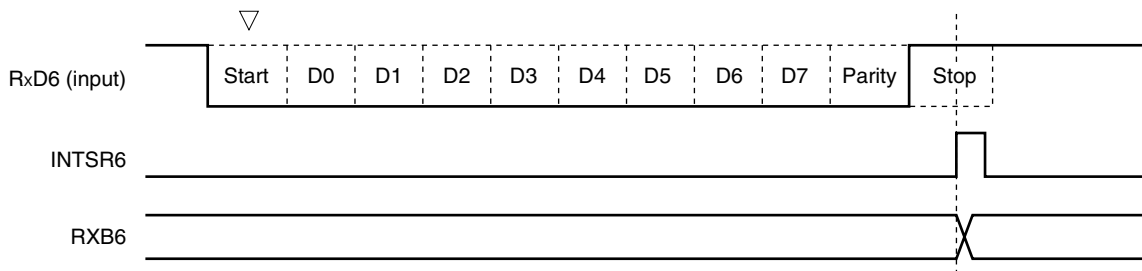
Reception is enabled and the RxD6 pin input is sampled when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1 and then bit 5 (RXE6) of ASIM6 is set to 1.

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 10-16). 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 10-16. 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 10-3**).

The contents of ASIS6 are cleared to 0 when ASIS6 is read.

**Table 10-3. 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).

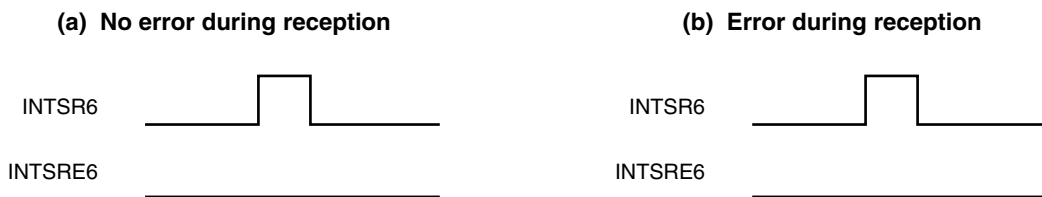
The reception error interrupt can be separated into reception completion interrupt (INTSR6) and error interrupt (INTSRE6) by clearing bit 0 (ISRM6) of asynchronous serial interface operation mode register 6 (ASIM6) to 0.

**Figure 10-17. 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)**





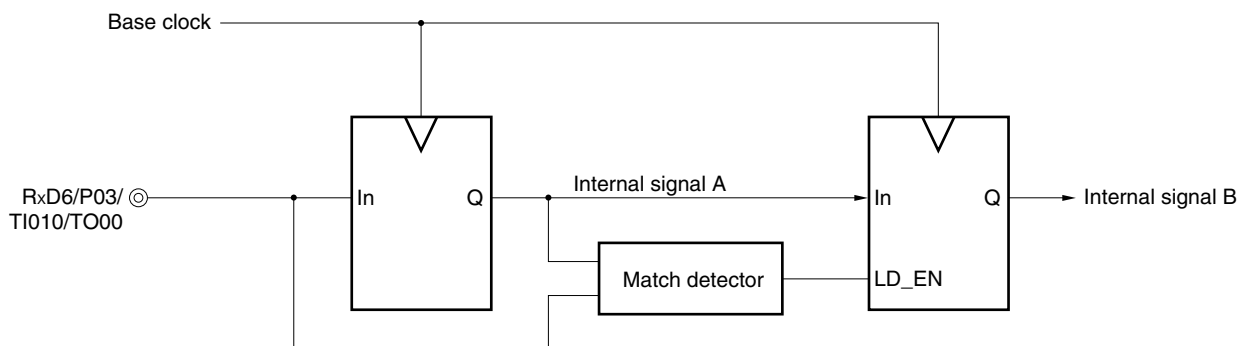
**(g) Noise filter of receive data**

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 10-18, the internal processing of the reception operation is delayed by two clocks from the external signal status.

**Figure 10-18. Noise Filter Circuit**

**10.4.3 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.

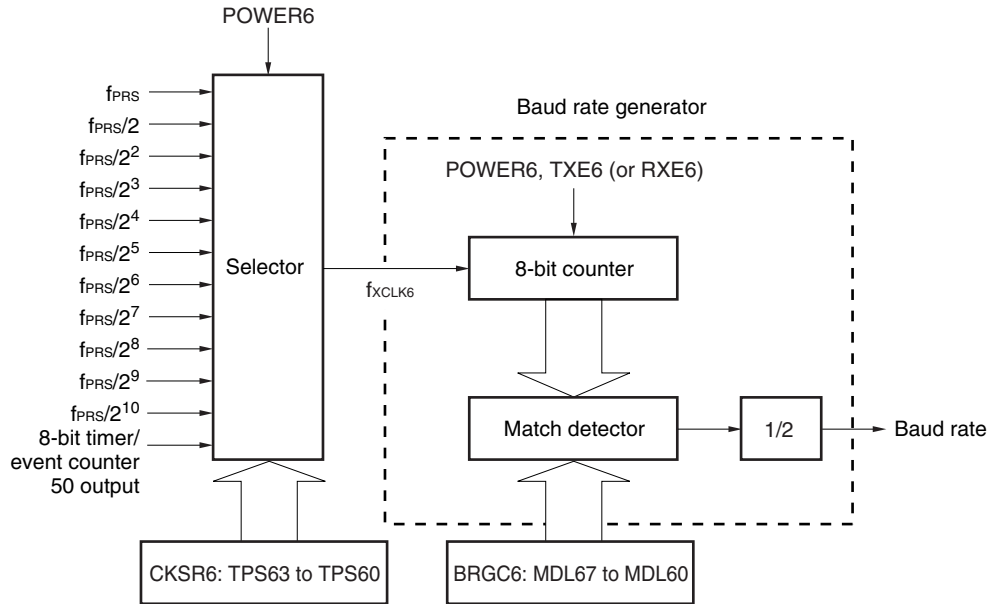
- **Reception counter**

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 10-19. 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.

## 10.4.4 Calculation of baud rate

## (1) Baud rate calculation expression

The baud rate can be calculated by the following expression.

- Baud rate =  $\frac{f_{\text{CLK6}}}{2 \times k}$  [bps]

$f_{\text{CLK6}}$ : 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 10-4. Set Value of TPS63 to TPS60

TPS63	TPS62	TPS61	TPS60	Base Clock ( $f_{\text{CLK6}}$ ) Selection		
				$f_{\text{PRS}} = 2 \text{ MHz}$	$f_{\text{PRS}} = 4 \text{ MHz}$	
0	0	0	0	$f_{\text{PRS}}$	2 MHz	4 MHz
0	0	0	1	$f_{\text{PRS}}/2$	1 MHz	2 MHz
0	0	1	0	$f_{\text{PRS}}/2^2$	500 kHz	1 MHz
0	0	1	1	$f_{\text{PRS}}/2^3$	250 kHz	500 kHz
0	1	0	0	$f_{\text{PRS}}/2^4$	125 kHz	250 kHz
0	1	0	1	$f_{\text{PRS}}/2^5$	62.5 kHz	125 kHz
0	1	1	0	$f_{\text{PRS}}/2^6$	31.25 kHz	62.5 kHz
0	1	1	1	$f_{\text{PRS}}/2^7$	15.625 kHz	31.25 kHz
1	0	0	0	$f_{\text{PRS}}/2^8$	7.813 kHz	15.625 kHz
1	0	0	1	$f_{\text{PRS}}/2^9$	3.906 kHz	7.813 kHz
1	0	1	0	$f_{\text{PRS}}/2^{10}$	1.953 kHz	3.906 kHz
1	0	1	1	TM50 output <sup>Note</sup>		
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.

- 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 = 4 MHz = 4,000,000 Hz

Set value of MDL67 to MDL60 bits of BRGC6 register = 00001101B (k = 13)

Target baud rate = 153600 bps

$$\begin{aligned} \text{Baud rate} &= 4 \text{ M} / (2 \times 13) \\ &= 4000000 / (2 \times 13) = 153846 \text{ [bps]} \end{aligned}$$

$$\begin{aligned} \text{Error} &= (153846/153600 - 1) \times 100 \\ &= 0.16 \text{ [%]} \end{aligned}$$

**(3) Example of setting baud rate**

**Table 10-5. Set Data of Baud Rate Generator**

Baud Rate [bps]	f <sub>PRS</sub> = 2.0 MHz				f <sub>PRS</sub> = 4.0 MHz			
	TPS63-TPS60	k	Calculated Value	ERR [%]	TPS63-TPS60	k	Calculated Value	ERR [%]
300	8H	13	301	0.16	8H	26	301	0.16
600	7H	13	601	0.16	7H	26	601	0.16
1200	6H	13	1202	0.16	6H	26	1202	0.16
2400	5H	13	2404	0.16	5H	26	2404	0.16
4800	4H	13	4808	0.16	4H	26	4808	0.16
9600	3H	13	9615	0.16	3H	26	9615	0.16
19200	2H	13	19231	0.16	2H	26	19231	0.16
24000	1H	21	23810	-0.79	1H	42	23810	-0.79
31250	1H	16	31250	0	1H	32	31250	0
38400	1H	13	38462	0.16	1H	26	38462	0.16
48000	0H	21	47619	-0.79	0H	42	47619	-0.79
76800	0H	13	76923	0.16	0H	26	76923	0.16
115200	0H	9	111111	-3.55	0H	17	117647	2.1
153600	-	-	-	-	0H	13	153846	0.16
312500	-	-	-	-	0H	-	-	-
625000	-	-	-	-	0H	-	-	-

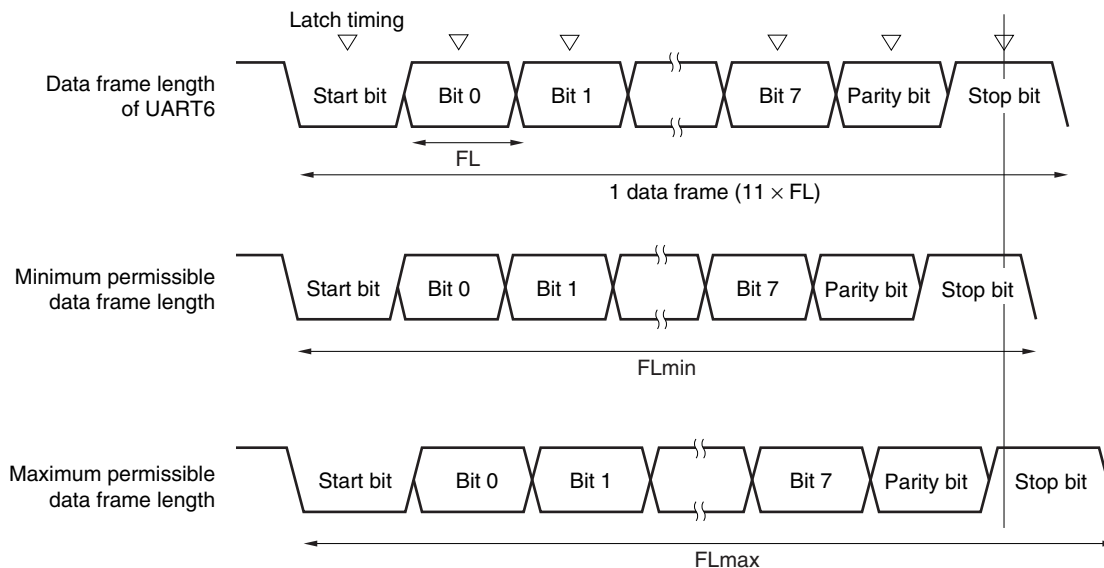
**Remark** TPS63 to TPS60: Bits 3 to 0 of clock selection register 6 (CKSR6) (setting of base clock (f<sub>CLK6</sub>))  
 k: Value set by MDL67 to MDL60 bits of baud rate generator control register 6 (BRGC6) (k = 4, 5, 6, ..., 255)  
 f<sub>PRS</sub>: Peripheral hardware clock frequency  
 ERR: Baud 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 10-20. Permissible Baud Rate Range During Reception**



As shown in Figure 10-20, 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 10-6. Maximum/Minimum Permissible Baud Rate Error**

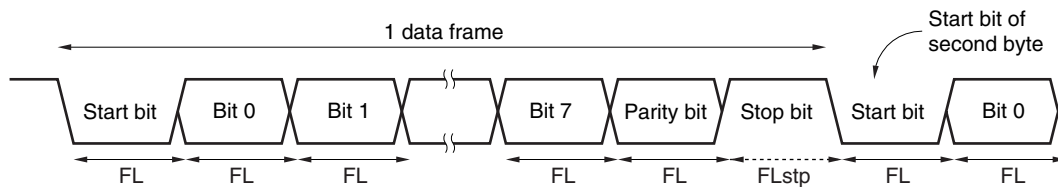
Division Ratio (k)	Maximum Permissible Baud Rate Error	Minimum Permissible Baud Rate Error
4	+2.33%	-2.44%
8	+3.53%	-3.61%
20	+4.26%	-4.31%
50	+4.56%	-4.58%
100	+4.66%	-4.67%
255	+4.72%	-4.73%

- 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 10-21. 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  $f_{XCLK6}$ , the following expression is satisfied.

$$FL_{stp} = FL + 2/f_{XCLK6}$$

Therefore, the data frame length during continuous transmission is:

$$\text{Data frame length} = 11 \times FL + 2/f_{XCLK6}$$

## CHAPTER 11 INTERRUPT FUNCTIONS

### 11.1 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 (PR0L, PR0H, PR1L).

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 11-1**.

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.

- External: 8, Internal: 10

#### (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.

### 11.2 Interrupt Sources and Configuration

The  $\mu$ PD179F11x, 179F12x microcontroller products have a total of 19 interrupt sources, including maskable interrupts and software interrupts. In addition, they also have up to four reset sources (see **Table 11-1**).



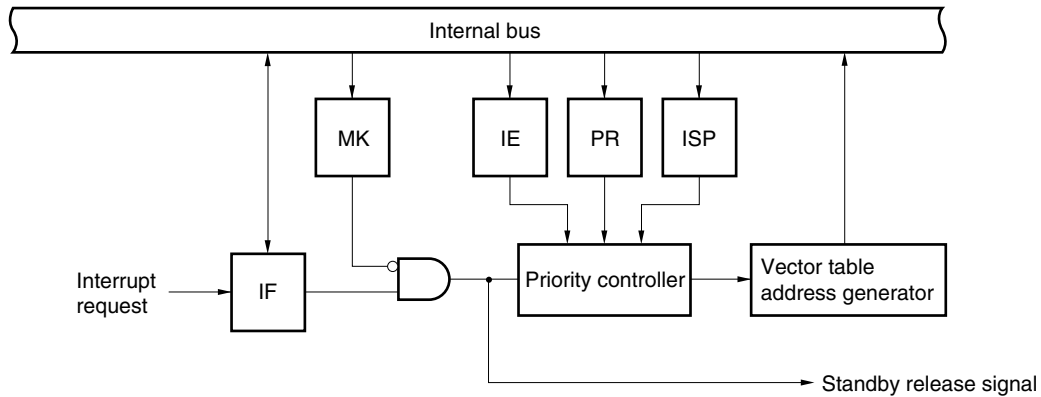
Table 11-1. Interrupt Source List (1/2)

Interrupt Type	Default Priority <sup>Note 1</sup>	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type <sup>Note 2</sup>
		Name	Trigger			
Maskable	0	INTLVI	Low-voltage detection <sup>Note 3</sup>	Internal	0004H	(A)
	1	INTP0	Pin input edge detection	External	0006H	(B)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTP4			000EH	
	6	INTP5			0010H	
	7	INTSRE6	UART6 reception error generation	Internal	0012H	(A)
	8	INTSR6	End of UART6 reception		0014H	
	9	INTST6	End of UART6 transmission		0016H	
	10	INTTMH1	Match between TMH1 and CMP01 (when compare register is specified)		001AH	
	11	INTTMH0	Match between TMH0 and CMP00 (when compare register is specified)		001CH	
	12	INTTM50	Match between TM50 and CR50 (when compare register is specified)		001EH	
	13	INTTM000	Match between TM00 and CR000 (when compare register is specified), TI010 pin valid edge detection (when capture register is specified)		0020H	
	14	INTTM010	Match between TM00 and CR010 (when compare register is specified), TI000 pin valid edge detection (when capture register is specified)		0022H	
	15	INTTM51 <sup>Note 4</sup>	Match between TM51 and CR51 (when compare register is specified)		002AH	
	16	INTKR0	Key interrupt detection		External	
17	INTKR1	002EH				
Software	–	BRK	BRK instruction execution	–	003EH	(D)
Reset	–	RESET	Reset input	–	0000H	–
		POC	Power-on-clear			
		LVI	Low-voltage detection <sup>Note 5</sup>			
		WDT	WDT overflow			

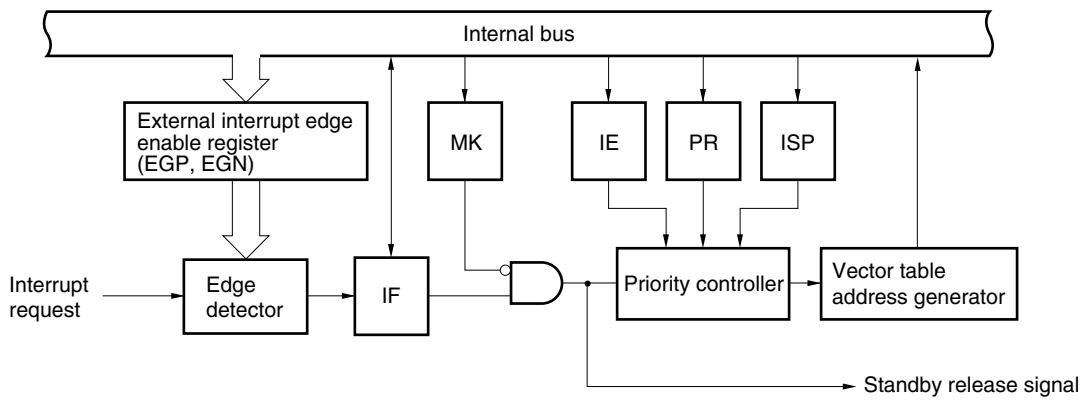
- Notes**
1. The default priority determines the sequence of processing vectored interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 17 indicates the lowest priority.
  2. Basic configuration types (A) to (D) correspond to (A) to (D) in Figure 11-1.
  3. When bit 1 (LVIMD) of the low-voltage detection register (LVIM) is cleared to 0.
  4. 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 8-16 Carrier Generator Mode Operation Timing**).
  5. When bit 1 (LVIMD) of the low-voltage detection register (LVIM) is set to 1.

Figure 11-1. Basic Configuration of Interrupt Function (1/2)

(A) Internal maskable interrupt



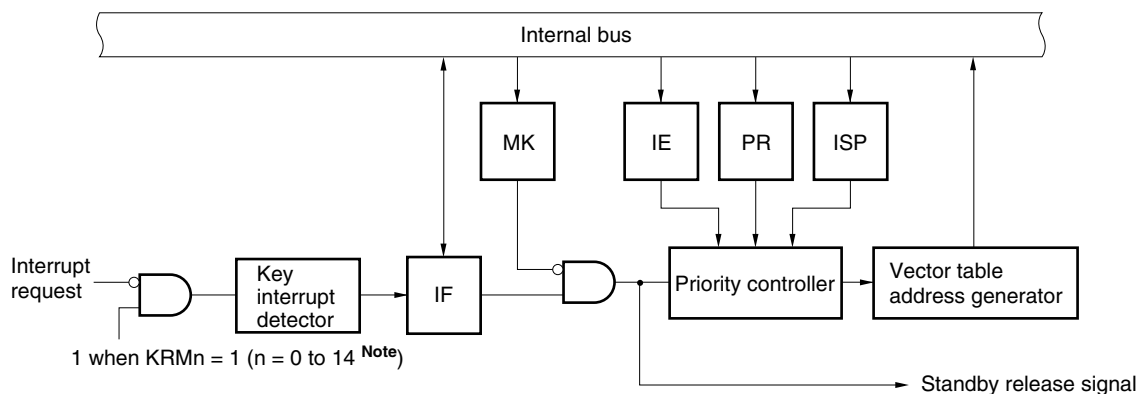
(B) External maskable interrupt (INTP0 to INTP5)



- IF: Interrupt request flag
- IE: Interrupt enable flag
- ISP: In-service priority flag
- MK: Interrupt mask flag
- PR: Priority specification flag

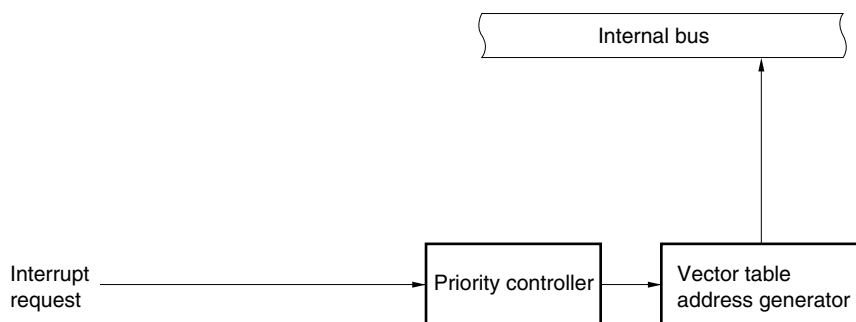
Figure 11-1. Basic Configuration of Interrupt Function (2/2)

(C) External maskable interrupt (INTKR)



**Note** KRM9 to KRM14 are available only in the 38-pin products.

(D) Software interrupt



- IF: Interrupt request flag
- IE: Interrupt enable flag
- ISP: In-service priority flag
- MK: Interrupt mask flag
- PR: Priority specification flag
- KRM: Key return mode register

### 11.3 Registers Controlling Interrupt Functions

The following 6 types of registers are used to control the interrupt functions.

- Interrupt request flag register (IF0L, IF0H, IF1L)
- Interrupt mask flag register (MK0L, MK0H, MK1L)
- Priority specification flag register (PR0L, PR0H, PR1L)
- External interrupt rising edge enable register (EGP)
- External interrupt falling edge enable register (EGN)
- Program status word (PSW)

Table 11-2 shows a list of interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

**Table 11-2. Flags Corresponding to Interrupt Request Sources**

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	
INTTMH1	TMIFH1		TMMKH1		TMPRH1	
INTTMH0	TMIFH0		TMMKH0		TMPRH0	
INTTM50	TMIF50		TMMK50		TMPR50	
INTTM000	TMIF000		TMMK000		TMPR000	
INTTM010	TMIF010		TMMK010		TMPR010	
INTTM51 <sup>Note</sup>	TMIF51		TMMK51		TMPR51	
INTKR0	KRIF0	IF1L	KRMK0	MK1L	KRPR0	PR1L
INTKR1	KRIF1		KRMK1		KRPR1	

**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 8-16 Carrier Generator Mode Operation Timing**).

**(1) Interrupt request flag registers (IF0L, IF0H, IF1L)**

The interrupt request flags are set to 1 when the corresponding interrupt request is generated or an instruction is executed. They are cleared to 0 when an instruction is executed upon acknowledgment of an interrupt request or upon reset signal generation.

When an interrupt is acknowledged, the interrupt request flag is automatically cleared and then the interrupt routine is entered.

IF0L, IF0H, and IF1L are set by a 1-bit or 8-bit memory manipulation instruction. When IF0L and IF0H are combined to form 16-bit registers IF0, they are set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to 00H.

**Figure 11-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L)**

Address: FFE0H After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF0L	SREIF6	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0	LVIIIF

Address: FFE1H After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	<3>	2	<1>	<0>
IF0H	TMIF010	TMIF000	TMIF50	TMIFH0	TMIFH1	0	STIF6	SRIF6

Address: FFE2H After reset: 00H R/W

Symbol	7	6	<5>	<4>	<3>	2	1	0
IF1L	0	0	KRIF1	KRIF0	TMIF51	0	0	0

XXIFX	Interrupt request flag
0	No interrupt request signal is generated
1	Interrupt request is generated, interrupt request status

- Cautions**
1. Be sure to clear bit 2 of IF0H, and bits 0 to 2, 6, 7 of IF1L to 0.
  2. When operating a timer after standby release, operate it once after clearing the interrupt request flag. An interrupt request flag may be set by noise.
  3. 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.

**(2) Interrupt mask flag registers (MK0L, MK0H, MK1L)**

The interrupt mask flags are used to enable/disable the corresponding maskable interrupt servicing. MK0L, MK0H, and MK1L are set by a 1-bit or 8-bit memory manipulation instruction. When MK0L and MK0H are combined to form 16-bit registers MK0, they are set by a 16-bit memory manipulation instruction. Reset signal generation sets these registers to FFH.

**Figure 11-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L)**

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	1	STMK6	SRMK6

Address: FFE6H After reset: FFH R/W

Symbol	7	6	<5>	<4>	<3>	2	1	0
MK1L	1	1	KRMK1	KRMK0	TMMK51	1	1	1

XXMKX	Interrupt servicing control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

**Caution** Be sure to set bit 2 of MK0H, and bits 0 to 2, 6, 7 of MK1L to 1.

**(3) Priority specification flag registers (PR0L, PR0H, PR1L)**

The priority specification flag registers are used to set the corresponding maskable interrupt priority order. PR0L, PR0H, and PR1L are set by a 1-bit or 8-bit memory manipulation instruction. If PR0L and PR0H are combined to form 16-bit registers PR0, they are set by a 16-bit memory manipulation instruction. Reset signal generation sets these registers to FFH.

**Figure 11-4. Format of Priority Specification Flag Registers (PR0L, PR0H, PR1L)**

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	1	STPR6	SRPR6

Address: FFEAH After reset: FFH R/W

Symbol	7	6	<5>	<4>	<3>	2	1	0
PR1L	1	1	KRPR1	KRPR0	TMPR51	1	1	1

XXPRX	Priority level selection
0	High priority level
1	Low priority level

**Caution** Be sure to set bit 2 of PR0H, and bits 0 to 2, 6, 7 of PR1L to 1.



**(4) External interrupt rising edge enable register (EGP), external interrupt falling edge enable register (EGN)**

These registers specify the valid edge for INTP0 to INTP5.

EGP and EGN are set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to 00H.

**Figure 11-5. 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

EGPn	EGNn	INTPn pin valid edge selection (n = 0 to 5)
0	0	Edge detection disabled
0	1	Falling edge
1	0	Rising edge
1	1	Both rising and falling edges

**Caution** Be sure to clear bits 6 and 7 of EGP and EGN to 0.

Table 11-3 shows the ports corresponding to EGPn and EGNn.

**Table 11-3. Ports Corresponding to EGPn and EGNn**

Detection Enable Register		Edge Detection Port	Interrupt Request Signal
EGP0	EGN0	P120	INTP0
EGP1	EGN1	P06	INTP1
EGP2	EGN2	P05	INTP2
EGP3	EGN3	P04	INTP3
EGP4	EGN4	P27	INTP4
EGP5	EGN5	RESET/P123	INTP5

**Caution** Select the port mode by clearing EGPn and EGNn to 0 because an edge may be detected when the external interrupt function is switched to the port function.

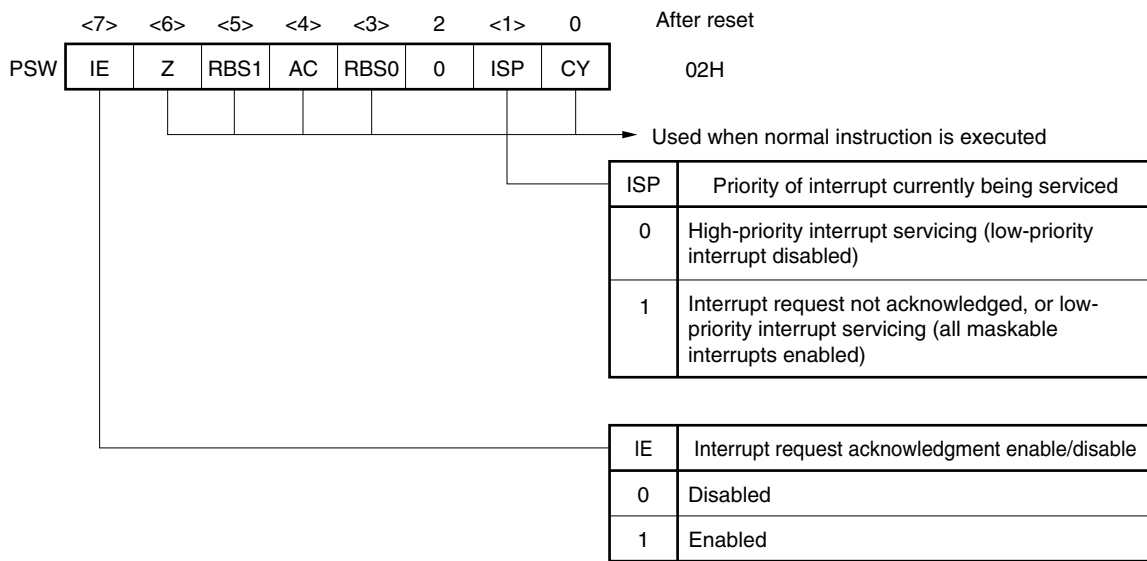
**Remark** n = 0 to 5

**(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 11-6. Format of Program Status Word**



## 11.4 Interrupt Servicing Operations

### 11.4.1 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 11-4 below.

For the interrupt request acknowledgment timing, see **Figures 11-8** and **11-9**.

**Table 11-4. Time from Generation of Maskable Interrupt Until Servicing**

	Minimum Time	Maximum Time <sup>Note</sup>
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.

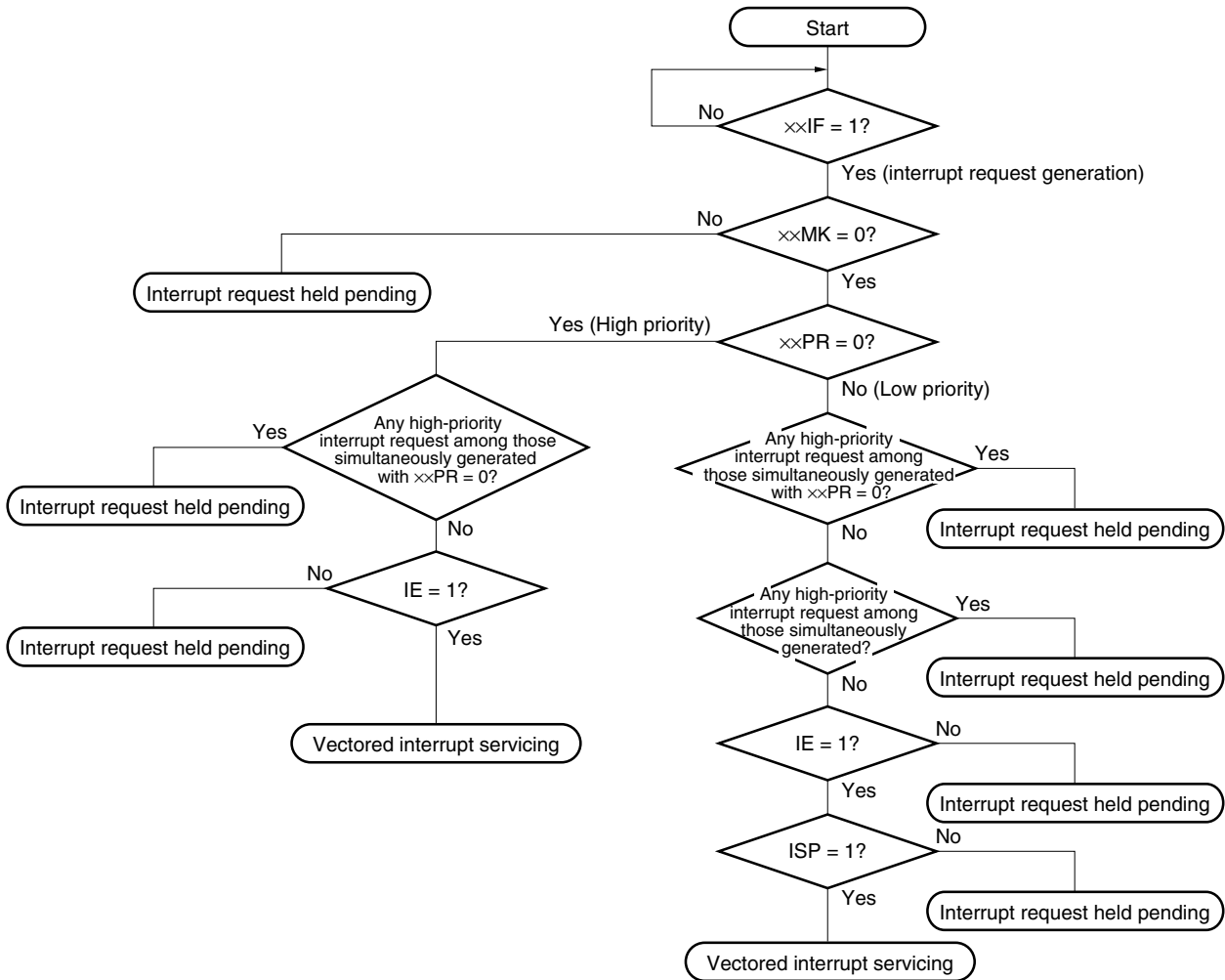
An interrupt request that is held pending is acknowledged when it becomes acknowledgeable.

Figure 11-7 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 11-7. Interrupt Request Acknowledgment Processing Algorithm



××IF: Interrupt request flag

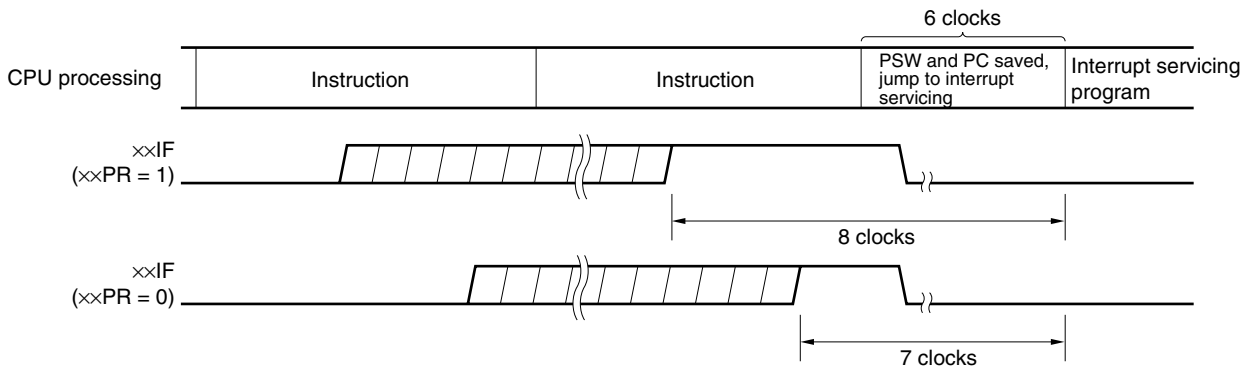
××MK: Interrupt mask flag

××PR: Priority specification flag

IE: Flag that controls acknowledgment of maskable interrupt request (1 = Enable, 0 = Disable)

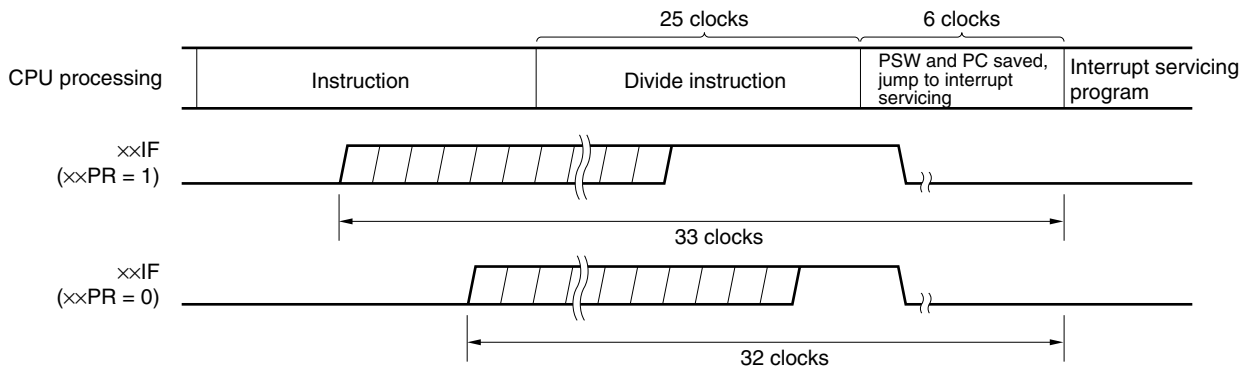
ISP: Flag that indicates the priority level of the interrupt currently being serviced (0 = high-priority interrupt servicing, 1 = No interrupt request acknowledged, or low-priority interrupt servicing)

Figure 11-8. Interrupt Request Acknowledgment Timing (Minimum Time)



**Remark** 1 clock:  $1/f_{CPU}$  ( $f_{CPU}$ : CPU clock)

Figure 11-9. Interrupt Request Acknowledgment Timing (Maximum Time)



**Remark** 1 clock:  $1/f_{CPU}$  ( $f_{CPU}$ : CPU clock)

#### 11.4.2 Software interrupt request acknowledgment

A software interrupt request is acknowledged by BRK instruction execution. Software interrupts cannot be disabled.

If a software interrupt request is acknowledged, the contents are saved into the stacks in the order of the program status word (PSW), then program counter (PC), the IE flag is reset (0), and the contents of the vector table (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.

**11.4.3 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 11-5 shows relationship between interrupt requests enabled for multiple interrupt servicing and Figure 11-10 shows multiple interrupt servicing examples.

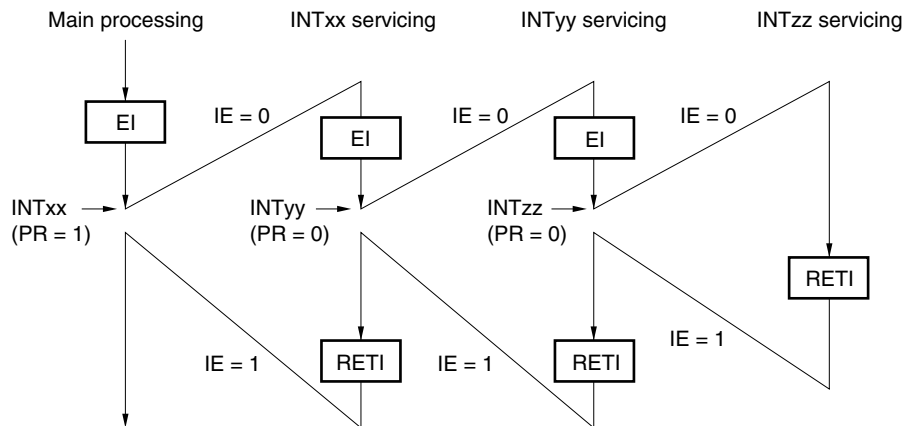
**Table 11-5. Relationship Between Interrupt Requests Enabled for Multiple Interrupt Servicing During Interrupt Servicing**

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**
1. ○: Multiple interrupt servicing enabled
  2. ×: Multiple interrupt servicing disabled
  3. 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.
  4. PR is a flag contained in PR0L, PR0H, and PR1L.  
 PR = 0: Higher priority level  
 PR = 1: Lower priority level

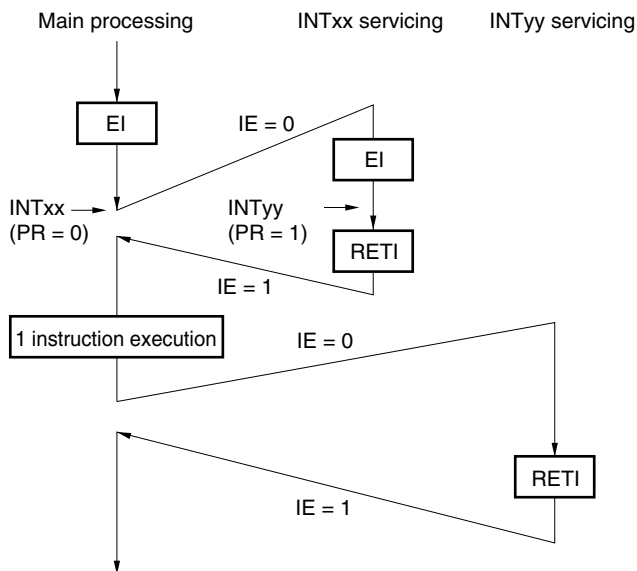
Figure 11-10. Examples of Multiple Interrupt Servicing (1/2)

**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.

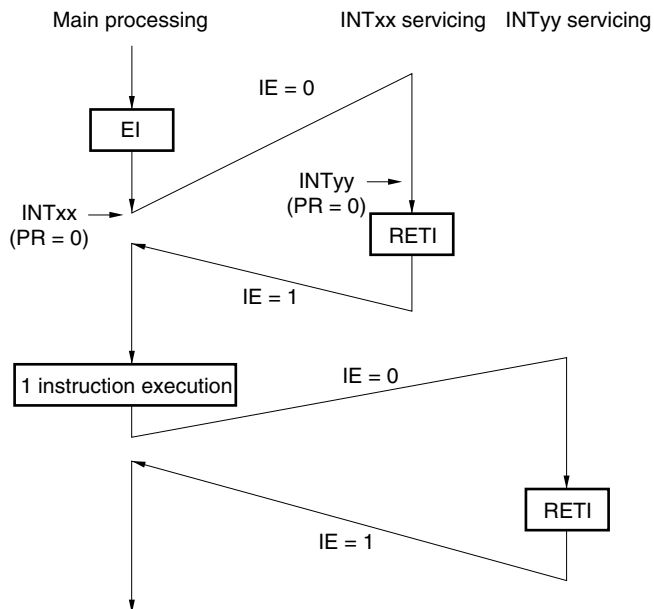
**Example 2. Multiple interrupt servicing does not occur due to priority control**



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 11-10. 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



#### 11.4.4 Interrupt request hold

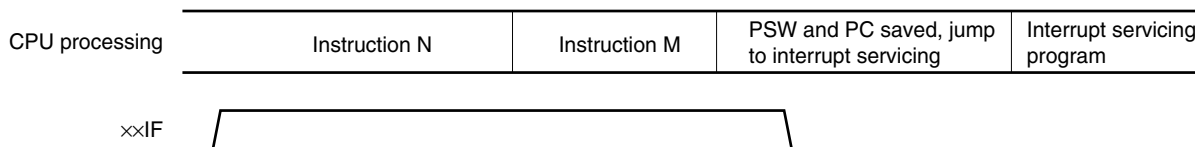
There are instructions where, even if an interrupt request is issued for them while another instruction is being executed, request acknowledgment is held pending until the end of execution of the next instruction. These instructions (interrupt request hold instructions) are listed below.

- 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, MK0L, MK0H, MK1L, PR0L, PR0H, and PR1L 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 11-11 shows the timing at which interrupt requests are held pending.

**Figure 11-11. 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 12 KEY INTERRUPT FUNCTION

### 12.1 Functions of Key Interrupt

A key interrupt (INTKR0 and INTKR1) can be generated by setting the key return mode register 0, 1 (KRML and KRMH) and inputting a falling edge to the key interrupt input pins <sup>Note</sup>.

When using the  $\overline{\text{RESET}}$ /P123/INTP5/KR8 pin for KR8, set bit 3 (RSTM) of the reset pin mode register (RSTMASK) to "1".

**Note** 30-pin products: KR0 to KR8  
 38-pin products: KR0 to KR14

**Table 12-1. Assignment of Key Interrupt Detection Pins**

Flag	Description
KRMLm	Controls KRm signal in 1-bit units.
KRMHm	Controls KRn signal in 1-bit units.

**Remark** m = 0 to 7, n = 8 to 14

### 12.2 Configuration of Key Interrupt

The key interrupt includes the following hardware.

**Table 12-2. Configuration of Key Interrupt**

Item	Configuration
Control register	Key return mode register 0 (KRML) Key return mode register 1 (KRMH)

**Figure 12-1. Block Diagram of Key Interrupt (1/2)**

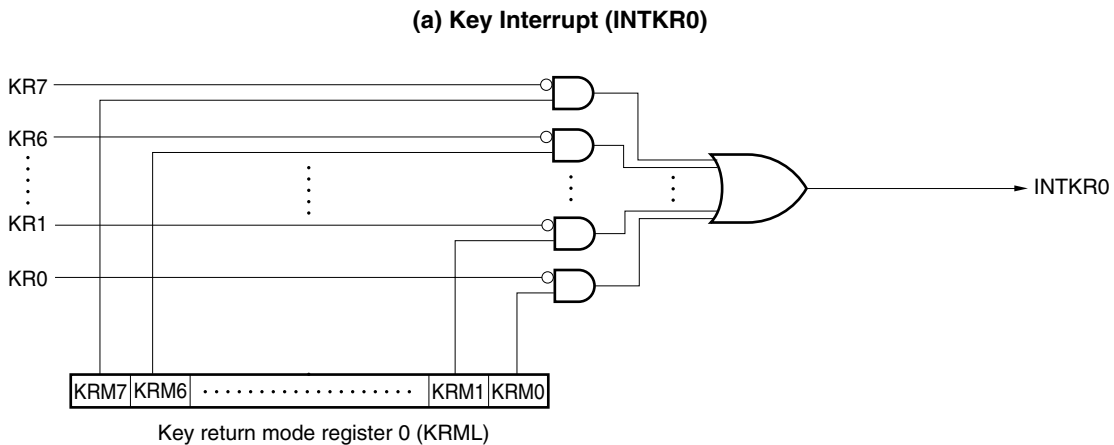
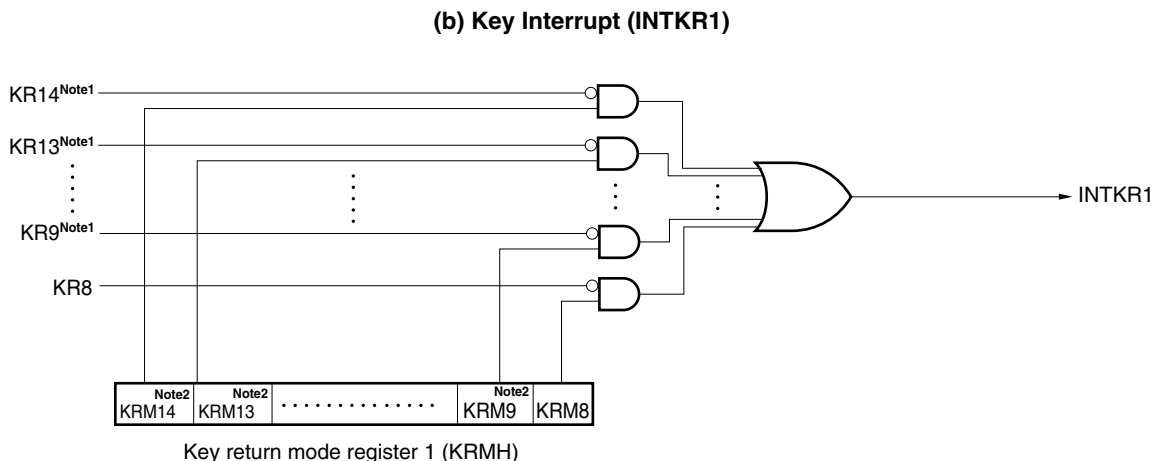


Figure 12-1. Block Diagram of Key Interrupt (2/2)



- Notes 1.** 38-pin products only  
**2.** These bits are fixed to “0” in the 30-pin products.

**12.3 Register Controlling Key Interrupt**

**(1) Key return mode register 0 (KRML)**

This register controls the KRM0 to KRM7 bits using the KR0 to KR7 signals, respectively. KRML is set by a 1-bit or 8-bit memory manipulation instruction. Reset signal generation sets KRML to 00H.

Figure 12-2. Format of Key Return Mode Register 0 (KRML)

Address: FF6EH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
KRML	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

PE6	Status flag indicating parity error
0	Does not detect key interrupt signal
1	Detects key interrupt signal

- Cautions 1.** If any of the KRM0 to KRM7 bits used is set to 1, set bits 0 to 7 of the corresponding pull-up resistor register 1 (PU1) to 1.  
**2.** If KRML is changed, the interrupt request flag may be set. Therefore, disable interrupts and then change the KRML register. Clear the interrupt request flag and enable interrupts.  
**3.** The bits not used in the key interrupt mode can be used as normal ports.

**(2) Key return mode register 1 (KRMH)**

This register controls the KRM8, and KRM9 to KRM14<sup>Note</sup> bits using the KR8, and KR9 to KR14<sup>Note</sup> signals, respectively.

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

Reset signal generation sets KRMH to 00H.

**Figure 12-3. Format of Key Return Mode Register 1 (KRMH)**

Address: FF6FH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
KRMH	0	KRM14 <sup>Note</sup>	KRM13 <sup>Note</sup>	KRM12 <sup>Note</sup>	KRM11 <sup>Note</sup>	KRM10 <sup>Note</sup>	KRM9 <sup>Note</sup>	KRM8

PE6	Status flag indicating parity error
0	Does not detect key interrupt signal
1	Detects key interrupt signal

**Note** 38-pin products only

- Cautions**
1. If any of the KRM8 to KRM14 bits used is set to 1, set bits 0 to 5 of the corresponding pull-up resistor register 3 (PU3) to 1, and bit 3 of the corresponding pull-up resistor register 12 (PU12) to 1.
  2. If KRMH is changed, the interrupt request flag may be set. Therefore, disable interrupts and then change the KRMH register. Clear the interrupt request flag and enable interrupts.
  3. The bits not used in the key interrupt mode can be used as normal ports.
  4. For the 30-pin products, be sure to set bits 0 to 6 of KRMH and PM3 to “0”.

## CHAPTER 13 STANDBY FUNCTION

### 13.1 Standby Function and Configuration

#### 13.1.1 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.

<R>

**Caution** When shifting to the STOP mode, be sure to stop the peripheral hardware operation operating with main system clock before executing STOP instruction.

#### 13.1.2 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.

<R> **Figure 13-1. Format of Oscillation Stabilization Time Counter Status Register (OSTC)**

Address: FFA3H After reset: 00H R

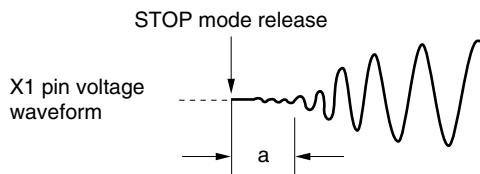
Symbol	7	6	5	4	3	2	1	0
OSTC	0	0	0	MOST11	MOST13	MOST14	MOST15	MOST16

MOST11	MOST13	MOST14	MOST15	MOST16	Oscillation stabilization time status			
					$f_x = 1 \text{ MHz}$	$f_x = 2 \text{ MHz}$	$f_x = 4 \text{ MHz}$	
0	0	0	0	0	Less than $2^{11}/f_x$	Less than 2.04 ms	Less than 1.02 ms	Less than 510 $\mu\text{s}$
1	0	0	0	0	$2^{11}/f_x \text{ min.}$	2.04 ms min	1.02 ms min	510 $\mu\text{s}$ min
1	1	0	0	0	$2^{13}/f_x \text{ min.}$	8.20 ms min	4.10 ms min	2.04 ms min
1	1	1	0	0	$2^{14}/f_x \text{ min.}$	16.38 ms min.	8.19 ms min.	4.10 ms min
1	1	1	1	0	$2^{15}/f_x \text{ min.}$	32.77 ms min.	16.38 ms min.	8.19 ms min.
1	1	1	1	1	$2^{16}/f_x \text{ min.}$	65.45 ms min.	32.77 ms min.	16.38 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 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  $\leq$  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.**
  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.

<R>

**Figure 13-2. Format of Oscillation Stabilization Time Select Register (OSTS)**

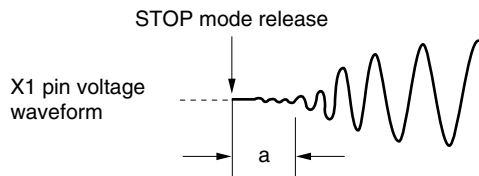
Address: FFA4H After reset: 05H R/W

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			
			$f_x = 1 \text{ MHz}$	$f_x = 2 \text{ MHz}$	$f_x = 4 \text{ MHz}$	
0	0	1	$2^{11}/f_x$	2.04 ms	1.02 ms	510 $\mu\text{s}$
0	1	0	$2^{13}/f_x$	8.19 ms	4.10 ms	2.04 ms
0	1	1	$2^{14}/f_x$	16.38 ms	8.19 ms	4.10 ms
1	0	0	$2^{15}/f_x$	32.77 ms	16.38 ms	8.19 ms
1	0	1	$2^{16}/f_x$	65.45 ms	32.77 ms	16.38 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

## 13.2 Standby Function Operation

### 13.2.1 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 13-1. 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 continued	
	$f_x$	Status before HALT mode was set is continued	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 continued		
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			
Watchdog timer		Operable. Clock supply to watchdog timer stops when "internal low-speed oscillator can be stopped by software" is set by option byte.		
Serial interface	UART6	Operable		
Power-on-clear function				
Low-voltage detection function				
External interrupt				

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



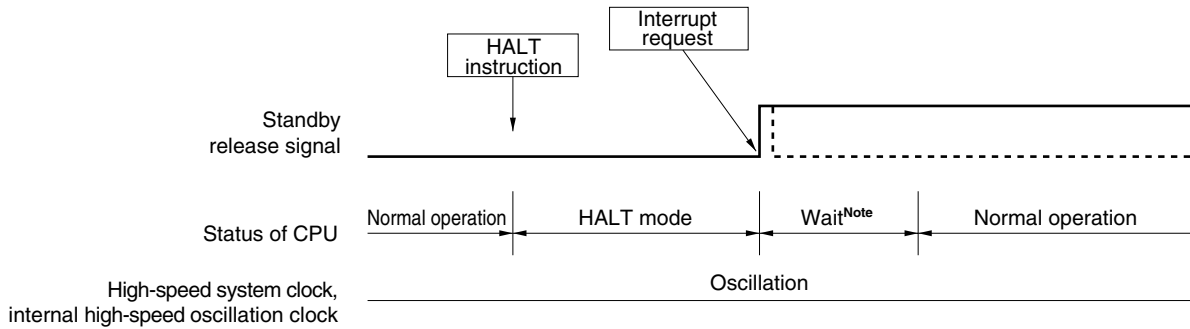
**(2) HALT mode release**

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 13-3. HALT Mode Release by Interrupt Request Generation**



**Note** The wait time is as follows:

- When vectored interrupt servicing is carried out: 8 or 9 clocks
- When vectored interrupt servicing is not carried out: 2 or 3 clocks

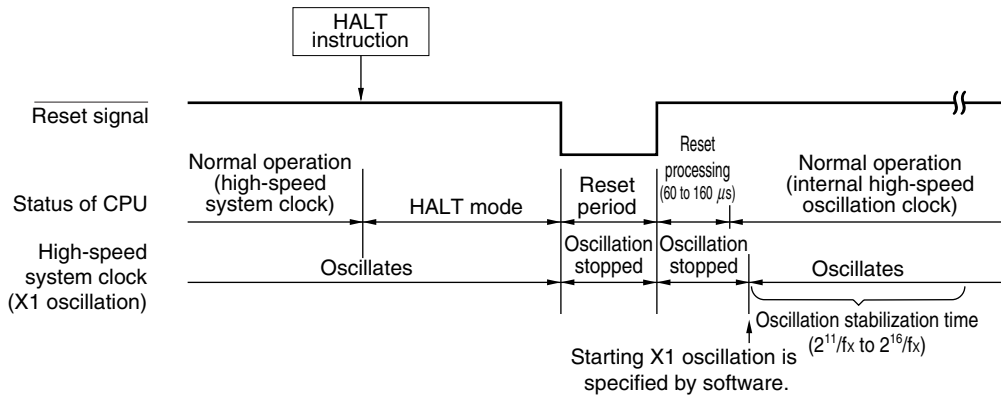
**Remark** The broken line indicates the case when the interrupt request which has released the standby mode is acknowledged.

**(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 13-4. HALT Mode Release by Reset (1/2)**

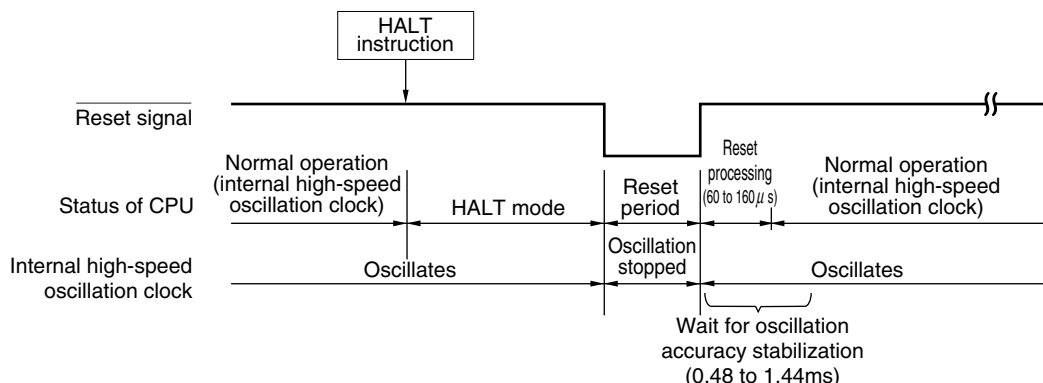
**(1) When high-speed system clock is used as CPU clock**



**Remark** fx: X1 clock oscillation frequency

Figure 13-4. HALT Mode Release by Reset (2/2)

(2) When internal high-speed oscillation clock is used as CPU clock



Remark fx: X1 clock oscillation frequency

Table 13-2. Operation in Response to Interrupt Request in HALT Mode

Release Source	MKxx	PRxx	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

13.2.2 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 13-3. 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 continued		
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 <sup>Note 1</sup>		Operation stopped		
8-bit timer/event counter	50 <sup>Note 1</sup>	Operable only when TI50 is selected as the count clock		
	51 <sup>Note 1</sup>	Operable only when TI51 is selected, or TMH1 carrier clock is selected as the count clock when 8-bit timer H1 operates in carrier generator mode		
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 $f_{RL}$ or $f_{RL}/2^7$ is selected as the count clock		
Watchdog timer		Operable. Clock supply to watchdog timer stops when “internal low-speed oscillator can be stopped by software” is set by option byte.		
Serial interface	UART6	Operable only when TM50 output is selected as the serial clock during 8-bit timer/event counter 50 operation		
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.

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

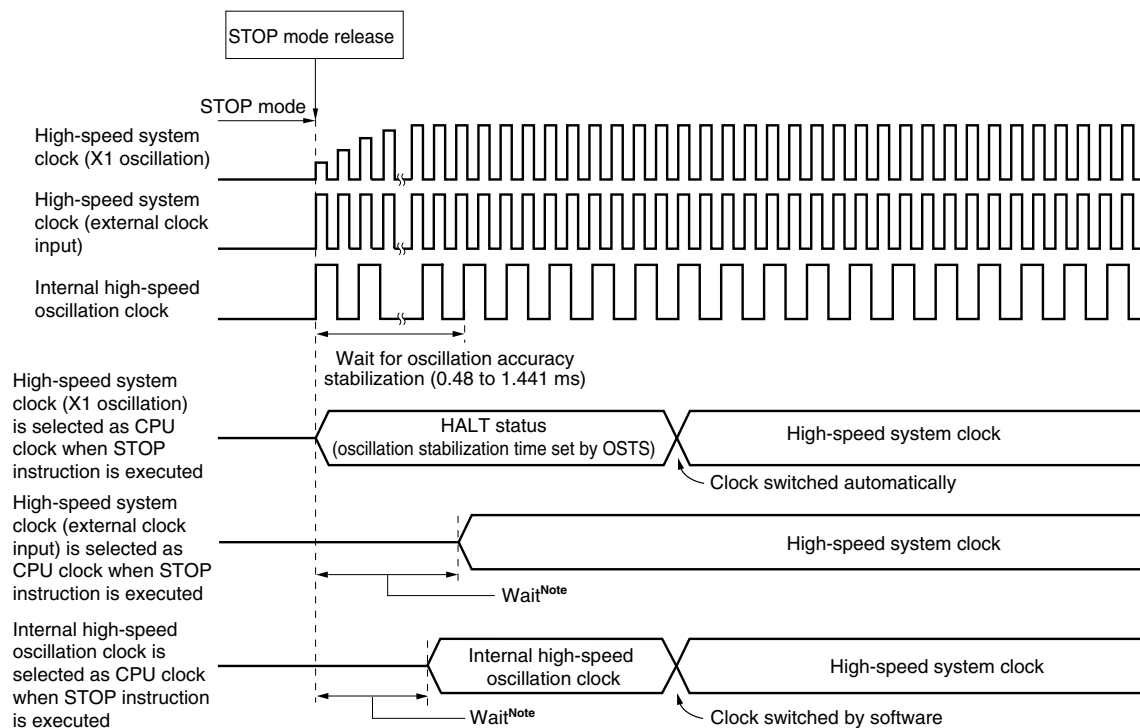
- Cautions**
1. To use the peripheral hardware that stops operation in the STOP mode, and the peripheral hardware for which the clock that stops oscillating in the STOP mode after the STOP mode is released, restart the peripheral hardware.
  2. Even if “internal low-speed oscillator can be stopped by 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), temporarily switch the CPU clock to the internal high-speed oscillation clock before the next execution of 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).

<R> **Cautions 4.** Be sure to confirm that the operation of the high-speed internal oscillator is stable (RSTS = 1) when performing a STOP instruction.

(2) STOP mode release

**Figure 13-5. 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: 8 or 9 clocks
- When vectored interrupt servicing is not carried out: 2 or 3 clocks

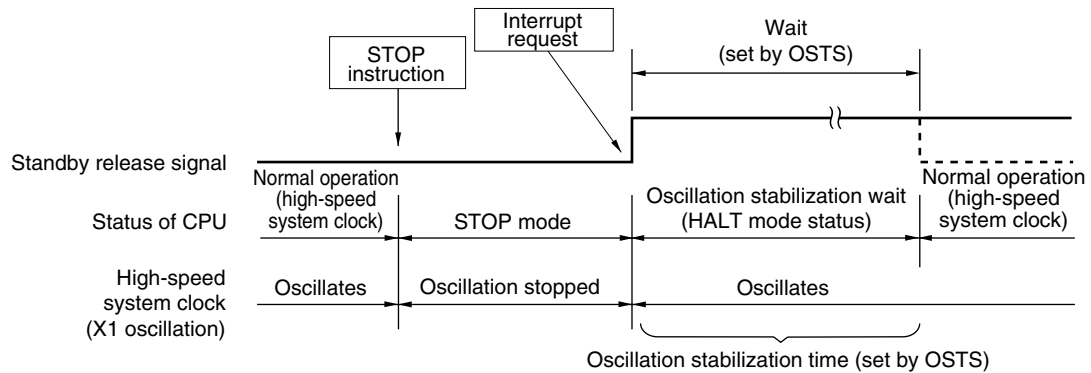
The STOP mode can be released by the following two sources.

**(a) Release by unmasked interrupt request**

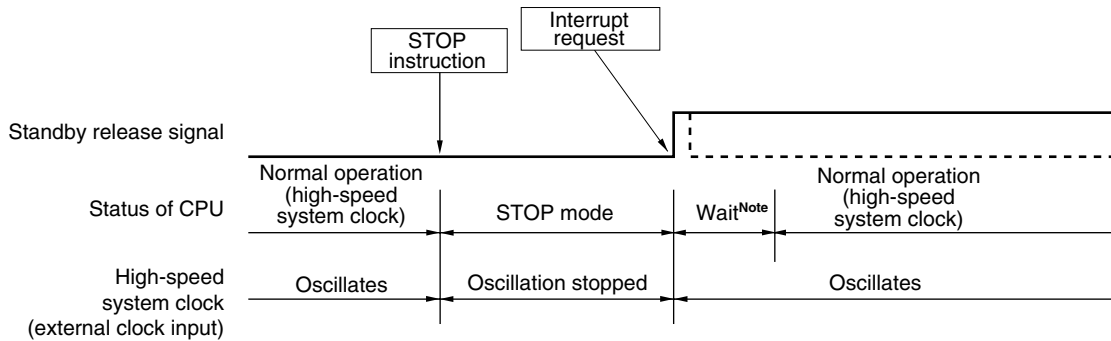
When an unmasked interrupt request is generated, the STOP mode is released. If interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

**Figure 13-6. 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**



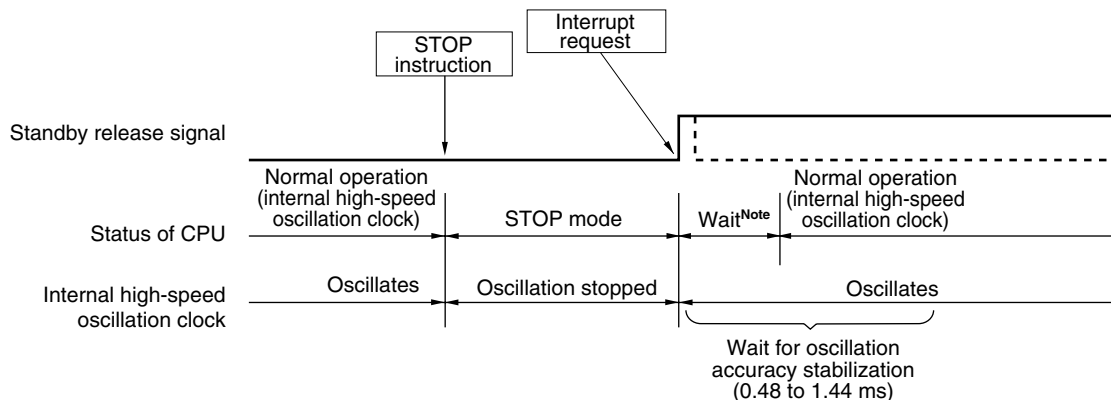
**Note** The wait time is as follows:

- When vectored interrupt servicing is carried out: 8 or 9 clocks
- When vectored interrupt servicing is not carried out: 2 or 3 clocks

**Remark** The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

Figure 13-6. 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: 8 or 9 clocks
- When vectored interrupt servicing is not carried out: 2 or 3 clocks

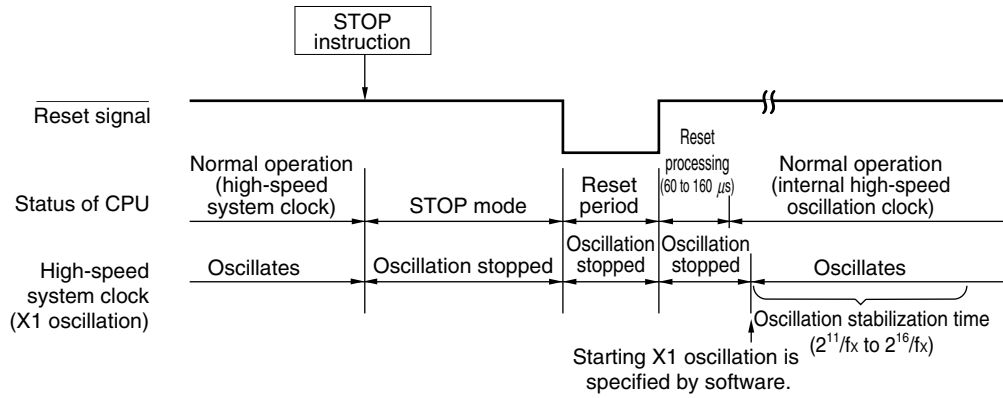
**Remark** The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

**(b) Release by reset signal generation**

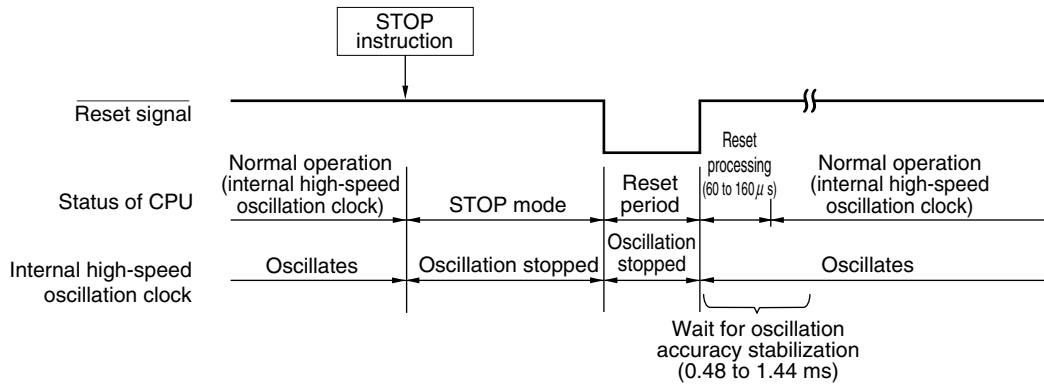
When the reset signal is generated, STOP mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

**Figure 13-7. STOP 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 13-4. Operation in Response to Interrupt Request in STOP Mode**

Release Source	MKxx	PRxx	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 14 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.

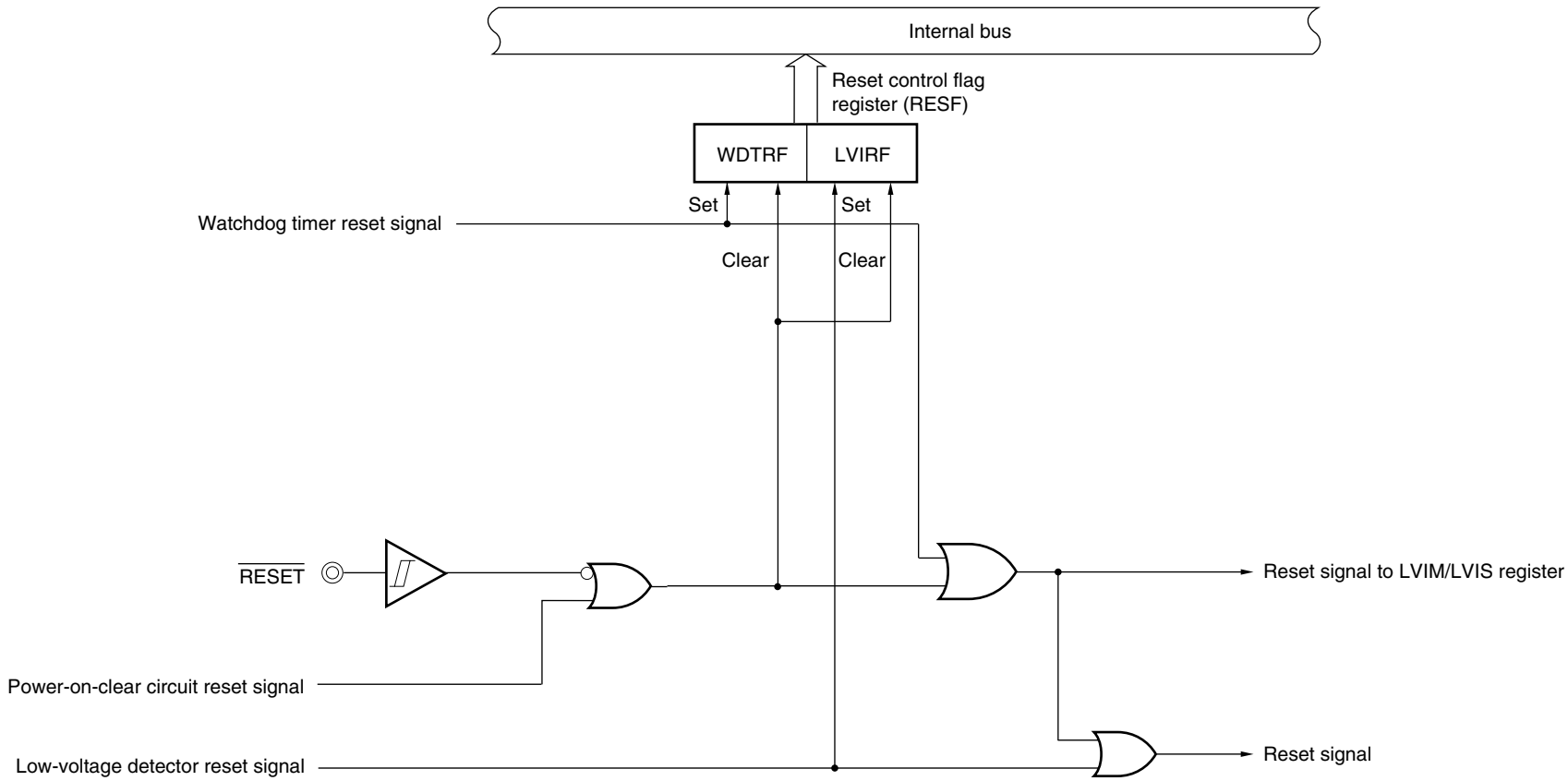
A reset is applied when a low level is input to the  $\overline{\text{RESET}}$  pin, the watchdog timer overflows, or by POC and LVI circuit voltage detection, and each item of hardware is set to the status shown in Tables 14-1 and 14-2. Each pin is high impedance during reset signal generation or during 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 14-2 to 14-4**) 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 15 POWER-ON-CLEAR CIRCUIT** and **CHAPTER 16 LOW-VOLTAGE DETECTOR**) after reset processing.

- Cautions**
1. For an external reset, input a low level for 10  $\mu\text{s}$  or more to the  $\overline{\text{RESET}}$  pin.
  2. During reset input, the X1 clock, internal high-speed oscillation clock, and internal low-speed oscillation clock stop oscillating. External main system clock input become 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 14-1. 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 14-2. Timing of Reset by RESET Input

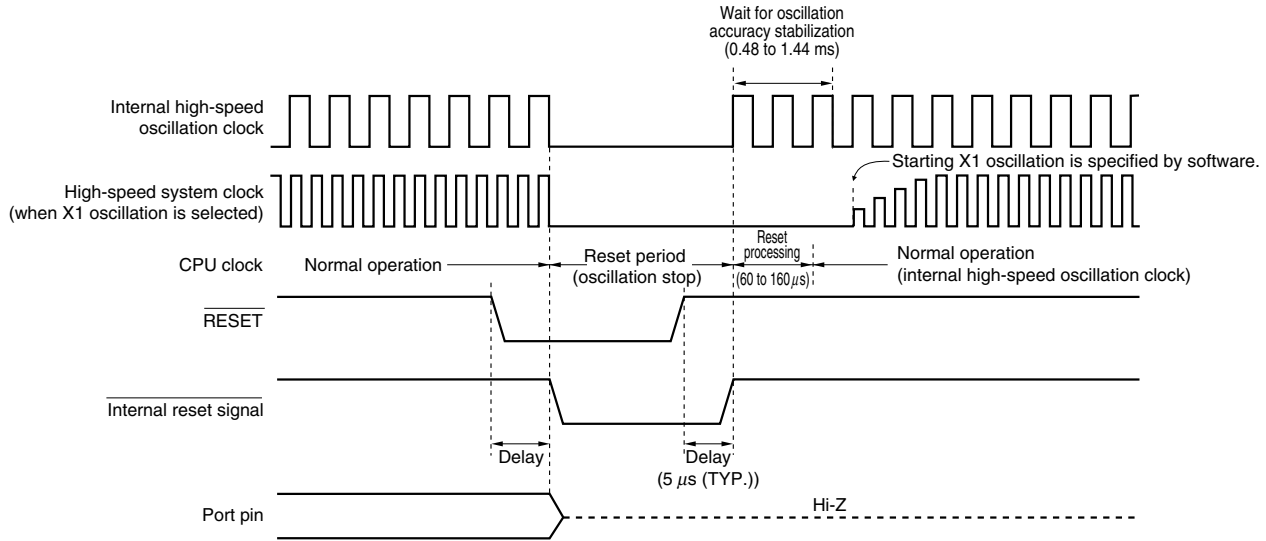
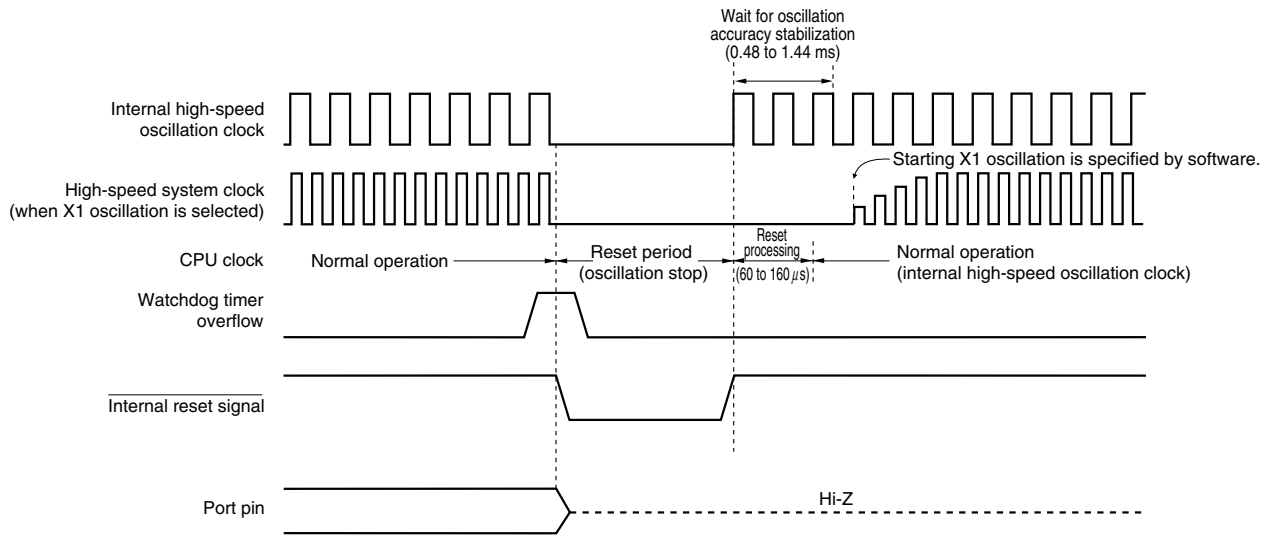
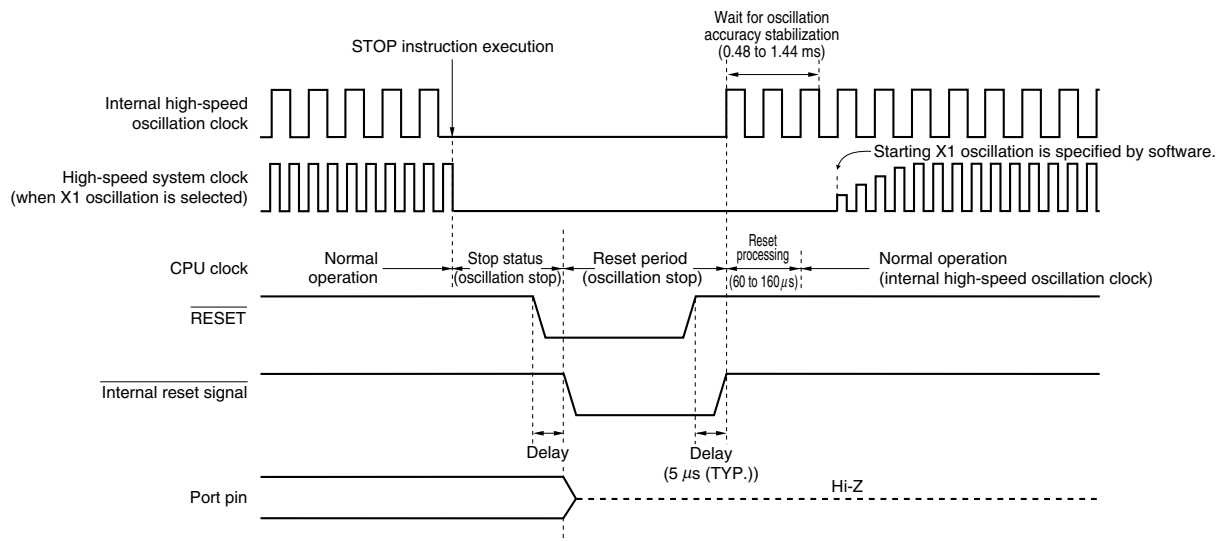


Figure 14-3. Timing of Reset Due to Watchdog Timer Overflow



**Caution** A watchdog timer internal reset resets the watchdog timer.

Figure 14-4. Timing of Reset in STOP Mode by RESET Input



**Remark** For the reset timing of the power-on-clear circuit and low-voltage detector, see **CHAPTER 15 POWER-ON-CLEAR CIRCUIT** and **CHAPTER 16 LOW-VOLTAGE DETECTOR**.

Table 14-1. Operation Statuses During Reset Period

Item	During Reset Period	
System clock	Clock supply to the CPU is stopped.	
Main system clock	f <sub>RH</sub>	Operation stopped
	f <sub>X</sub>	Operation stopped (pin is I/O port mode)
	f <sub>EXCLK</sub>	Clock input invalid (pin is I/O port mode)
f <sub>RL</sub>	Operation stopped	
CPU	Operation stopped	
Flash memory	Operation stopped	
RAM	Operation stopped	
Port (latch)	Operation stopped	
16-bit timer/event counter 00	Operation stopped	
8-bit timer/event counter	50	Operation stopped
	51	Operation stopped
8-bit timer	H0	Operation stopped
	H1	Operation stopped
Watchdog timer	Operation stopped	
Serial interface	UART6	Operation stopped
Power-on-clear function	Operable	
Low-voltage detection function	Operation stopped	
External interrupt	Operation stopped	

**Remark** f<sub>RH</sub>: Internal high-speed oscillation clock  
 f<sub>X</sub>: X1 oscillation clock  
 f<sub>EXCLK</sub>: External main system clock  
 f<sub>RL</sub>: Internal low-speed oscillation clock

Table 14-2. Hardware Statuses After Reset Acknowledgment (1/2)

Hardware		After Reset Acknowledgment <sup>Note 1</sup>
Program counter (PC)		The contents of the reset vector table (0000H, 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined <sup>Note 2</sup>
	General-purpose registers	Undefined <sup>Note 2</sup>
Port registers (P0 to P2, P3 <sup>Note 3</sup> , P12) (output latches)		00H
Port mode registers (PM0 to PM2, PM3 <sup>Note 3</sup> , PM12)		FFH
Pull-up resistor option registers (PU0 to PU2, PU3 <sup>Note 3</sup> , PU12)		00H
Port output mode resistors (POM0 to POM2, POM3 <sup>Note 3</sup> , POM12)		00H
Pull-up resistor option registers (PU0 to PU2, PU3 <sup>Note 3</sup> , PU12)		00H (08H for PU12)
FLMD0 Pull-up/Pull-down control register (FPCTL)		00H
FLMD0 Pull-up/Pull-down enable register (FPEN)		00H
Internal memory size switching register (IMS)		CFH <sup>Note 4</sup>
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

- 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. 38-pin products only.
  4. The initial  $\mu$  value of the internal memory size switching register after a reset release is fixed (IMS = CFH) in all the  $\mu$ PD179F11x, 179F12x microcontroller products, regardless of the internal memory capacity. Therefore, after a reset is released, be sure to set the following values for each product.

Flash Memory Version ( $\mu$ PD179F11x, 179F12x microcontrollers)	IMS
$\mu$ PD179F110	41H
$\mu$ PD179F111	42H
$\mu$ PD179F112, 179F122	04H
$\mu$ PD179F113, 179F123	C6H
$\mu$ PD179F114, 179F124	C8H

**Table 14-2. Hardware Statuses After Reset Acknowledgment (2/2)**

Hardware		Status After Reset Acknowledgment <sup>Note 1</sup>
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) <sup>Note 2</sup>	00H
Watchdog timer	Enable register (WDTE)	1AH/9AH <sup>Note 3</sup>
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
Key interrupt	Key return mode register 0, 1 (KRML, KRMH)	00H
Reset function	Reset control flag register (RESF)	00H <sup>Note 4</sup>
	Reset pin mode register (RSTMASK)	00H
Low-voltage detector	Low-voltage detection register (LVIM)	00H <sup>Note 4</sup>
	Low-voltage detection level selection register (LVIS)	00H <sup>Note 4</sup>
	RAM Data Retention control register (LVDET)	Undefined
Interrupt	Request flag registers 0L, 0H, 1L (IF0L, IF0H, IF1L)	00H
	Mask flag registers 0L, 0H, 1L (MK0L, MK0H, MK1L)	FFH
	Priority specification flag registers 0L, 0H, 1L (PR0L, PR0H, PR1L)	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.** 8-bit timer H1 only.

**3.** The reset value of WDTE is determined by the option byte setting.

**4.** These values vary depending on the reset source.

Reset Source		RESET Input	Reset by POC	Reset by WDT	Reset by LVI
RESF	WDTRF bit	Cleared (0)	Cleared (0)	Set (1)	Held
	LVIRF bit			Held	Set (1)
LVIM		Cleared (00H)	Cleared (00H)	Cleared (00H)	Held
LVIS					

### 14.1 Register for Confirming Reset Source

Many internal reset generation sources exist in the  $\mu$ PD179F11x, 179F12x microcontrollers. 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 14-5. Format of Reset Control Flag Register (RESF)**

Address: FFACH After reset: 00H<sup>Note</sup> 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 14-3.

**Table 14-3. RESF Status When Reset Request Is Generated**

Reset Source Flag	$\overline{\text{RESET}}$ Input	Reset by POC	Reset by WDT	Reset by LVI
WDTRF	Cleared (0)	Cleared (0)	Set (1)	Held
LVIRF			Held	Set (1)

## CHAPTER 15 POWER-ON-CLEAR CIRCUIT

### 15.1 Functions of Power-on-Clear Circuit

The power-on-clear circuit (POC) has the following functions.

- Generates internal reset signal at power on.  
The reset signal is released when the supply voltage ( $V_{DD}$ ) exceeds  $1.8\text{ V} \pm 0.1\text{ V}$ .
- Compares supply voltage ( $V_{DD}$ ) and detection voltage ( $V_{POC} = 1.8\text{ V} \pm 0.1\text{ V}$ ), 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.

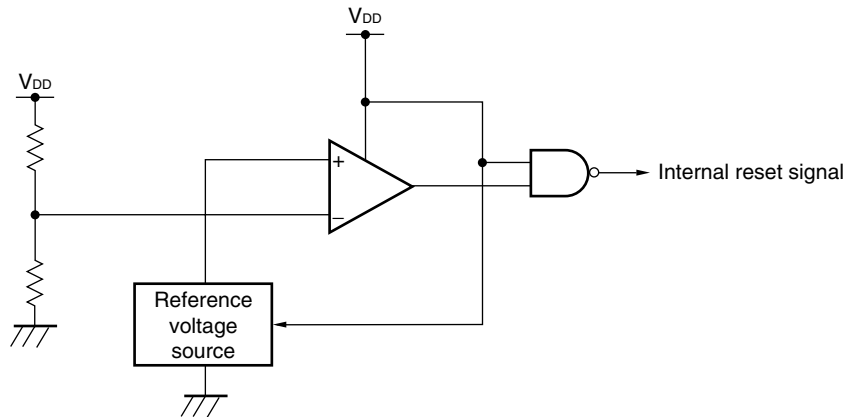
**Remark** The  $\mu$ PD179F11x, 179F12x microcontrollers incorporate multiple hardware functions that generate an internal reset signal. A flag that indicates the reset source is located in the reset control flag register (RESF) for when an internal reset signal is generated by the watchdog timer (WDT) 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 14 RESET FUNCTION**.

## 15.2 Configuration of Power-on-Clear Circuit

The block diagram of the power-on-clear circuit is shown in Figure 15-1.

**Figure 15-1. Block Diagram of Power-on-Clear Circuit**



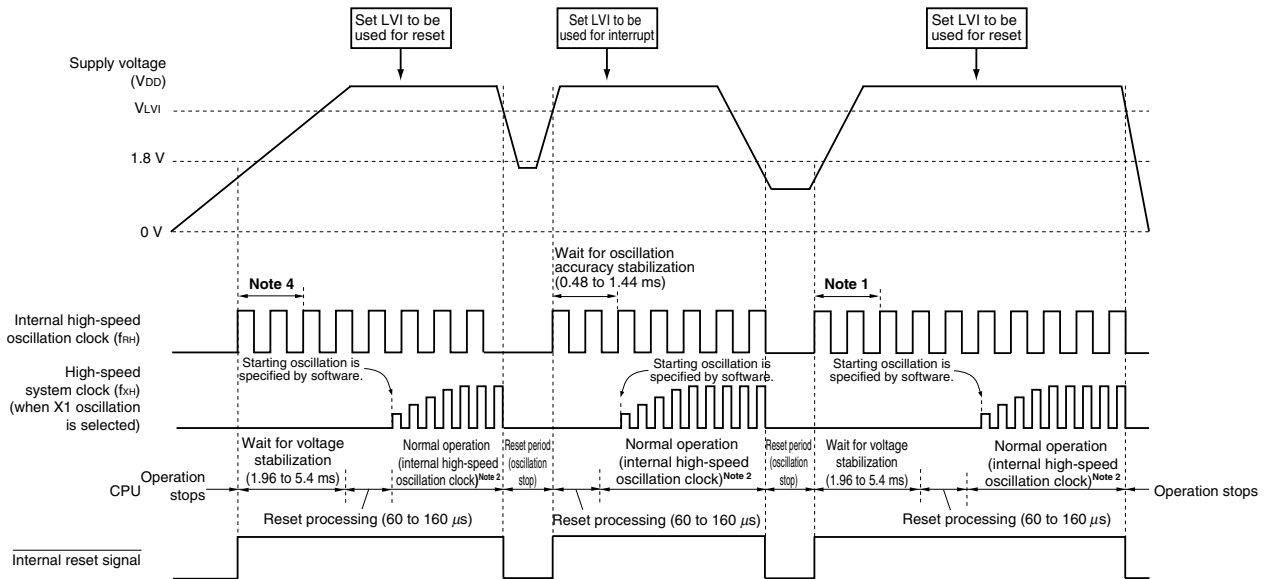
## 15.3 Operation of Power-on-Clear Circuit

- An internal reset signal is generated on power application. When the supply voltage ( $V_{DD}$ ) exceeds the detection voltage ( $V_{POC} = 1.8\text{ V} \pm 0.1\text{ V}$ ), the reset status is released.
- The supply voltage ( $V_{DD}$ ) and detection voltage ( $V_{POC} = 1.8\text{ V} \pm 0.1\text{ V}$ ) are compared. When  $V_{DD} < V_{POC}$ , the internal reset signal is generated. It is released when  $V_{DD} \geq V_{POC}$ .

The timing of generation of the internal reset signal by the power-on-clear circuit and low-voltage detector is shown below.



**Figure 15-2. Timing of Generation of Internal Reset Signal by Power-on-Clear Circuit and Low-Voltage Detector**



- Notes**
1. The oscillation accuracy stabilization time of the internal high-speed oscillation clock is included in the internal voltage stabilization time.
  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.

**Caution** Set the low-voltage detector by software after the reset status is released (see CHAPTER 16 LOW-VOLTAGE DETECTOR).

**Remark** V<sub>LVI</sub>: LVI detection voltage  
V<sub>POC</sub>: POC detection voltage

### 15.4 Cautions for Power-on-Clear Circuit

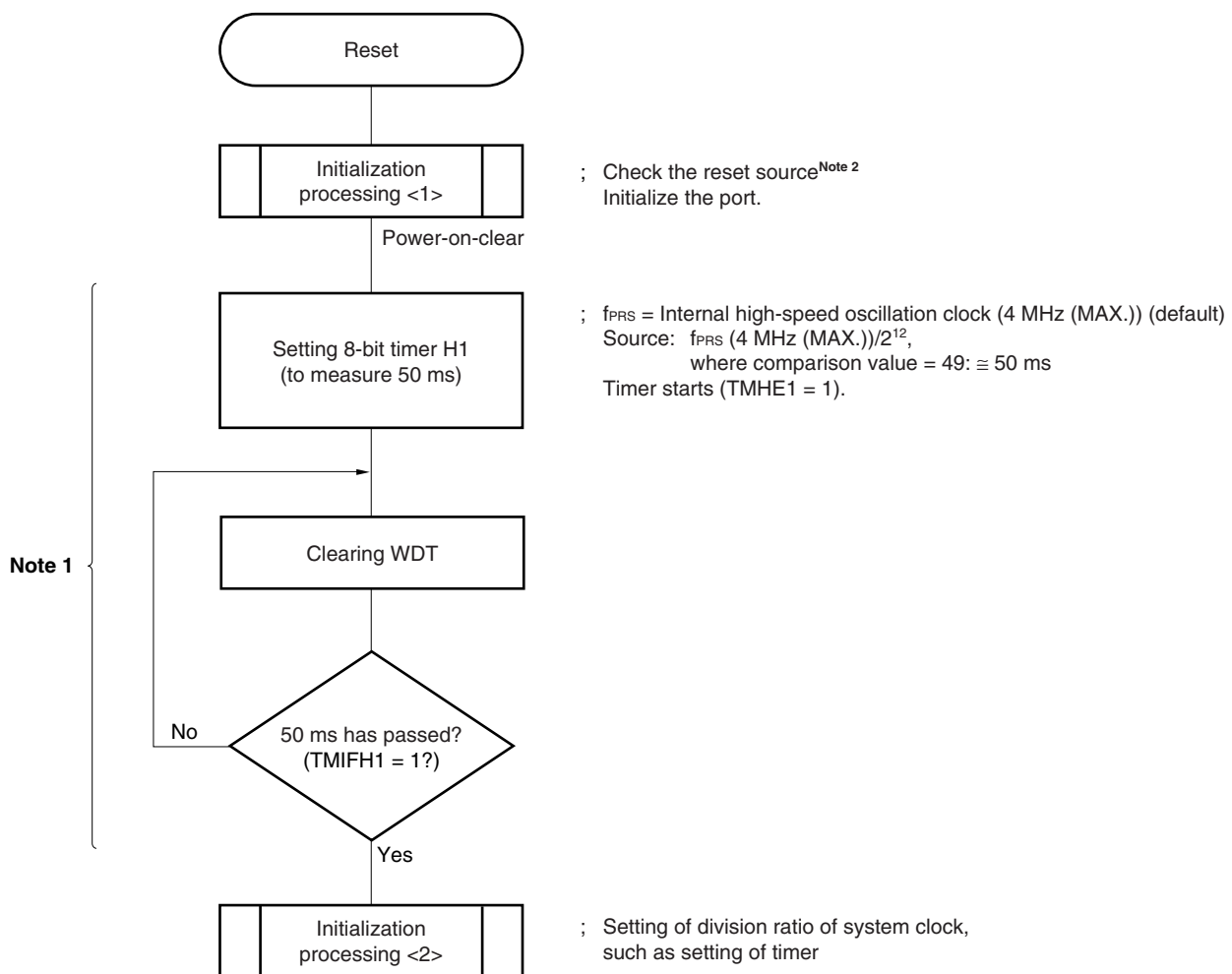
In a system where the supply voltage ( $V_{DD}$ ) fluctuates for a certain period in the vicinity of the POC detection voltage ( $V_{POC}$ ), the system may be repeatedly reset and released from the reset status. In this case, the time from release of reset to the start of the operation of the microcontroller 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 15-3. Example of Software Processing After Reset Release (1/2)**

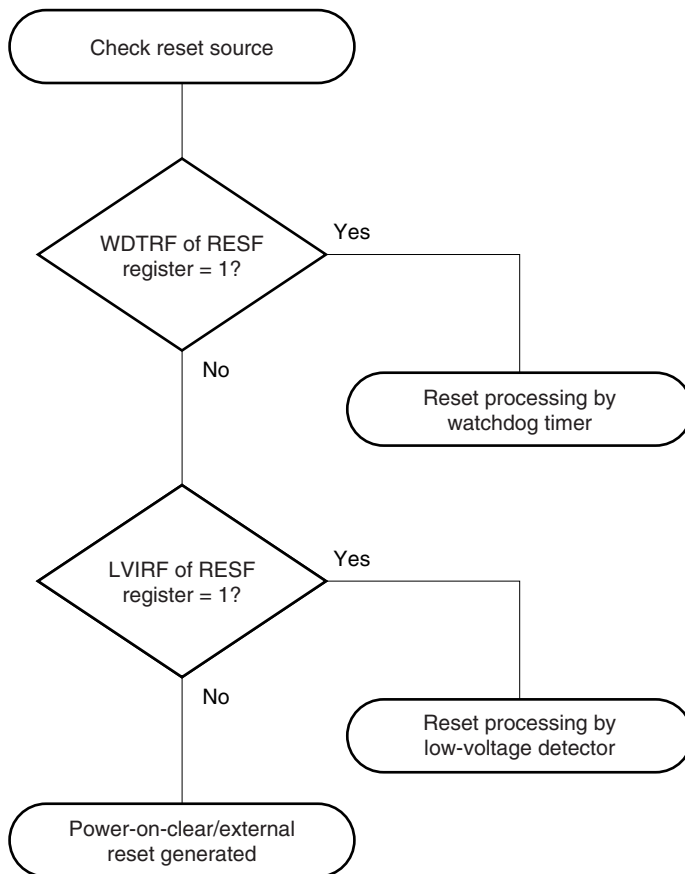
- If supply voltage fluctuation is 50 ms or less in vicinity of POC detection voltage



- 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 15-3. Example of Software Processing After Reset Release (2/2)

- Checking reset source



## CHAPTER 16 LOW-VOLTAGE DETECTOR

### 16.1 Functions of Low-Voltage Detector

The low-voltage detector (LVI) 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 ( $V_{EXLVI} = 1.21 \text{ V (TYP.)}$ : 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 (11 levels) of supply voltage can be changed by software.
- Operable in STOP mode.
- RAM data retention detection (refer **16.6 RAM Data Retention Detector**)

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

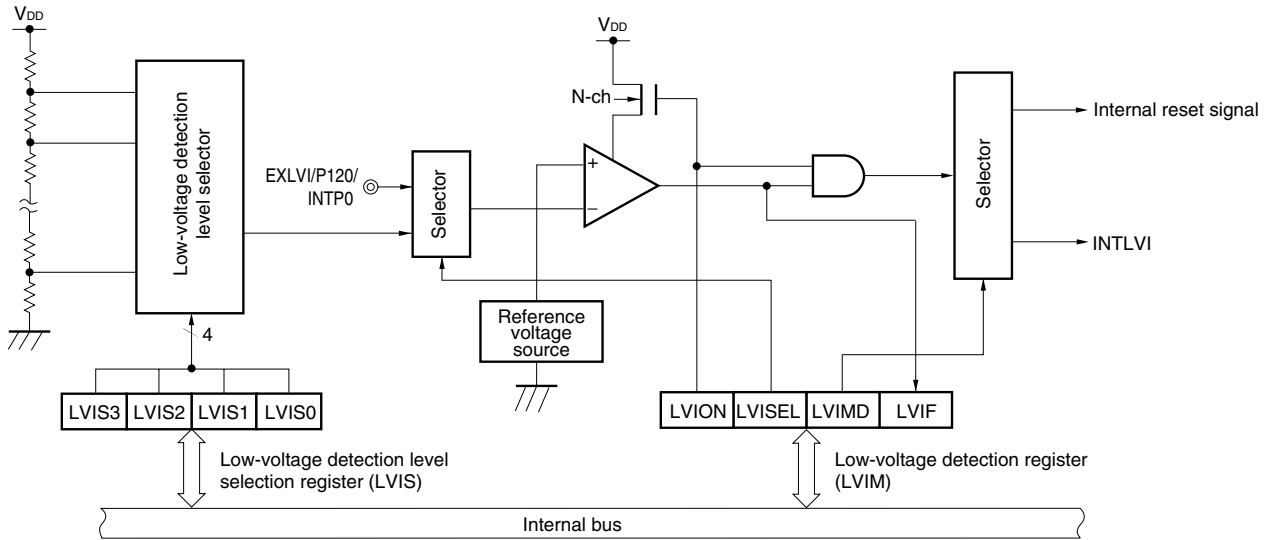
While the low-voltage detector is operating, whether the supply voltage or the input voltage from an external input pin is more than or less than the detection level can be checked by reading the low-voltage detection flag (LVIF: bit 0 of LVIM).

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 14 RESET FUNCTION**.

## 16.2 Configuration of Low-Voltage Detector

The block diagram of the low-voltage detector is shown in Figure 16-1.

**Figure 16-1. Block Diagram of Low-Voltage Detector**



## 16.3 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 16-2. Format of Low-Voltage Detection Register (LVIM)

Address: FFBEH After reset: 00H<sup>Note 1</sup> R/W<sup>Note 2</sup>

Symbol	<7>	6	5	4	3	<2>	<1>	<0>
LVIM	LVION	0	0	0	0	LVISEL	LVIMD	LVIF

LVION <sup>Notes 3, 4</sup>	Enables low-voltage detection operation
0	Disables operation
1	Enables operation

LVISEL <sup>Note 3</sup>	Voltage detection selection
0	Detects level of supply voltage ( $V_{DD}$ )
1	Detects level of input voltage from external input pin (EXLVI)

LVIMD <sup>Note 3</sup>	Low-voltage detection operation mode (interrupt/reset) selection
0	<ul style="list-style-type: none"> <li>LVISEL = 0: Generates an internal interrupt signal when the supply voltage (<math>V_{DD}</math>) drops lower than the detection voltage (<math>V_{LVI}</math>) (<math>V_{DD} &lt; V_{LVI}</math>) or when <math>V_{DD}</math> becomes <math>V_{LVI}</math> or higher (<math>V_{DD} \geq V_{LVI}</math>).</li> <li>LVISEL = 1: Generates an interrupt signal when the input voltage from an external input pin (EXLVI) drops lower than the detection voltage (<math>V_{EXLVI}</math>) (<math>EXLVI &lt; V_{EXLVI}</math>) or when EXLVI becomes <math>V_{EXLVI}</math> or higher (<math>EXLVI \geq V_{EXLVI}</math>).</li> </ul>
1	<ul style="list-style-type: none"> <li>LVISEL = 0: Generates an internal reset signal when the supply voltage (<math>V_{DD}</math>) &lt; detection voltage (<math>V_{LVI}</math>) and releases the reset signal when <math>V_{DD} \geq V_{LVI}</math>.</li> <li>LVISEL = 1: Generates an internal reset signal when the input voltage from an external input pin (EXLVI) &lt; detection voltage (<math>V_{EXLVI}</math>) and releases the reset signal when <math>EXLVI \geq V_{EXLVI}</math>.</li> </ul>

LVIF <sup>Note 4</sup>	Low-voltage detection flag
0	<ul style="list-style-type: none"> <li>LVISEL = 0: Supply voltage (<math>V_{DD}</math>) <math>\geq</math> detection voltage (<math>V_{LVI}</math>), or when operation is disabled</li> <li>LVISEL = 1: Input voltage from external input pin (EXLVI) <math>\geq</math> detection voltage (<math>V_{EXLVI}</math>), or when operation is disabled</li> </ul>
1	<ul style="list-style-type: none"> <li>LVISEL = 0: Supply voltage (<math>V_{DD}</math>) &lt; detection voltage (<math>V_{LVI}</math>)</li> <li>LVISEL = 1: Input voltage from external input pin (EXLVI) &lt; detection voltage (<math>V_{EXLVI}</math>)</li> </ul>

- 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  $\mu$ s (MAX.)) from when LVION is set to 1 until operation is stabilized. After operation has stabilized, 50  $\mu$ s (TYP.) are 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  $EXLVI < V_{DD}$ .
  - After an LVI reset has been generated, do not write values to LVIS and LVIM when LVION = 1.
  - 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 LVIIF becomes 1.

**(2) Low-voltage detection level selection register (LVIS)**

This register selects the low-voltage detection level of supply voltage ( $V_{DD}$ ).

When an input voltage from the external input pin (EXLVI) is detected, the detection voltage ( $V_{EXLVI} = 1.21\text{ V}$  (TYP.)) is fixed. Therefore, setting of LVIS is not necessary.

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 16-3. Format of Low-Voltage Detection Level Selection Register (LVIS)**

Address: FFBFH After reset: 00H<sup>Note 1</sup> R/W

Symbol	7	6	5	4	3	2	1	0
LVIS	0	0	0	0	LVIS3	LVIS2	LVIS1	LVIS0

LVIS3	LVIS2	LVIS1	LVIS0	Detection level
Other than below				Setting prohibited
0	1	0	1	$V_{LV15}$ (3.47 V $\pm$ 0.1 V)
0	1	1	0	$V_{LV16}$ (3.32 V $\pm$ 0.1 V)
0	1	1	1	$V_{LV17}$ (3.16 V $\pm$ 0.1 V)
1	0	0	0	$V_{LV18}$ (3.01 V $\pm$ 0.1 V)
1	0	0	1	$V_{LV19}$ (2.85 V $\pm$ 0.1 V)
1	0	1	0	$V_{LV110}$ (2.70 V $\pm$ 0.1 V)
1	0	1	1	$V_{LV111}$ (2.55 V $\pm$ 0.1 V)
1	1	0	0	$V_{LV112}$ (2.39 V $\pm$ 0.1 V)
1	1	0	1	$V_{LV113}$ (2.24 V $\pm$ 0.1 V)
1	1	1	0	$V_{LV114}$ (2.05 V $\pm$ 0.05 V)
1	1	1	1	$V_{LV115}$ (1.93 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 4 to 7 to “0”.
  2. Do not change the value of LVIS during LVI operation.
  3. After an LVI reset has been generated, do not write values to LVIS and LVIM when LVION = 1.

**(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 16-4. 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)

**16.4 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



## 16.4.1 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 bits 3 to 0 ( $LVIS3$  to  $LVIS0$ ) 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 ( $10 \mu s$  (MAX.)).
  - <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 16-5 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
 

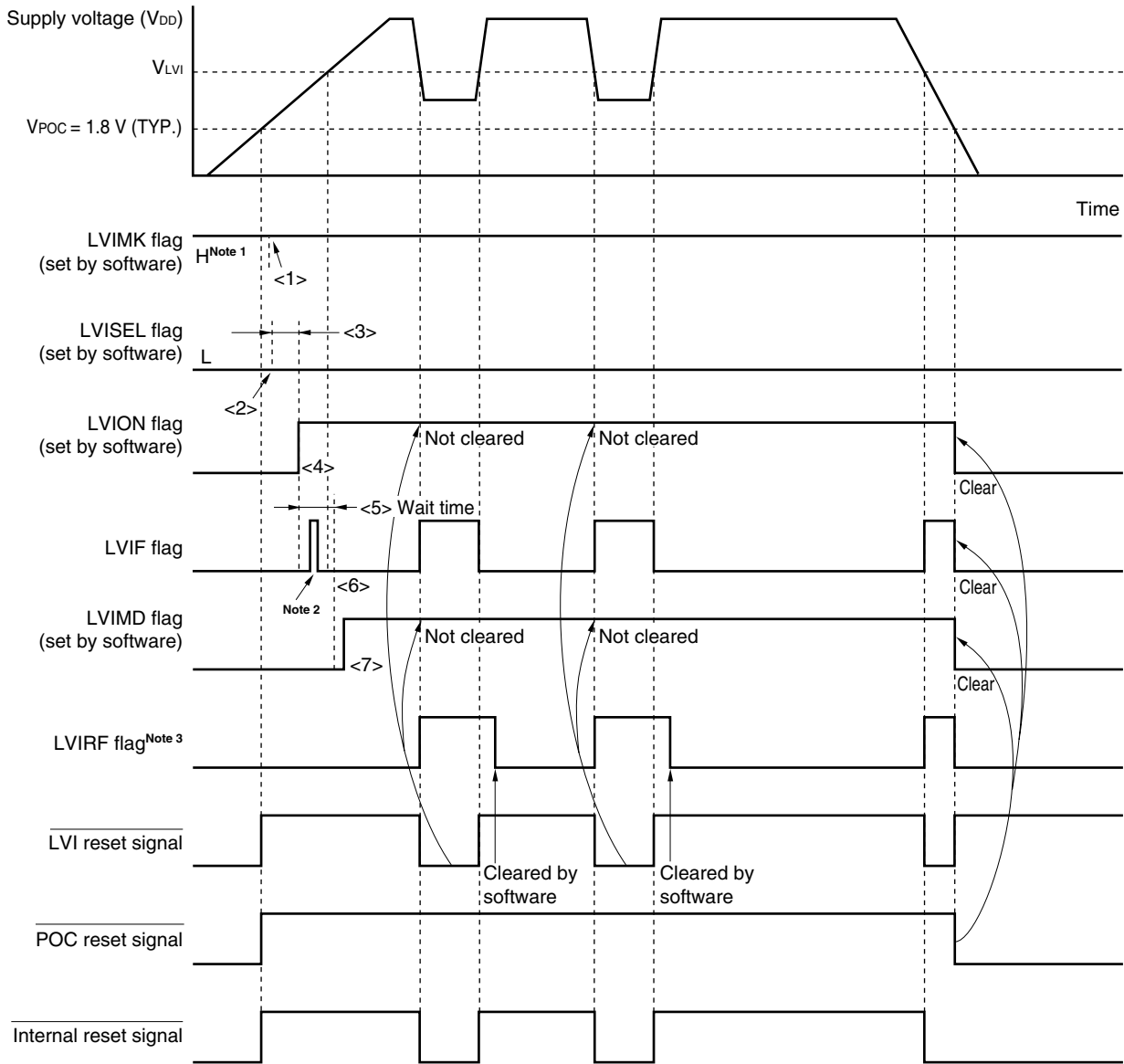
Either of the following procedures must be executed.

  - 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 16-5. Timing of Low-Voltage Detector Internal Reset Signal Generation (Detects Level of Supply Voltage ( $V_{DD}$ ))**



- 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 14 RESET FUNCTION**.

**Remark** <1> to <7> in Figure 16-5 above correspond to <1> to <7> in the description of "When starting operation" in **16.4.1 (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  $\mu$ s (MAX.)).
  - <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 16-6 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  $EXLVI \leq V_{DD}$ .

- When stopping operation
 

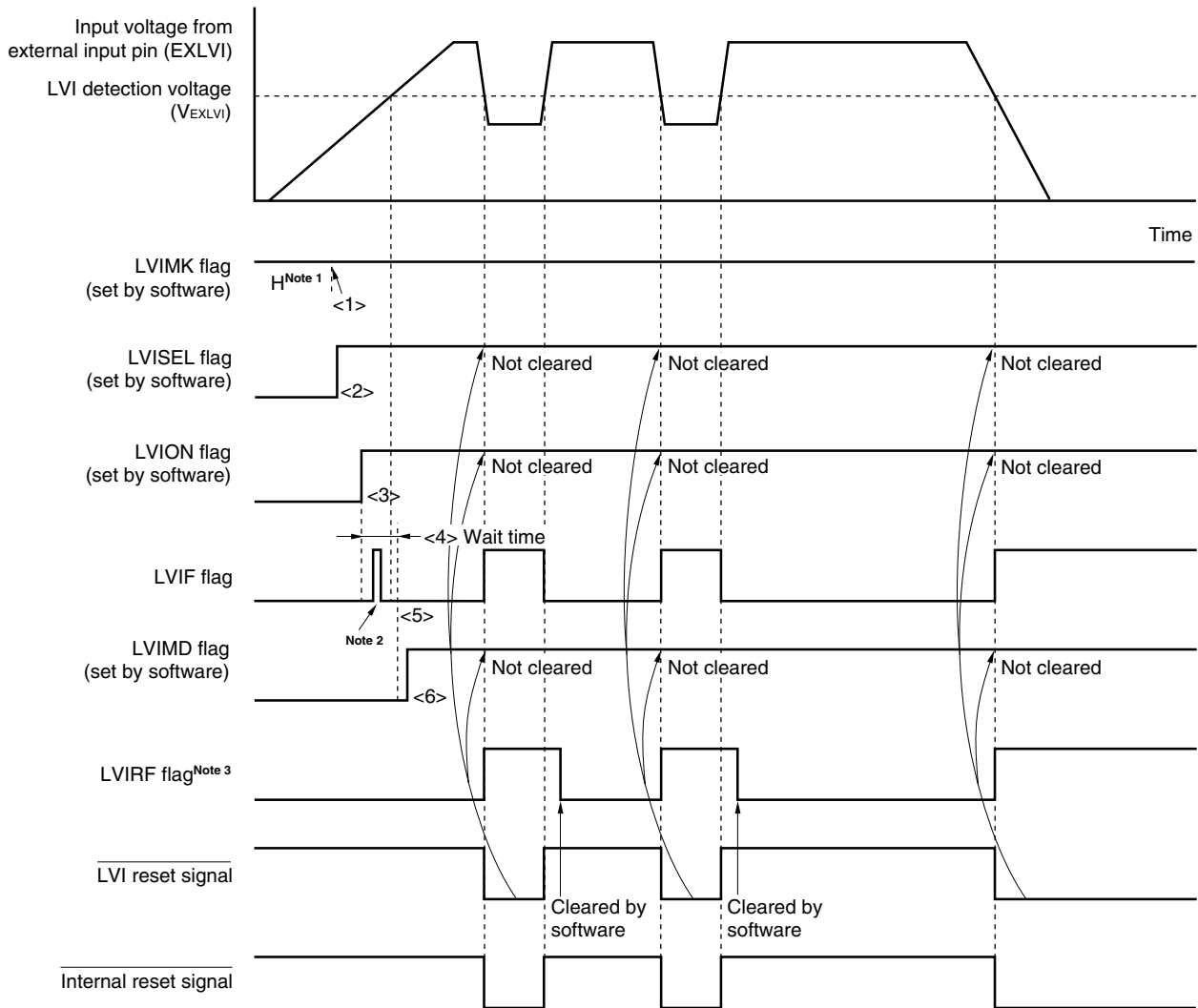
Either of the following procedures must be executed.

  - 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 16-6. 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 14 RESET FUNCTION**.

**Remark** <1> to <6> in Figure 16-6 above correspond to <1> to <6> in the description of “When starting operation” in **16.4.1 (2) When detecting level of input voltage from external input pin (EXLVI)**.

## 16.4.2 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 bits 3 to 0 (LVIS3 to 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 (MAX.)).
  - <6> 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.
  - <7> Clear the interrupt request flag of LVI (LVIIF) to 0.
  - <8> Release the interrupt mask flag of LVI (LVIMK).
  - <9> Clear bit 1 (LVIMD) of LVIM to 0 (generates interrupt signal when the level is detected) (default value).
  - <10> Execute the EI instruction (when vector interrupts are used).

Figure 16-7 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
 

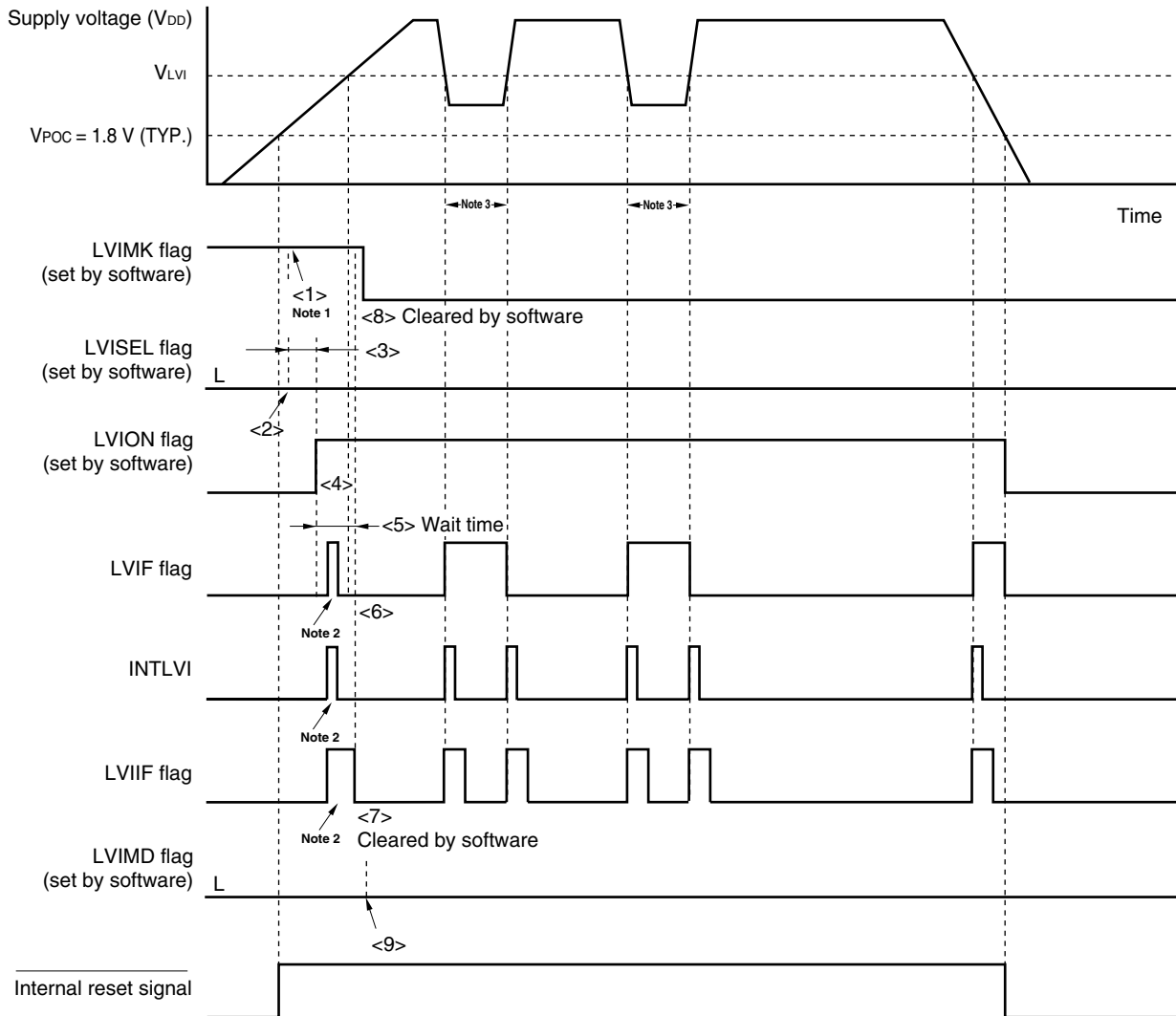
Either of the following procedures must be executed.

  - When using 8-bit memory manipulation instruction:
 

Write 00H to LVIM.
  - When using 1-bit memory manipulation instruction:
 

Clear LVION to 0.

**Figure 16-7. Timing of Low-Voltage Detector Interrupt Signal Generation  
(Detects Level of Supply Voltage ( $V_{DD}$ ))**



- 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 16-7 above correspond to <1> to <9> in the description of “When starting operation” in **16.4.2 (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> Set bit 7 (LVION) of LVIM to 1 (enables LVI operation).
  - <4> Use software to wait for an operation stabilization time (10  $\mu$ S (MAX.)).
  - <5> 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.
  - <6> Clear the interrupt request flag of LVI (LVIF) to 0.
  - <7> Release the interrupt mask flag of LVI (LVIMK).
  - <8> Clear bit 1 (LVIMD) of LVIM to 0 (generates interrupt signal when the level is detected) (default value).
  - <9> Execute the EI instruction (when vector interrupts are used).

Figure 16-8 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  $EXLVI \leq V_{DD}$ .

- When stopping operation
 

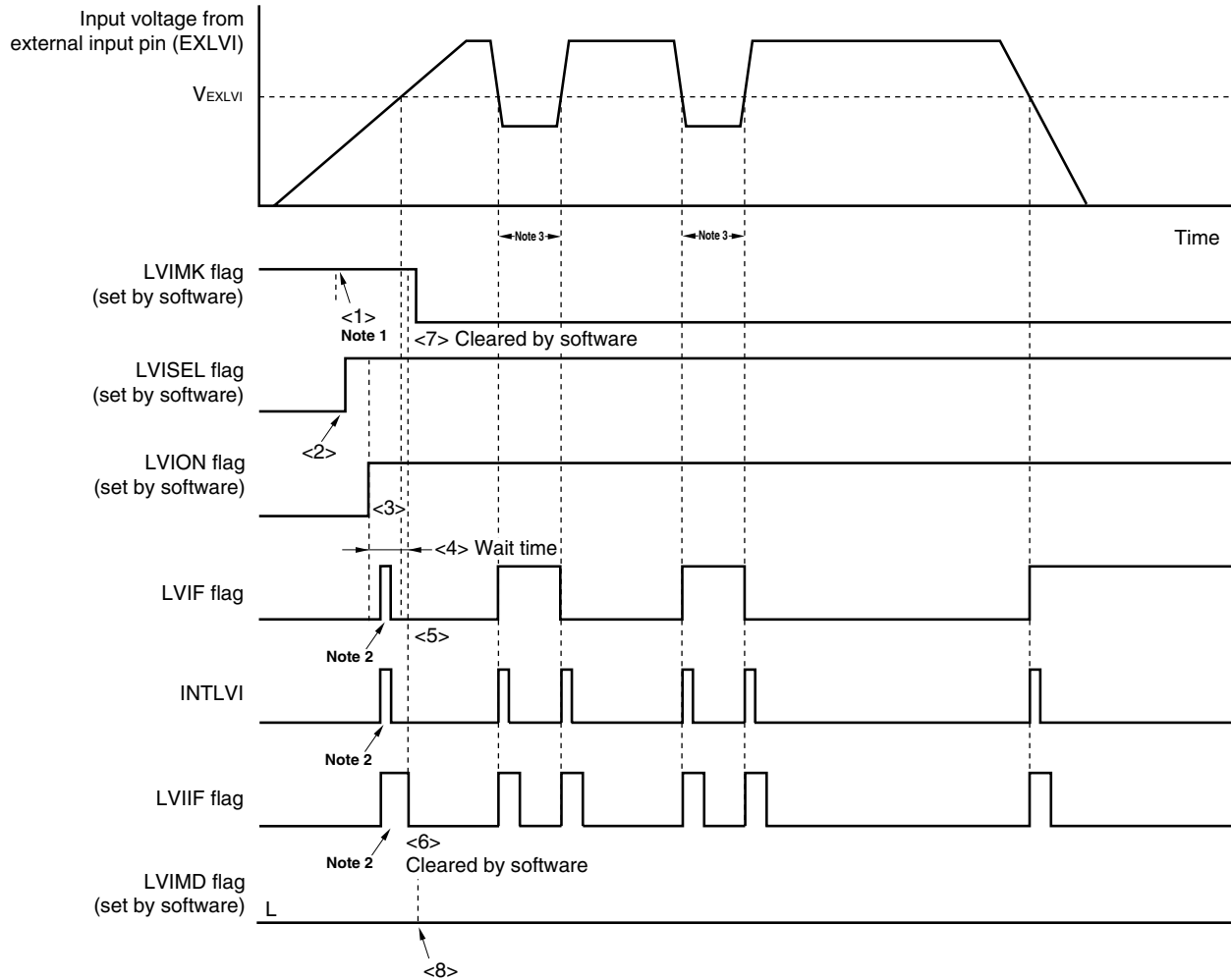
Either of the following procedures must be executed.

  - When using 8-bit memory manipulation instruction:
 

Write 00H to LVIM.
  - When using 1-bit memory manipulation instruction:
 

Clear LVION to 0.

**Figure 16-8. 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 16-8 above correspond to <1> to <8> in the description of "When starting operation" in 16.4.2 (2) **When detecting level of input voltage from external input pin (EXLVI)**.



## 16.5 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 16-9).

### (2) When used as interrupt

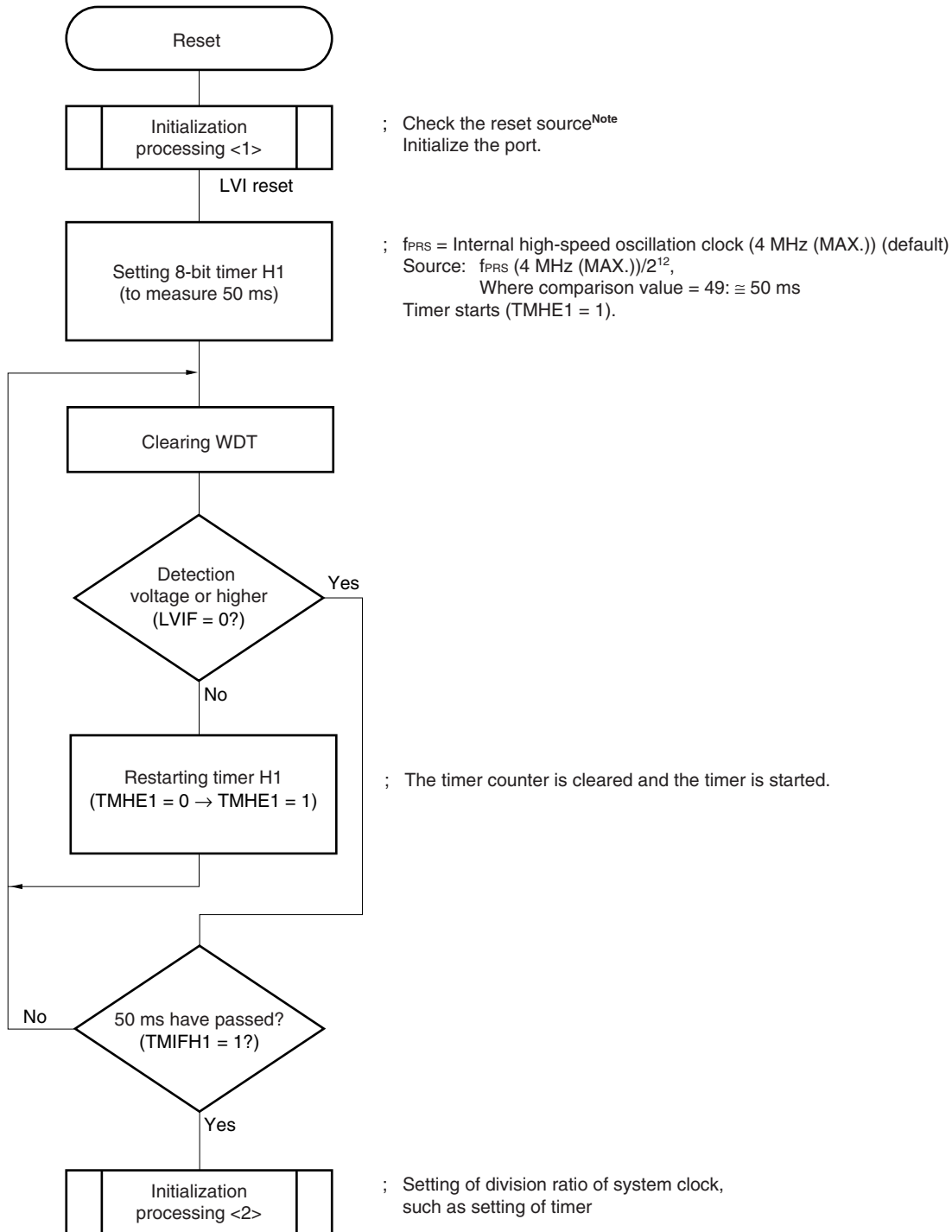
- (a) Confirm that “supply voltage ( $V_{DD}$ )  $\geq$  detection voltage ( $V_{LVI}$ )” when detecting the falling edge of  $V_{DD}$ , or “supply voltage ( $V_{DD}$ )  $<$  detection voltage ( $V_{LVI}$ )” when detecting the rising edge of  $V_{DD}$ , in the servicing routine of the LVI interrupt by 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 ( $V_{EXLVI} = 1.21 \text{ V (TYP.)}$ )

Figure 16-9. Example of Software Processing After Reset Release (1/2)

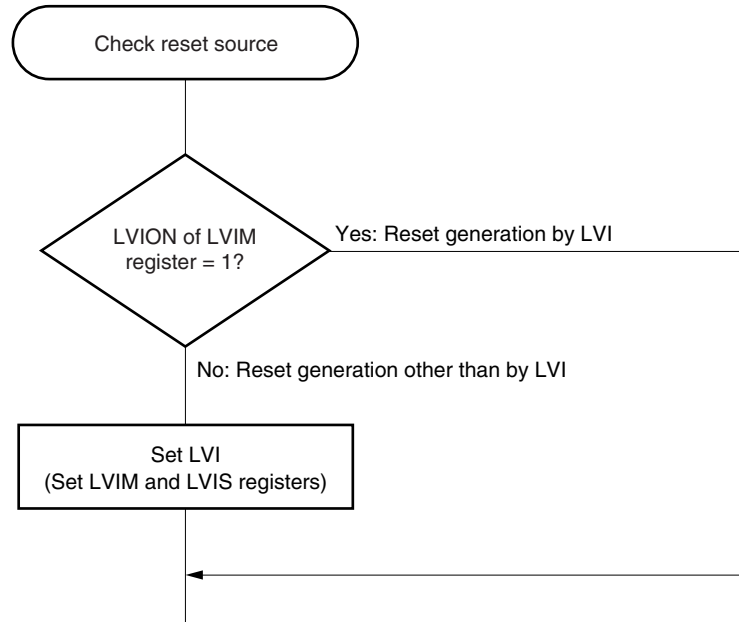
- If supply voltage fluctuation is 50 ms or less in vicinity of LVI detection voltage



**Note** A flowchart is shown on the next page.

Figure 16-9. Example of Software Processing After Reset Release (2/2)

- Checking reset source



## 16.6 RAM Data Retention Detector

By using this circuit, it is possible to judge whether power supply ( $V_{DD}$ ) drops below than the voltage which can not retain the data value of RAM when Battery is changed.

The RAM data retention detection voltage ( $V_{LD}$ ) is 1.4V +/- 0.1V.

The RAM data retention detector is controlled by the following registers.

- RAM Data Retention control register (LVDET)

### (1) RAM Data Retention control register (LVDET)

This register sets RAM data retention detection and the operation mode.

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

**Figure 16-10. Format of RAM Data Retention Control Register (LVDET)**

Address: FFB0H After reset : Undefined <sup>Note</sup> R/W

Symbol	7	6	5	4	3	2	1	0
LVDET	0	0	0	0	0	0	LVDET0	0

LVDET0	RAM data retention flag
0	RAM data may change
1	RAM data are retained

**Note** When  $V_{DD}$  is below than  $V_{LD}$  (1.4V(Typ.)), the value becomes "00H". In other case, the value of LVDET will be remained. When RAM data retention detector function is used, please set LVDET0 to "1".

RAM data retention flag will be cleared to "0" when the power supply ( $V_{DD}$ ) drops below than RAM data retention detect voltage ( $V_{LD}$ :1.4+/-0.1 V).

It is possible to judge whether RAM data value are remained by checking RAM data retention flag (LVDET0) after reset.

## CHAPTER 17 OPTION BYTE

### 17.1 Functions of Option Bytes

The flash memory at 0080H to 0084H of the  $\mu$ PD179F11x, 179F12x microcontroller 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.

**Caution** Be sure to set 00H to 0081H and 0083H.

#### (1) 0080H

- Internal low-speed oscillator operation
  - Can be stopped by software
  - Cannot be stopped
- Watchdog timer interval time setting
- Watchdog timer counter operation
  - Enabled counter operation
  - Disabled counter operation
- Watchdog timer window open period setting

#### (2) 0084H

- On-chip debug operation control
  - 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

## 17.2 Format of Option Byte

The format of the option byte is shown below.

Figure 17-1. 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

<R>

WDCS2	WDCS1	WDCS0	Watchdog timer overflow time
0	0	0	$2^{10}/f_{RL}$ (3.41 ms)
0	0	1	$2^{11}/f_{RL}$ (6.83 ms)
0	1	0	$2^{12}/f_{RL}$ (13.65 ms)
0	1	1	$2^{13}/f_{RL}$ (27.31 ms)
1	0	0	$2^{14}/f_{RL}$ (54.61 ms)
1	0	1	$2^{15}/f_{RL}$ (109.23 ms)
1	1	0	$2^{16}/f_{RL}$ (218.45 ms)
1	1	1	$2^{17}/f_{RL}$ (436.91 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.

<R>

2. The watchdog timer continues its operation during self programming of the flash memory. During processing, the interrupt acknowledge time is delayed. Set the overflow time and window size taking this delay into consideration.
3. 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 1 (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.
4. Be sure to clear bit 7 to 0.

**Remarks** 1.  $f_{RL}$ : Internal low-speed oscillation clock frequency

<R>

2. ( ):  $f_{RL} = 300$  kHz (MAX.)

Figure 17-1. Format of Option Byte (2/2)

Address: 0081H to 0083H<sup>Note</sup>

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

**Note** Be sure to set 00H to 0081H and 0083H, as these addresses are reserved areas.

Address: 0084H<sup>Note</sup>

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** To use the on-chip debug function, set 02H or 03H to 0084H.

**Remark** For the on-chip debug security ID, see **CHAPTER 19 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 <sup>10</sup> /f <sub>RL</sub> , ; Internal low-speed oscillator can be stopped by software.
	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 14 RESET FUNCTION**.

## CHAPTER 18 FLASH MEMORY

The  $\mu$ PD179F11x, 179F12x microcontrollers incorporate the flash memory to which a program can be written, erased, and overwritten while mounted on the board.

### 18.1 Internal Memory Size Switching Register

The internal memory capacity can be selected 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 18-1 after a reset release.

**Figure 18-1. 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
0	1	0	512 bytes
1	1	0	1024 bytes
Other than above			Setting prohibited

ROM3	ROM2	ROM1	ROM0	Internal ROM capacity selection
0	0	0	1	4 KB
0	0	1	0	8 KB
0	1	0	0	16 KB
0	1	1	0	24 KB
1	0	0	0	32 KB
Other than above				Setting prohibited

**Table 18-1. Internal Memory Size Switching Register Settings**

Flash Memory Versions ( $\mu$ PD179F11x, 179F12x microcontroller)	IMS Setting
$\mu$ PD179F110	41H
$\mu$ PD179F111	42H
$\mu$ PD179F112, 179F122	04H
$\mu$ PD179F113, 179F123	C6H
$\mu$ PD179F114, 179F124	C8H



## 18.2 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  $\mu$ PD179F11x, 179F12x microcontrollers 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  $\mu$ PD179F11x, 179F12x microcontrollers are mounted on the target system.

**Remark** The FA series is a product of Naito Densai Machida Mfg. Co., Ltd.

**Table 18-2. Example of wiring Between  $\mu$ PD179F11x microcontroller and Dedicated Flash Memory Programmer**

Pin Configuration of Dedicated Flash Memory Programmer			$\mu$ PD179F11x microcontroller	
Signal Name	I/O	Pin Function	Pin Name	Pin No.
SI/RxD	Input	Receive signal	TxD6/P13	29
SO/TxD	Output	Transmit signal	RxD6/P14	28
SCK	Output	Transfer clock	–	–
CLK	Output	Clock to $\mu$ PD179F11x, 179F12x microcontroller	<b>Note</b>	<b>Note</b>
/RESET	Output	Reset signal	$\overline{\text{RESET}}$	6
FLMD0	Output	Mode signal	FLMD0	9
V <sub>DD</sub>	I/O	V <sub>DD</sub> voltage generation/ power monitoring	V <sub>DD</sub>	14
			AV <sub>REF</sub>	33
GND	–	Ground	V <sub>SS</sub>	13
			AV <sub>SS</sub>	34

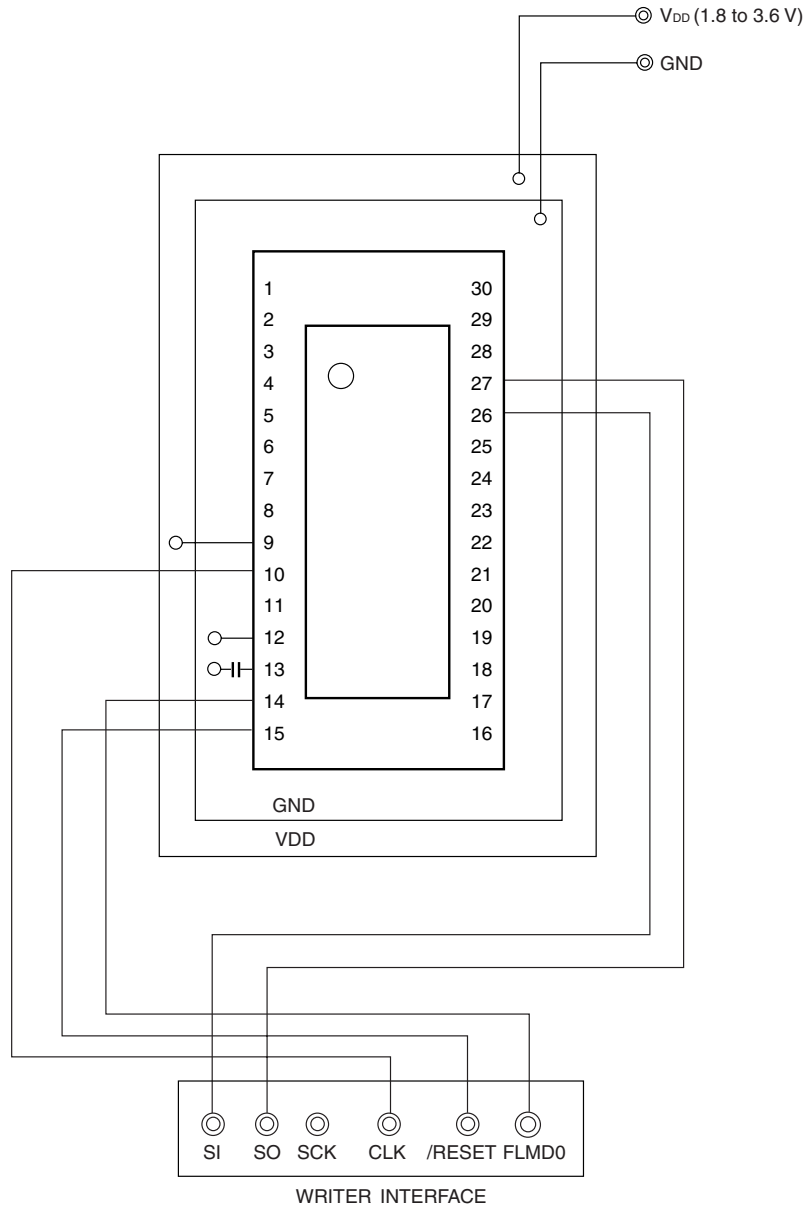
**Note** Only the X1 clock (fx) or external main system clock (f<sub>EXCLK</sub>) can be used. When using the clock output of the dedicated flash memory programmer, pin connection varies depending on the type of the dedicated flash memory programmer used.

<R>

- PG-FP4, FL-PR4, PG-FP5, FL-PR5: Connect CLK of the programmer to EXCLK/X2/P122/OCD0B (pin 10).

Examples of the recommended connection when using the adapter for flash memory writing are shown below.

**Figure 18-2. Example of Wiring Adapter for Flash Memory Writing (30-Pin Products)**



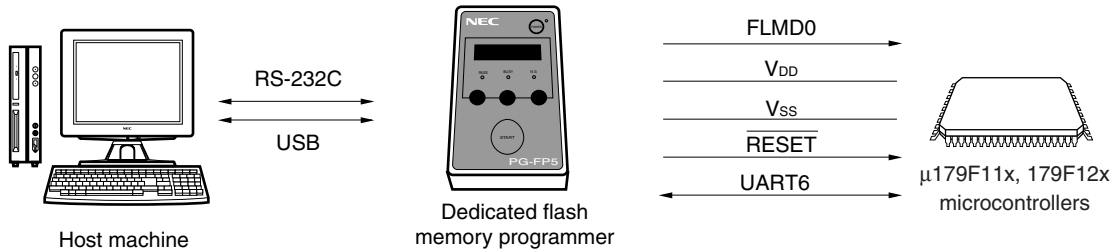
<R> **Remark** The above figure illustrates an example of wiring when using the clock output from the PG-FP4, FL-PR4, PG-FP5 or FL-PR5.

### 18.3 Programming Environment

The environment required for writing a program to the flash memory of the  $\mu$ PD179F11x, 179F12x microcontrollers are illustrated below.

<R>

**Figure 18-3. 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  $\mu$ PD179F11x, 179F12x microcontroller, 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.

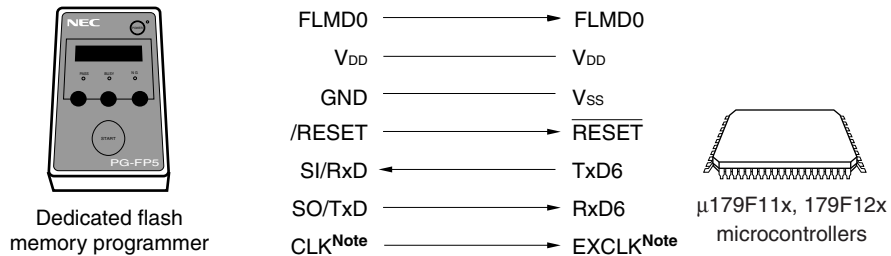
### 18.4 Communication Mode

Communication between the dedicated flash memory programmer and the  $\mu$ PD179F11x, 179F12x microcontrollers are established by serial communication of the  $\mu$ PD179F11x, 179F12x microcontrollers.

<R>

**Figure 18-4. Communication with Dedicated Flash Memory Programmer**

Transfer rate: 115200 bps



**Note** The above figure illustrates an example of wiring when using the clock output from the PG-FP4, FL-PR4, PG-FP5 or FL-PR5.

The dedicated flash memory programmer generates the following signals for the  $\mu$ PD179F11x, 179F12x microcontroller. For details, refer to the user's manual for the PG-FP4 or FL-PR4.

**Table 18-3. Pin Connection**

Dedicated Flash Memory Programmer			$\mu$ PD179F11x, 179F12x microcontrollers	Connection
Signal Name	I/O	Pin Function	Pin Name	
FLMD0	Output	Mode signal	FLMD0	⊙
V <sub>DD</sub>	I/O	V <sub>DD</sub> voltage generation/power monitoring	V <sub>DD</sub> , AV <sub>REF</sub>	⊙
GND	–	Ground	V <sub>SS</sub> , AV <sub>SS</sub>	⊙
CLK	Output	Clock output to $\mu$ PD179F11x, 179F12x microcontroller	<b>Note 1</b>	○ <sup>Note 1</sup>
/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	$\overline{\text{SCK10}}$	×

**Note** Only the X1 clock (fx) or external main system clock (f<sub>EXCLK</sub>) can be used. When using the clock output of the dedicated flash memory programmer, pin connection varies depending on the type of the dedicated flash memory programmer used.

<R> • PG-FP4, FL-PR4, PG-FP5, FL-PR5: Connect CLK of the programmer to EXCLK/X2/P122.

**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.

## 18.5 Handling of Pins on Board

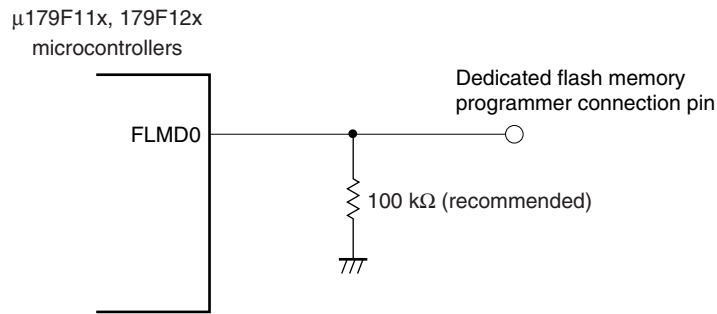
To write the flash memory on-board, connectors that connect the dedicated flash memory programmer must be provided on the target system. First provide a function that selects the normal operation mode or flash memory programming mode on the board.

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.

### 18.5.1 FLMD0 pin

In the normal operation mode, 0 V is input to the FLMD0 pin. In the flash memory programming mode, the V<sub>DD</sub> write voltage is supplied to the FLMD0 pin. An FLMD0 pin connection example is shown below.

**Figure 18-5. FLMD0 Pin Connection Example**



**18.5.2 Serial interface pins**

The pins used by each serial interface are listed below.

**Table 18-4. Pins Used by Each Serial Interface**

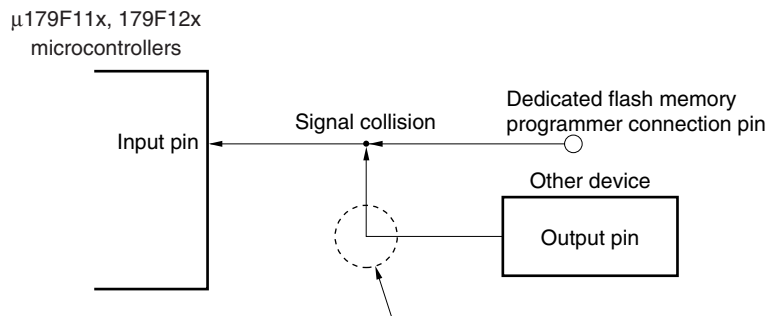
Serial Interface	Pins Used
UART6	TxD6, RxD6

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 18-6. Signal Collision (Input Pin of Serial Interface)**

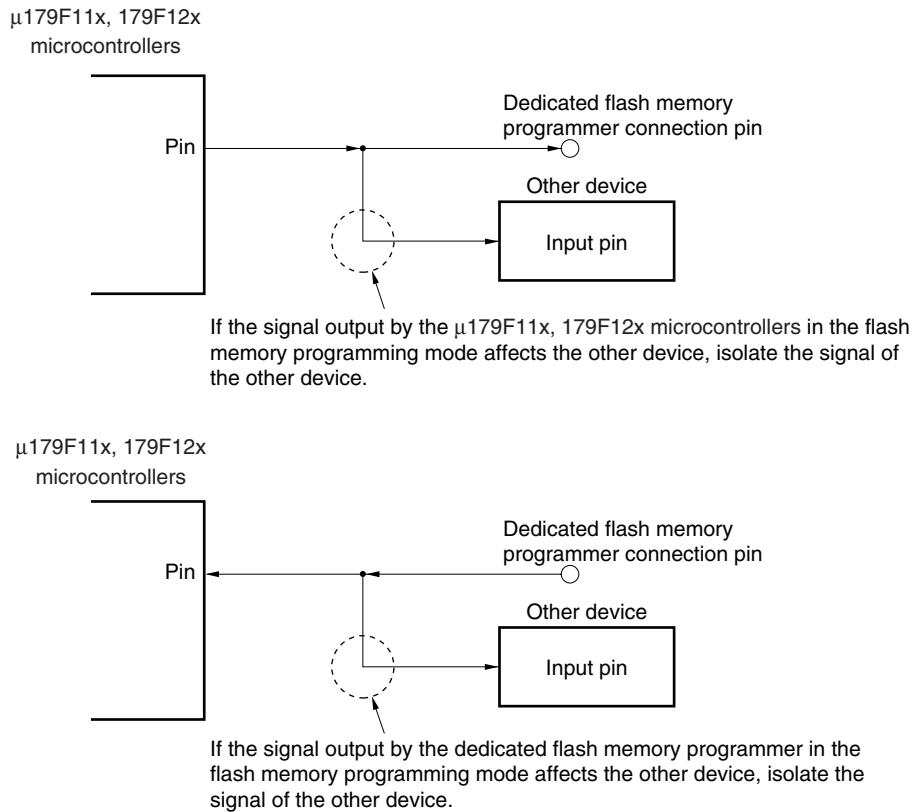


In the flash memory programming mode, the signal output by the device collides with the signal sent from the dedicated flash memory programmer. Therefore, isolate the signal of the other device.

**(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 18-7. Malfunction of Other Device

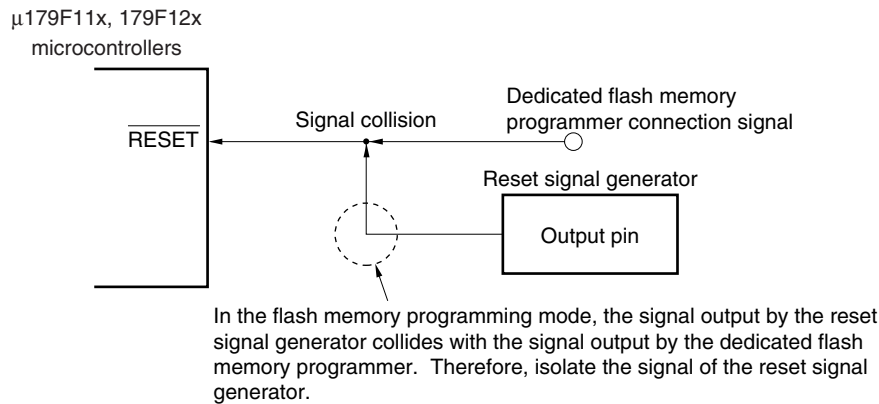


18.5.3 RESET pin

If the reset signal of the dedicated flash memory programmer is connected to the 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 18-8. Signal Collision (RESET Pin)



#### 18.5.4 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.

#### 18.5.5 REGC pin

Connect the REGC pin to GND via a capacitor (0.47  $\mu$ F: recommended) in the same manner as during normal operation.

#### 18.5.6 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 as follows.

- PG-FP4, FL-PR4, PG-FP5, FL-PR5: Connect CLK of the programmer to EXCLK/X2/P122/OCD0B.

<R>

**Caution** Only the X1 clock (fx) or external main system clock ( $f_{EXCLK}$ ) can be used.

#### 18.5.7 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.

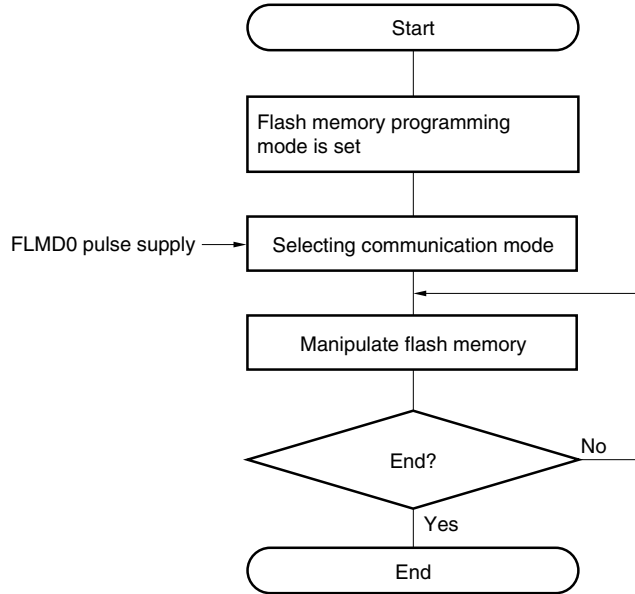
## 18.6 Programming Method

### 18.6.1 Controlling flash memory

The following figure illustrates the procedure to manipulate the flash memory.

<R>

**Figure 18-9. Flash Memory Manipulation Procedure**

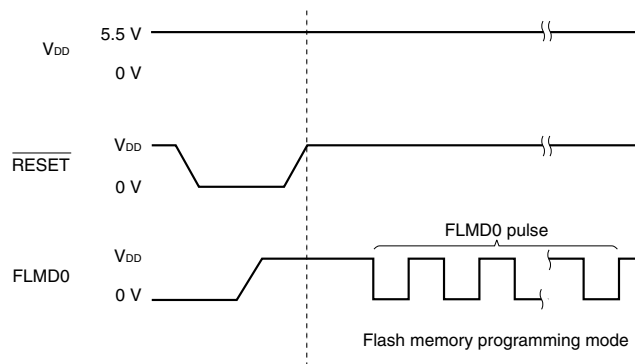


### 18.6.2 Flash memory programming mode

To rewrite the contents of the flash memory by using the dedicated flash memory programmer, set the  $\mu$ PD179F11x, 179F12x microcontroller 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 18-10. Flash Memory Programming Mode**



**Table 18-5. Relationship Between FLMD0 Pin and Operation Mode After Reset Release**

FLMD0	Operation Mode
0	Normal operation mode
$V_{DD}$	Flash memory programming mode



### 18.6.3 Selecting communication mode

In the  $\mu$ PD179F11x, 179F12x microcontrollers, 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 18-6. Communication Modes**

Communication Mode	Standard Setting <sup>Note 1</sup>				Pins Used	Periphera l Clock	Number of FLMD0 Pulses
	Port	Speed	Frequency	Multiply Rate			
UART (UART6)	UART-Ext-Osc	115,200 bps <sup>Note 3</sup>	2 to 4 MHz <sup>Note 2</sup>	1.0	TxD6, RxD6	f <sub>x</sub>	0
	UART-Ext-FP4CK					f <sub>EXCLK</sub>	3

- 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** 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**

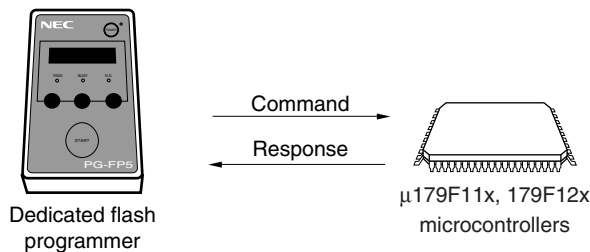
f<sub>x</sub>: X1 clock  
 f<sub>EXCLK</sub>: External main system clock  
 f<sub>RH</sub>: Internal high-speed oscillation clock

### 18.6.4 Communication commands

The  $\mu$ PD179F11x, 179F12x microcontrollers communicate with the dedicated flash memory programmer by using commands. The signals sent from the flash memory programmer to the  $\mu$ PD179F11x, 179F12x microcontrollers are called commands, and the signals sent from the  $\mu$ PD179F11x, 179F12x microcontrollers to the dedicated flash memory programmer are called response.

<R>

**Figure 18-11. Communication Commands**



The flash memory control commands of the  $\mu$ PD179F11x, 179F12x microcontrollers are listed in the table below. All these commands are issued from the programmer and the  $\mu$ PD179F11x, 179F12x microcontrollers perform processing corresponding to the respective commands.

Table 18-7. 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.
<R> Erase	Block Erase	Erases all 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 the device information (such as the part number and flash memory configuration).
	Version Get	Gets the device 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  $\mu$ PD179F11x, 179F12x microcontroller returns a response for the command issued by the dedicated flash memory programmer. The response names sent from the  $\mu$ PD179F11x, 179F12x microcontrollers are listed below.

Table 18-8. Response Names

Response Name	Function
ACK	Acknowledges command/data.
NAK	Acknowledges illegal command/data.

## 18.7 Security Settings

The  $\mu$ PD179F11x, 179F12x microcontrollers 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)  
Setting prohibited for  $\mu$ PD179F11x, 179F12x microcontrollers.
- Disabling block erase  
Execution of the block erase command for a specific block in the flash memory is prohibited during on-board/off-board programming. However, blocks can be erased by means of self programming.
- Disabling write  
Execution of the write and block erase commands for entire blocks in the flash memory is prohibited during on-board/off-board programming. However, blocks can be written by means of self programming.

- Disabling rewriting block 00H to block 03H

Execution of the block erase command, and write command on 00H to block 03H (0000H to 0FFFH) in the flash memory is prohibited by this setting.

**Caution** If a security setting that rewrites block 00H to block 03H has been applied, block 00H to block 03H of that device will not be rewritten.

<R>

The block erase, and write commands are enabled by the default setting when the flash memory is shipped. Security can be set by on-board/off-board programming. Each security setting can be used in combination.

Security setting can not be changed if once security prohibition setting is set for  $\mu$ PD179F11x, 179F12x microcontrollers

Table 18-9 shows the relationship between the erase and write commands when the  $\mu$ PD179F11x, 179F12x microcontrollers' security function is enabled.

**Table 18-9. Relationship Between Enabling Security Function and Command**

**(1) During on-board/off-board programming**

<R>

Valid Security	Executed Command	
	Block Erase <sup>Note</sup>	Write
Prohibition of block erase	Blocks cannot be erased.	Can be performed.
Prohibition of writing		Cannot be performed.
Prohibition of rewriting block 00H to block 03H		Boot cluster 0 cannot be written.

**Note** The block erase command erases all blocks in a batch when in on-board or off-board programming mode.

**(2) During self programming**

<R>

Valid Security	Executed Command	
	Block Erase <sup>Note</sup>	Write
Prohibition of block erase	Blocks can be erased.	Can be performed.
Prohibition of writing		
Prohibition of rewriting block 00H to block 03H	Boot cluster 0 cannot be erased.	Boot cluster 0 cannot be written.

**Note** The block erase command can be used to erase blocks in one-block (1 KB) units when in self programming mode.

Table 18-10 shows how to perform security settings in each programming mode.

**Table 18-10. Setting Security in Each Programming Mode**

**(1) On-board/off-board programming**

Security	Security Setting	How to Disable Security Setting
Prohibition of block erase	Set via GUI of dedicated flash memory programmer, etc.	Cannot be disabled after set.
Prohibition of writing		
Prohibition of rewriting block 00H to block 03H		

**(2) Self programming**

Security	Security Setting	How to Disable Security Setting
Prohibition of block erase	Cannot be set.	Cannot be disabled after set.
Prohibition of writing		
Prohibition of rewriting block 00H to block 03H		

<R>

## 18.8 Flash Memory Programming by Self-Programming

The  $\mu$ PD179F11x, 179F12x microcontrollers supports a self programming function that can be used to rewrite the flash memory via a user program. Because this function allows a user application to rewrite the flash memory by using a self programming library, it can be used to upgrade the program in the field.

If an interrupt occurs during self programming, self programming can be temporarily stopped and interrupt servicing can be executed. To execute interrupt servicing, restore the normal operation mode after self programming has been stopped, and execute the EI instruction. After the self programming mode is later restored, self programming can be resumed.

**Remark** For details of the self programming function and self programming library, refer to 78K0 Microcontroller Self-Programming Library Type01 User's Manual (U18274E).

### Cautions 1. Input a high level to the FLMD0 pin during self programming.

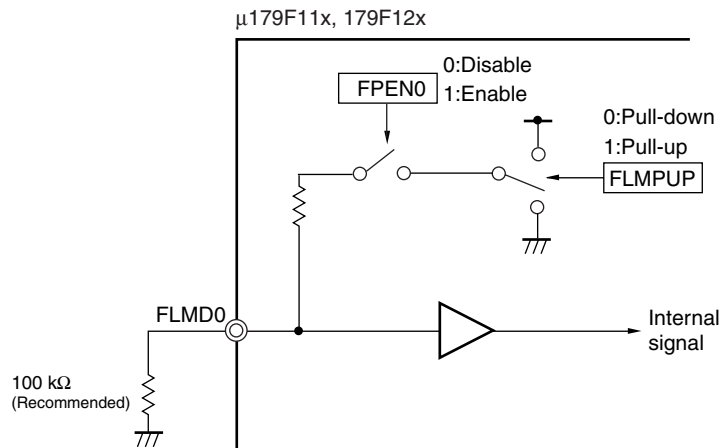
#### (1) Setting by External input

- Set FPEN0 (FLMD0 pull-up/pull-down enable register (FPEN) bit 0) to "0".
- Input High-level to FLMD0 (100KOhm pull-down)

#### (2) Setting by internal pull-up resistor

- Pull-down FLMD0 pin via 100KOhm.
- Set FPEN0 (FLMD0 pull-up/pull-down enable register (FPEN) bit 0) and FLMDPUP (FLMD0 pul-up/pull-down control register (FPCTL) bit 0) to "1". By setting them, input level of FLMD0 pin becomes "High".

Figure 18-12. FLMD0 pin setting when Self Programming



**Figure 18-13. Format of FLMD0 pull-up/pull-down control register (FPCTL)**

Address: FF35H    After reset: 00H    R/W

Symbol	7	6	5	4	3	2	1	0
FPCTL	0	0	0	0	0	0	0	FLMDPUP

FLMDPUP	FLMD0 Pull-up/Pull-down setting
0	Pull-down
1	Pull-up

**Figure 18-14. Format of FLMD0 pull-up/pull-down enable register (FPEN)**

Address: FF37H    After reset: 00H    R/W

Symbol	7	6	5	4	3	2	1	0
FPEN	0	0	0	0	0	0	0	FPEN0

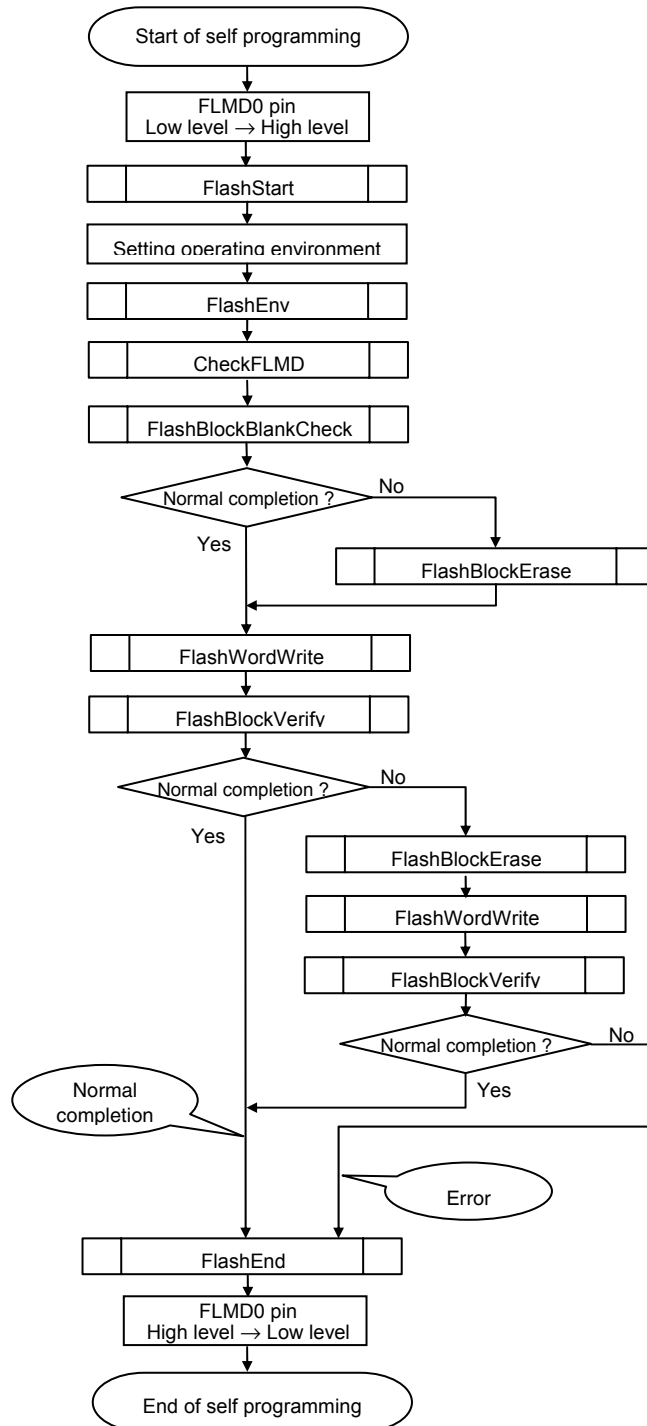
  

FPEN0	FLMD0 pull-up/pull-down enable setting
0	Disable pull-up/pull-down
1	Enable pull-up/pull-down

- Cautions**
2. **Be sure to execute the DI instruction before starting self programming.**  
The self programming function checks the interrupt request flags (IF0L, IF0H, IF1L). If an interrupt request is generated, self programming is stopped.
  3. **Self programming is also stopped by an interrupt request that is not masked even in theDI status. To prevent this, mask the interrupt by using the interrupt mask flag registers (MK0L, MK0H, MK1L).**

The following figure illustrates a flow of rewriting the flash memory by using a self programming library.

**Figure 18-15. Flow of Self Programming (Rewriting Flash Memory)**



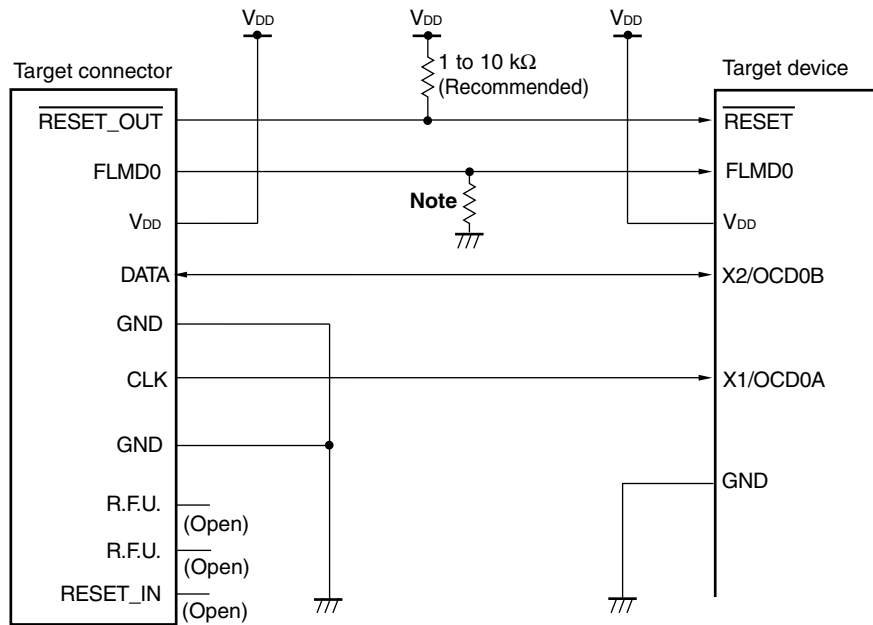
**Remark** For details of the self programming library, refer to **78K0 Microcontroller Self-Programming Library Type01 User's Manual (U18274E)**.

## CHAPTER 19 ON-CHIP DEBUG FUNCTION

### 19.1 Connecting QB-MINI2

The  $\mu$ PD179F11x, 179F12x microcontrollers use the  $V_{DD}$ , FLMD0,  $\overline{\text{RESET}}$ , OCD0A/X1 (or OCD1A/P04), OCD0B/X2 (or OCD1B/P05), and  $V_{SS}$  pins to communicate with the host machine via an on-chip debug emulator (QB-MINI2). Whether OCD0A/X1 and OCD1A/P04, or OCD0B/X2 and OCD1B/P05 are used can be selected.

**Figure 19-1. Connection Example of QB-MINI2 and  $\mu$ PD179F11x, 179F12x microcontrollers  
(When OCD0A/X1 and OCD0B/X2 Are Used)**

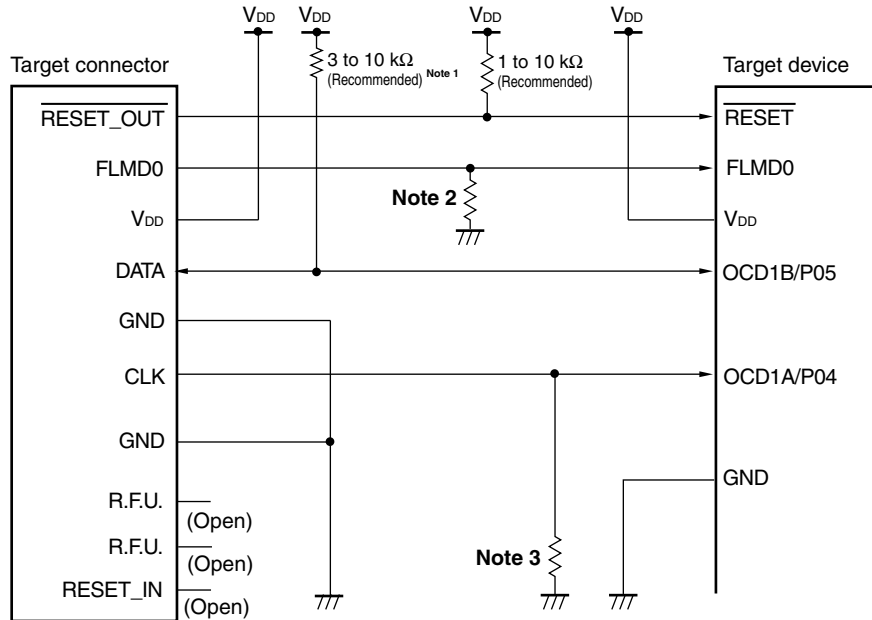


**Note** Make pull-down resistor 100 k $\Omega$ .

- Cautions**
1. Input the clock from the OCD0A/X1 pin during on-chip debugging.
  2. When OCD0A/X1 and OCD0B/X2 are used for OCD, The voltage level of OCD1A/P04 and OCD1B/P05 during reset have to be fixed level (High-level or Low-level).
  3. Because  $\overline{\text{RESET}}$  pin is used by QB-MINI2, alternate function can not be used for emulation.



**Figure 19-2. Connection Example of QB-MINI2 and  $\mu$ PD179F11x, 179F12x microcontroller  
(When OCD1A/P04 and OCD1B/P05 Are Used)**



- Notes**
1. This is the processing of the pin when OCD1B/P05 is set as the input port (to prevent the pin from being left opened when not connected to QB-MINI2).
  2. Make pull-down resistor 100 k $\Omega$ .
  3. Make pull-down resistor 10 k $\Omega$  (recommended).

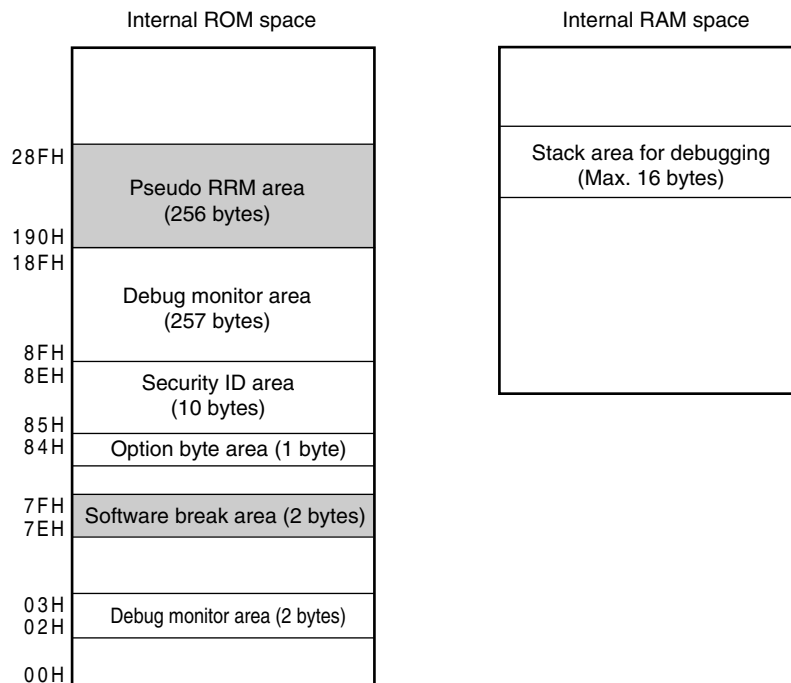
- Cautions**
1. When OCD1A/P04, OCD1B/P05 is used for On-chip debug, do not set P04 and P05 to output mode during on-chip debug mode.
  2. Because  $\overline{\text{RESET}}$  pin is used by QB-MINI2, alternate function can not be used for emulation.

## 19.2 Reserved Area Used by QB-MINI2

QB-MINI2 uses the reserved areas shown in Figure 19-3 below to implement communication with the target device, or each debug function. The shaded reserved areas are used for the respective debug functions to be used, and the other areas are always used for debugging. These reserved areas can be secured by using user programs and compiler options.

For details on reserved area, refer to **QB-MINI2 User's Manual (U18371E)**.

**Figure 19-3. Reserved Area Used by QB-MINI2**



**Remark** Shaded reserved areas: Area used for the respective debug functions to be used  
 Other reserved areas: Areas always used for debugging

## CHAPTER 20 INSTRUCTION SET

This chapter lists each instruction set of The  $\mu$ PD179F11x, 179F12x microcontrollers in table form. For details of each operation and operation code, refer to the separate document **78K/0 Series Instructions User's Manual (U12326E)**.

### 20.1 Conventions Used in Operation List

#### 20.1.1 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 20-1. Operand Identifiers and Specification Methods**

Identifier	Specification Method
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special function register symbol <sup>Note</sup>
sfrp	Special function register symbol (16-bit manipulatable register even addresses only) <sup>Note</sup>
saddr	FE20H to FF1FH Immediate data or labels
saddrp	FE20H to FF1FH Immediate data or labels (even address only)
addr16	0000H to FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions)
addr11	0800H to 0FFFH Immediate data or labels
addr5	0040H to 007FH Immediate data or labels (even address only)
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
RBn	RB0 to RB3

**Note** 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**.

**20.1.2 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 <sub>H</sub> , X <sub>L</sub> :	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
jdisp8:	Signed 8-bit data (displacement value)

**20.1.3 Description of flag operation column**

(Blank):	Not affected
0:	Cleared to 0
1:	Set to 1
×:	Set/cleared according to the result
R:	Previously saved value is restored

20.2 Operation List

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	<b>MOV</b>	r, #byte	2	4	–	r ← byte			
		saddr, #byte	3	6	7	(saddr) ← byte			
		sfr, #byte	3	–	7	sfr ← byte			
		A, r <small>Note 3</small>	1	2	–	A ← r			
		r, A <small>Note 3</small>	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, laddr16	3	8	9	A ← (addr16)			
		laddr16, A	3	8	9	(addr16) ← A			
		PSW, #byte	3	–	7	PSW ← byte			× × ×
		A, PSW	2	–	5	A ← PSW			
		PSW, A	2	–	5	PSW ← A			× × ×
		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				
	<b>XCH</b>	A, r <small>Note 3</small>	1	2	–	A ↔ r			
		A, saddr	2	4	6	A ↔ (saddr)			
		A, sfr	2	–	6	A ↔ sfr			
		A, laddr16	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<sub>CPU</sub>) 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	<b>MOVW</b>	rp, #word	3	6	–	$rp \leftarrow \text{word}$			
		saddrp, #word	4	8	10	$(saddrp) \leftarrow \text{word}$			
		sfrp, #word	4	–	10	$sfrp \leftarrow \text{word}$			
		AX, saddrp	2	6	8	$AX \leftarrow (saddrp)$			
		saddrp, AX	2	6	8	$(saddrp) \leftarrow AX$			
		AX, sfrp	2	–	8	$AX \leftarrow sfrp$			
		sfrp, AX	2	–	8	$sfrp \leftarrow AX$			
		AX, rp <small>Note 3</small>	1	4	–	$AX \leftarrow rp$			
		rp, AX <small>Note 3</small>	1	4	–	$rp \leftarrow AX$			
		AX, !addr16	3	10	12	$AX \leftarrow (addr16)$			
	!addr16, AX	3	10	12	$(addr16) \leftarrow AX$				
<b>XCHW</b>	AX, rp <small>Note 3</small>	1	4	–	$AX \leftrightarrow rp$				
8-bit operation	<b>ADD</b>	A, #byte	2	4	–	$A, CY \leftarrow A + \text{byte}$	x	x	x
		saddr, #byte	3	6	8	$(saddr), CY \leftarrow (saddr) + \text{byte}$	x	x	x
		A, r <small>Note 4</small>	2	4	–	$A, CY \leftarrow A + r$	x	x	x
		r, A	2	4	–	$r, CY \leftarrow r + A$	x	x	x
		A, saddr	2	4	5	$A, CY \leftarrow A + (saddr)$	x	x	x
		A, !addr16	3	8	9	$A, CY \leftarrow A + (addr16)$	x	x	x
		A, [HL]	1	4	5	$A, CY \leftarrow A + (HL)$	x	x	x
		A, [HL + byte]	2	8	9	$A, CY \leftarrow A + (HL + \text{byte})$	x	x	x
		A, [HL + B]	2	8	9	$A, CY \leftarrow A + (HL + B)$	x	x	x
	A, [HL + C]	2	8	9	$A, CY \leftarrow A + (HL + C)$	x	x	x	
	<b>ADDC</b>	A, #byte	2	4	–	$A, CY \leftarrow A + \text{byte} + CY$	x	x	x
		saddr, #byte	3	6	8	$(saddr), CY \leftarrow (saddr) + \text{byte} + CY$	x	x	x
		A, r <small>Note 4</small>	2	4	–	$A, CY \leftarrow A + r + CY$	x	x	x
		r, A	2	4	–	$r, CY \leftarrow r + A + CY$	x	x	x
		A, saddr	2	4	5	$A, CY \leftarrow A + (saddr) + CY$	x	x	x
		A, !addr16	3	8	9	$A, CY \leftarrow A + (addr16) + C$	x	x	x
		A, [HL]	1	4	5	$A, CY \leftarrow A + (HL) + CY$	x	x	x
		A, [HL + byte]	2	8	9	$A, CY \leftarrow A + (HL + \text{byte}) + CY$	x	x	x
		A, [HL + B]	2	8	9	$A, CY \leftarrow A + (HL + B) + CY$	x	x	x
A, [HL + C]		2	8	9	$A, CY \leftarrow A + (HL + C) + CY$	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	<b>SUB</b>	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 <small>Note 3</small>	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
	<b>SUBC</b>	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 <small>Note 3</small>	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) – CY	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
	<b>AND</b>	A, #byte	2	4	–	A ← A ∧ byte	x		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∧ byte	x		
		A, r <small>Note 3</small>	2	4	–	A ← A ∧ r	x		
		r, A	2	4	–	r ← r ∧ A	x		
		A, saddr	2	4	5	A ← A ∧ (saddr)	x		
		A, !addr16	3	8	9	A ← A ∧ (addr16)	x		
		A, [HL]	1	4	5	A ← A ∧ (HL)	x		
		A, [HL + byte]	2	8	9	A ← A ∧ (HL + byte)	x		
		A, [HL + B]	2	8	9	A ← A ∧ (HL + B)	x		
		A, [HL + C]	2	8	9	A ← A ∧ (HL + C)	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<sub>cpu</sub>) 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	<b>OR</b>	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	
	<b>XOR</b>	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	
	<b>CMP</b>	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<sub>CPU</sub>) 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	<b>ADDW</b>	AX, #word	3	6	–	$AX, CY \leftarrow AX + \text{word}$	x	x	x
	<b>SUBW</b>	AX, #word	3	6	–	$AX, CY \leftarrow AX - \text{word}$	x	x	x
	<b>CMPW</b>	AX, #word	3	6	–	$AX - \text{word}$	x	x	x
Multiply/divide	<b>MULU</b>	X	2	16	–	$AX \leftarrow A \times X$			
	<b>DIVUW</b>	C	2	25	–	$AX \text{ (Quotient)}, C \text{ (Remainder)} \leftarrow AX \div C$			
Increment/decrement	<b>INC</b>	r	1	2	–	$r \leftarrow r + 1$	x	x	
		saddr	2	4	6	$(\text{saddr}) \leftarrow (\text{saddr}) + 1$	x	x	
	<b>DEC</b>	r	1	2	–	$r \leftarrow r - 1$	x	x	
		saddr	2	4	6	$(\text{saddr}) \leftarrow (\text{saddr}) - 1$	x	x	
	<b>INCW</b>	rp	1	4	–	$rp \leftarrow rp + 1$			
	<b>DECW</b>	rp	1	4	–	$rp \leftarrow rp - 1$			
Rotate	<b>ROR</b>	A, 1	1	2	–	$(CY, A_7 \leftarrow A_0, A_{m-1} \leftarrow A_m) \times 1 \text{ time}$			x
	<b>ROL</b>	A, 1	1	2	–	$(CY, A_0 \leftarrow A_7, A_{m+1} \leftarrow A_m) \times 1 \text{ time}$			x
	<b>RORC</b>	A, 1	1	2	–	$(CY \leftarrow A_0, A_7 \leftarrow CY, A_{m-1} \leftarrow A_m) \times 1 \text{ time}$			x
	<b>ROLC</b>	A, 1	1	2	–	$(CY \leftarrow A_7, A_0 \leftarrow CY, A_{m+1} \leftarrow A_m) \times 1 \text{ time}$			x
	<b>ROR4</b>	[HL]	2	10	12	$A_{3-0} \leftarrow (HL)_{3-0}, (HL)_{7-4} \leftarrow A_{3-0}, (HL)_{3-0} \leftarrow (HL)_{7-4}$			
	<b>ROL4</b>	[HL]	2	10	12	$A_{3-0} \leftarrow (HL)_{7-4}, (HL)_{3-0} \leftarrow A_{3-0}, (HL)_{7-4} \leftarrow (HL)_{3-0}$			
BCD adjustment	<b>ADJBA</b>		2	4	–	Decimal Adjust Accumulator after Addition	x	x	x
	<b>ADJBS</b>		2	4	–	Decimal Adjust Accumulator after Subtract	x	x	x
Bit manipulate	<b>MOV1</b>	CY, saddr.bit	3	6	7	$CY \leftarrow (\text{saddr.bit})$			x
		CY, sfr.bit	3	–	7	$CY \leftarrow \text{sfr.bit}$			x
		CY, A.bit	2	4	–	$CY \leftarrow A.\text{bit}$			x
		CY, PSW.bit	3	–	7	$CY \leftarrow \text{PSW.bit}$			x
		CY, [HL].bit	2	6	7	$CY \leftarrow (HL).\text{bit}$			x
		saddr.bit, CY	3	6	8	$(\text{saddr.bit}) \leftarrow CY$			
		sfr.bit, CY	3	–	8	$\text{sfr.bit} \leftarrow CY$			
		A.bit, CY	2	4	–	$A.\text{bit} \leftarrow CY$			
		PSW.bit, CY	3	–	8	$\text{PSW.bit} \leftarrow CY$			x
[HL].bit, CY	2	6	8	$(HL).\text{bit} \leftarrow CY$					

- 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	<b>AND1</b>	CY, saddr.bit	3	6	7	$CY \leftarrow CY \wedge (\text{saddr.bit})$			×
		CY, sfr.bit	3	–	7	$CY \leftarrow CY \wedge \text{sfr.bit}$			×
		CY, A.bit	2	4	–	$CY \leftarrow CY \wedge A.\text{bit}$			×
		CY, PSW.bit	3	–	7	$CY \leftarrow CY \wedge \text{PSW.bit}$			×
		CY, [HL].bit	2	6	7	$CY \leftarrow CY \wedge (\text{HL}).\text{bit}$			×
	<b>OR1</b>	CY, saddr.bit	3	6	7	$CY \leftarrow CY \vee (\text{saddr.bit})$			×
		CY, sfr.bit	3	–	7	$CY \leftarrow CY \vee \text{sfr.bit}$			×
		CY, A.bit	2	4	–	$CY \leftarrow CY \vee A.\text{bit}$			×
		CY, PSW.bit	3	–	7	$CY \leftarrow CY \vee \text{PSW.bit}$			×
		CY, [HL].bit	2	6	7	$CY \leftarrow CY \vee (\text{HL}).\text{bit}$			×
	<b>XOR1</b>	CY, saddr.bit	3	6	7	$CY \leftarrow CY \oplus (\text{saddr.bit})$			×
		CY, sfr.bit	3	–	7	$CY \leftarrow CY \oplus \text{sfr.bit}$			×
		CY, A.bit	2	4	–	$CY \leftarrow CY \oplus A.\text{bit}$			×
		CY, PSW. bit	3	–	7	$CY \leftarrow CY \oplus \text{PSW.bit}$			×
		CY, [HL].bit	2	6	7	$CY \leftarrow CY \oplus (\text{HL}).\text{bit}$			×
	<b>SET1</b>	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	$\text{PSW.bit} \leftarrow 1$	×	×	×
		[HL].bit	2	6	8	$(\text{HL}).\text{bit} \leftarrow 1$			
	<b>CLR1</b>	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	$\text{PSW.bit} \leftarrow 0$	×	×	×
		[HL].bit	2	6	8	$(\text{HL}).\text{bit} \leftarrow 0$			
	<b>SET1</b>	CY	1	2	–	$CY \leftarrow 1$			1
	<b>CLR1</b>	CY	1	2	–	$CY \leftarrow 0$			0
	<b>NOT1</b>	CY	1	2	–	$CY \leftarrow \overline{CY}$			×

- 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 ( $f_{\text{CPU}}$ ) selected by the processor clock control register (PCC).
  2. This clock cycle applies to the internal ROM program.

<R>

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Call/return	<b>CALL</b>	!addr16	3	7	–	$(SP - 1) \leftarrow (PC + 3)_H, (SP - 2) \leftarrow (PC + 3)_L,$ $PC \leftarrow \text{addr16}, SP \leftarrow SP - 2$			
	<b>CALLF</b>	!addr11	2	5	–	$(SP - 1) \leftarrow (PC + 2)_H, (SP - 2) \leftarrow (PC + 2)_L,$ $PC_{15-11} \leftarrow 00001, PC_{10-0} \leftarrow \text{addr11},$ $SP \leftarrow SP - 2$			
	<b>CALLT</b>	[addr5]	1	6	–	$(SP - 1) \leftarrow (PC + 1)_H, (SP - 2) \leftarrow (PC + 1)_L,$ $PC_H \leftarrow (\text{addr5} + 1), PC_L \leftarrow (\text{addr5}),$ $SP \leftarrow SP - 2$			
	<b>BRK</b>		1	6	–	$(SP - 1) \leftarrow PSW, (SP - 2) \leftarrow (PC + 1)_H,$ $(SP - 3) \leftarrow (PC + 1)_L, PC_H \leftarrow (003FH),$ $PC_L \leftarrow (003EH), SP \leftarrow SP - 3, IE \leftarrow 0$			
	<b>RET</b>		1	6	–	$PC_H \leftarrow (SP + 1), PC_L \leftarrow (SP),$ $SP \leftarrow SP + 2$			
	<b>RETI</b>		1	6	–	$PC_H \leftarrow (SP + 1), PC_L \leftarrow (SP),$ $PSW \leftarrow (SP + 2), SP \leftarrow SP + 3$	R	R	R
	<b>RETB</b>		1	6	–	$PC_H \leftarrow (SP + 1), PC_L \leftarrow (SP),$ $PSW \leftarrow (SP + 2), SP \leftarrow SP + 3$	R	R	R
Stack manipulate	<b>PUSH</b>	PSW	1	2	–	$(SP - 1) \leftarrow PSW, SP \leftarrow SP - 1$			
		rp	1	4	–	$(SP - 1) \leftarrow rp_H, (SP - 2) \leftarrow rp_L,$ $SP \leftarrow SP - 2$			
	<b>POP</b>	PSW	1	2	–	$PSW \leftarrow (SP), SP \leftarrow SP + 1$	R	R	R
		rp	1	4	–	$rp_H \leftarrow (SP + 1), rp_L \leftarrow (SP),$ $SP \leftarrow SP + 2$			
	<b>MOVW</b>	SP, #word	4	–	10	$SP \leftarrow \text{word}$			
		SP, AX	2	–	8	$SP \leftarrow AX$			
AX, SP		2	–	8	$AX \leftarrow SP$				
Unconditional branch	<b>BR</b>	!addr16	3	6	–	$PC \leftarrow \text{addr16}$			
		\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$			
		AX	2	8	–	$PCH \leftarrow A, PCL \leftarrow X$			
Conditional branch	<b>BC</b>	\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $CY = 1$			
		\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $CY = 0$			
		\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $Z = 1$			
		\$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	<b>BT</b>	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			
	<b>BF</b>	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			
	<b>BTCLR</b>	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			
	<b>DBNZ</b>	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	<b>SEL</b>	RBn	2	4	–	RBS1, 0 ← n			
	<b>NOP</b>		1	2	–	No Operation			
	<b>EI</b>		2	–	6	IE ← 1 (Enable Interrupt)			
	<b>DI</b>		2	–	6	IE ← 0 (Disable Interrupt)			
	<b>HALT</b>		2	6	–	Set HALT Mode			
	<b>STOP</b>		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.

20.3 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 <sup>Note</sup>	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 ADD ADDC SUB SUBC AND OR XOR CMP	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 ROLC	
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 <sup>Note</sup>	sfrp	saddrp	laddr16	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW <sup>Note</sup>						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
laddr16		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 21 ELECTRICAL SPECIFICATIONS

### Absolute Maximum Ratings (T<sub>A</sub> = 25°C)

Parameter	Symbol	Conditions	Ratings	Unit	
Supply voltage	V <sub>DD</sub>		−0.5 to +6.5	V	
	V <sub>SS</sub>		−0.5 to +0.3	V	
REGC pin input voltage	V <sub>IREGC</sub>		−0.5 to +3.6 and −0.5 to V <sub>DD</sub>	V	
Input voltage	V <sub>I1</sub>	P00 to P07, P10 to P17, P20 to P22, P23 <sup>Note2</sup> , P24 <sup>Note2</sup> , P25 to P27, P30 to P35 <sup>Note2</sup> , P120-P123, FLMD0	−0.3 to V <sub>DD</sub> + 0.3 <sup>Note1</sup>	V	
Output voltage	V <sub>O</sub>		−0.3 to V <sub>DD</sub> + 0.3 <sup>Note1</sup>	V	
Output current, high	I <sub>OH1</sub>	Per pin	P00 to P06, P10 to P17, P20 to P22, P23 <sup>Note2</sup> ,	−40	mA
		Total of all pins	P24 <sup>Note2</sup> , P25 to P27, P30 to P35 <sup>Note2</sup> , P120 to P122	−80	mA
	I <sub>OH2</sub>	Per pin	REM	−40	mA
Output current, low	I <sub>OL1</sub>	Per pin	P00 to P03, P07, P10 to P17, P20 to P22,	40	mA
		Total of all pins	P23 <sup>Note2</sup> , P30 to P35 <sup>Note2</sup> , P121, P122	150	mA
	I <sub>OL2</sub>	Per pin	P04 to P06, P24 <sup>Note2</sup> , P25 to P27, P120	40	mA
		Total of all pins		150	mA
Operating ambient temperature	T <sub>A</sub>		−40 to +85	°C	
Storage temperature	T <sub>stg</sub>		−65 to +150	°C	

- Notes**
1. Must be 6.5 V or lower.
  2. 38-pin products only.

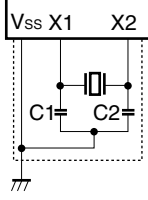
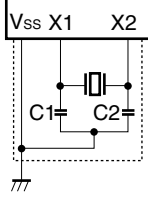
**Caution** Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that 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.



**X1 Oscillator Characteristics**

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $1.8\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

Resonator	Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Ceramic resonator		X1 clock oscillation frequency ( $f_x$ ) <sup>Note</sup>		1.0		4.0	MHz
Crystal resonator		X1 clock oscillation frequency ( $f_x$ ) <sup>Note</sup>		1.0		4.0	MHz

**Note** Indicates only oscillator characteristics. Refer to **AC Characteristics** for instruction execution time.

**Cautions 1.** When using the X1 oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

- Keep the wiring length as short as possible.
  - Do not cross the wiring with the other signal lines.
  - Do not route the wiring near a signal line through which a high fluctuating current flows.
  - Always make the ground point of the oscillator capacitor the same potential as  $V_{SS}$ .
  - Do not ground the capacitor to a ground pattern through which a high current flows.
  - Do not fetch signals from the oscillator.
2. Since the CPU is started by the 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  $+85^\circ\text{C}$ ,  $1.8\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

Resonator	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
4 MHz internal oscillator	Internal high-speed oscillation clock frequency ( $f_{RH}$ ) <sup>Note</sup>	RSTS = 1 $T_A = -10$ to $+70^\circ\text{C}$	3.92	4.0	4.08	MHz
240 kHz internal oscillator	Internal low-speed oscillation clock frequency ( $f_{RL}$ )	$2.1\text{ V} \leq V_{DD} \leq 3.6\text{ V}$	216	240	276	kHz
		$1.8\text{ V} \leq V_{DD} < 2.1\text{ V}$	180	240	300	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 (1/2)

(T<sub>A</sub> = -40 to +85°C, 1.8 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
Output current, high <sup>Note 1</sup>	I <sub>OH1</sub>	Per pin for P00 to P06, P10 to P17, P20 to P22, P23 <sup>Note2</sup> , P24 <sup>Note2</sup> , P25 to P27, P30 to P35 <sup>Note2</sup> , P120 to P122			-2.0	mA	
		Total of all pins P04 to P06, P24 <sup>Note2</sup> , P25 to P27, P120			-4.0	mA	
		Total of all pins P00 to P06, P10 to P17, P20 to P22, P23 <sup>Note2</sup> , P24 <sup>Note2</sup> , P25 to P27, P30 to P35 <sup>Note2</sup> , P121, P122			-20.0	mA	
	I <sub>OH2</sub>	P07 (REM)   V <sub>DD</sub> = 3.0 V, V <sub>OH2</sub> = 1.0 V	-6.0	-13.0	-25.0	mA	
	Total of all pins <sup>Note3</sup>					-49.0	mA
Output current, low <sup>Note 4</sup>	I <sub>OL1</sub>	P00 to P03, P07, P10 to P17, P20 to P22, P23 <sup>Note2</sup> , P30 to P35 <sup>Note2</sup> , P121, P122	Per pin			1.0	mA
			Total			20.0	mA
	I <sub>OL2</sub>	P04 to P06, P24 <sup>Note2</sup> , P25 to P27, P120	Per pin			3.0	mA
			Total			30.0	mA
	Total of all pins <sup>Note3</sup>					50.0	mA
Input voltage, high	V <sub>IH1</sub>	P07, P10 to P17, P20 to P22, P23 <sup>Note2</sup> , P30 to P35 <sup>Note2</sup> , P121, P122, P123 (KR8)	0.7V <sub>DD</sub>		V <sub>DD</sub>	V	
	V <sub>IH2</sub>	P00 to P06, P24 <sup>Note2</sup> , P25 to P27, P120, RESET/P123 (INTP5)	0.8V <sub>DD</sub>		V <sub>DD</sub>	V	
	V <sub>IH3</sub>	FLMD0	0.9V <sub>DD</sub>		V <sub>DD</sub>	V	
Input voltage, low	V <sub>IL1</sub>	P07, P10 to P17, P20 to P22, P23 <sup>Note2</sup> , P30 to P35 <sup>Note2</sup> , P121, P122, P123 (KR8)	0		0.3V <sub>DD</sub>	V	
	V <sub>IL2</sub>	P00 to P06, P24 <sup>Note2</sup> , P25 to P27, P120, RESET/P123 (INTP5)	0		0.2V <sub>DD</sub>	V	
	V <sub>IL3</sub>	FLMD0	0		0.1V <sub>DD</sub>	V	
Output voltage, high	V <sub>OH1</sub>	P00 to P06, P10 to P17, P20 to P22, P23 <sup>Note2</sup> , P24 <sup>Note2</sup> , P25 to P27, P30 to P35 <sup>Note2</sup> , P120 to P122   V <sub>DD</sub> = 1.8 V, I <sub>OH1</sub> = -1.0 mA	V <sub>DD</sub> -0.5			V	
	V <sub>OH2</sub>	P07   V <sub>DD</sub> = 3.0 V, I <sub>OH2</sub> = -6.0 mA	1.0			V	
Output voltage, low	V <sub>OL1</sub>	P00 to P03, P07, P10 to P17, P20 to P22, P23 <sup>Note2</sup> , P30 to P35 <sup>Note2</sup> , P121, P122   V <sub>DD</sub> = 1.8 V, I <sub>OL1</sub> = 0.5 mA			0.4	V	
	V <sub>OL2</sub>	P04 to P06, P24 <sup>Note2</sup> , P25 to P27, P120   V <sub>DD</sub> = 2.0 V, I <sub>OL2</sub> = 1.5 mA			0.1	V	

**Notes 1.** Value of current at which the device operation is guaranteed even if the current flows from V<sub>DD</sub> to an output pin.

**2.** 38-pin products only.

**3.** 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<sub>OH</sub> is n%: Total output current of pins = (I<sub>OH</sub> × 0.7)/(n × 0.01)

<Example> Where the duty factor is 50%, I<sub>OH</sub> = 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.

**4.** Value of current at which the device operation is guaranteed even if the current flows from an output pin to GND.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

## DC Characteristics (2/2)

 (T<sub>A</sub> = -40 to +85°C, 1.8 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit	
Input leakage current, high	I <sub>IH1</sub>	P00 to P07, P10 to P17, P20 to P22, P23 <sup>Note1</sup> , P24 <sup>Note1</sup> , P25 to P27, P30 to P35 <sup>Note1</sup> , P120, P123	V <sub>I</sub> = V <sub>DD</sub>			3	μA	
	I <sub>IH3</sub>	P121, P122 (X1, X2)	V <sub>I</sub> = V <sub>DD</sub>	I/O port mode		3	μA	
				OSC mode		20	μA	
Input leakage current, low	I <sub>IL1</sub>	P00 to P07, P10 to P17, P20 to P22, P23 <sup>Note1</sup> , P24 <sup>Note1</sup> , P25 to P27, P30 to P35 <sup>Note1</sup> , P120, P123	V <sub>I</sub> = V <sub>SS</sub>			-3	μA	
			I <sub>IL2</sub>	P121, P122 (X1, X2)	V <sub>I</sub> = V <sub>SS</sub>	I/O port mode	-3	μA
		OSC mode			-20	μA		
Pull-up resistance	R <sub>U1</sub>	V <sub>I</sub> = V <sub>SS</sub>	P00 to P07, P20 to P22, P23 <sup>Note1</sup> , P24 <sup>Note1</sup> , P25 to P27, P120 to P122		10	20	100	kΩ
	R <sub>U2</sub>			P10 to P17, P30 to P35 <sup>Note1</sup> , RESET	75	150	300	kΩ
	R <sub>U3</sub>		FLMD0	1.8 V ≤ V <sub>DD</sub> ≤ 3.6 V	10	19.5	52	kΩ
Supply current <sup>Note 2</sup>	I <sub>DD1</sub>	Operation mode	f <sub>XH</sub> = 4 MHz <sup>Note3</sup> , V <sub>DD</sub> = 3.0 V		0.8	1.2	mA	
			f <sub>RH</sub> = 4 MHz <sup>Note4</sup> , V <sub>DD</sub> = 3.0 V		0.7	1.0	mA	
	I <sub>DD2</sub>	HALT mode	f <sub>XH</sub> = 4 MHz <sup>Note3</sup> , V <sub>DD</sub> = 3.0 V		0.25	0.33	mA	
			f <sub>RH</sub> = 4 MHz <sup>Note4</sup> , V <sub>DD</sub> = 3.0 V		0.15	0.28	mA	
I <sub>DD3</sub>	STOP mode <sup>Note4</sup>	V <sub>DD</sub> = 3.0 V		1	20	μA		
Watchdog timer operating current	I <sub>WDT</sub> <sup>Note5</sup>	During 240 kHz internal low-speed oscillation clock operation			3.2	6.4	μA	
LVI operating current	I <sub>LVI</sub> <sup>Note6</sup>				5.8	12	μA	

**Notes** 1. 38-pin products only.

- Total current flowing into the internal power supply (V<sub>DD</sub>), including the peripheral operation current and the input leakage current flowing when the level of the input pin is fixed to V<sub>DD</sub> or V<sub>SS</sub>. However, the current flowing into the pull-up resistors and the output current of the port are not included.
- Not including the operating current of the 4 MHz internal oscillator and 240 kHz internal oscillator, and the current flowing into the watchdog timer and LVI circuit.
- Not including the operating current of the 240 kHz internal oscillator, and the current flowing into the watchdog timer and LVI circuit.
- Current flowing only to the watchdog timer, including the operating current of the 240 kHz internal oscillator. The current value of the μPD179F11x, 179F12x microcontroller is the sum of I<sub>DD2</sub> or I<sub>DD3</sub> and I<sub>WDT</sub> when the watchdog timer operates in the HALT or STOP mode.
- Current flowing only to the LVI circuit. The current value of the μPD179F11x, 179F12x microcontroller is the sum of I<sub>DD2</sub> or I<sub>DD3</sub> and I<sub>LVI</sub> when the LVI circuit operates in the HALT or STOP mode.

- 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

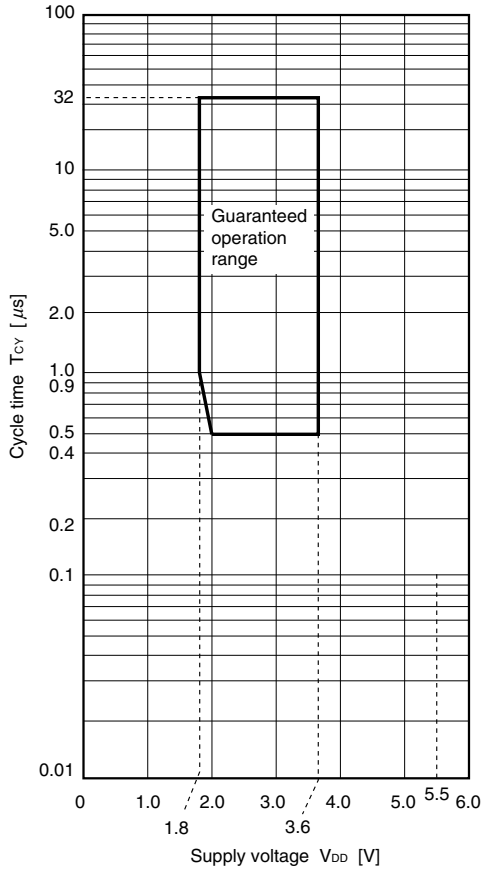
### (1) Basic operation

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $1.8\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

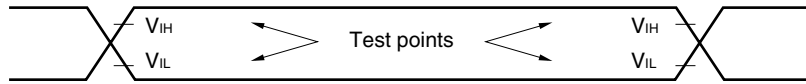
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
Instruction cycle (minimum instruction execution time)	$T_{CY}$	Main system clock ( $f_{XP}$ ) operation	$2.0\text{ V} \leq V_{DD} \leq 3.6\text{ V}$	0.5		32	$\mu\text{s}$
			$1.8\text{ V} \leq V_{DD} < 2.0\text{ V}$	1		32	$\mu\text{s}$
External main system clock frequency	$f_{EXCLK}$		1.0		4.0	MHz	
External main system clock input high-level width, low-level width	$t_{EXCLKH}$ , $t_{EXCLKL}$		$(1/f_{EXCLK} \times 1/2) - 1$			ns	
TI000, TI010 input high-level width, low-level width	$t_{TI0H}$ , $t_{TI0L}$	$2.1\text{ V} \leq V_{DD} \leq 3.6\text{ V}$	$2/f_{sam} + 0.2^{\text{Note}}$			$\mu\text{s}$	
		$1.8\text{ V} \leq V_{DD} < 2.1\text{ V}$	$2/f_{sam} + 0.5^{\text{Note}}$			$\mu\text{s}$	
TI50, TI51 input frequency	$f_{TI5}$				4	MHz	
TI50, TI51 input high-level width, low-level width	$t_{TI5H}$ , $t_{TI5L}$		250			ns	
Interrupt input high-level width, low-level width	$t_{INTH}$ , $t_{INTL}$		1			$\mu\text{s}$	
Key interrupt input low-level width	$t_{KR}$		250			ns	
RESET low-level width	$t_{RSL}$		10			$\mu\text{s}$	

**Note** Selection of  $f_{sam} = f_{PRS}$ ,  $f_{PRS}/4$ ,  $f_{PRS}/256$  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,  $f_{sam} = f_{PRS}$ .

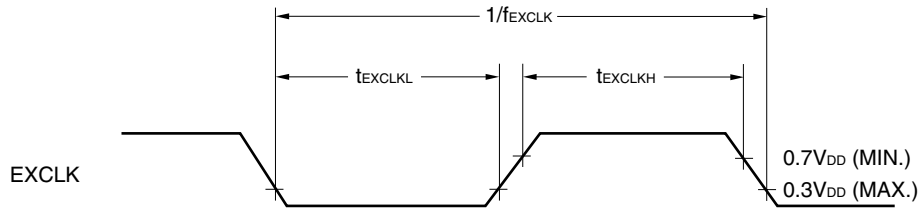
**T<sub>CY</sub> vs. V<sub>DD</sub> (Main System Clock Operation)**



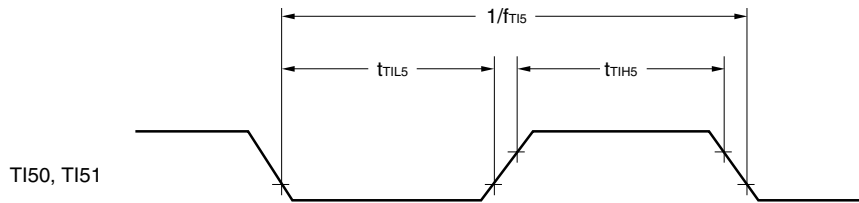
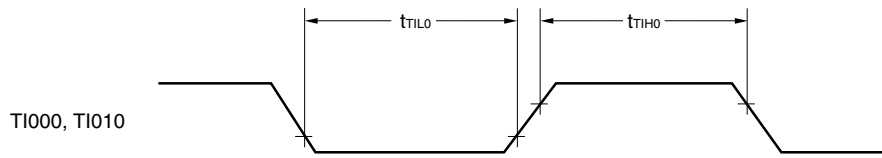
**AC Timing Test Points**



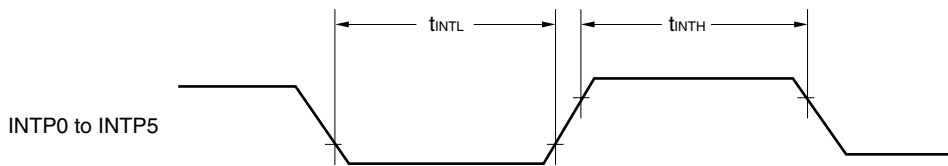
**External Main System Clock Timing, External Subsystem Clock Timing**



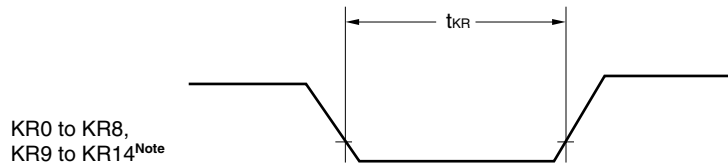
**TI Timing**



**Interrupt Request Input Timing**

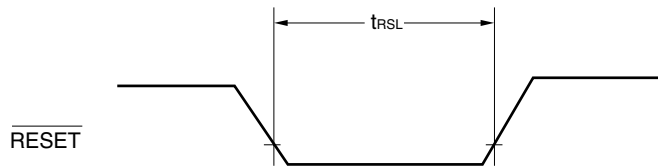


**Key Interrupt Input Timing**



**Note** 38-pin products only

**RESET Input Timing**



**(2) Serial interface**

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $1.8\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

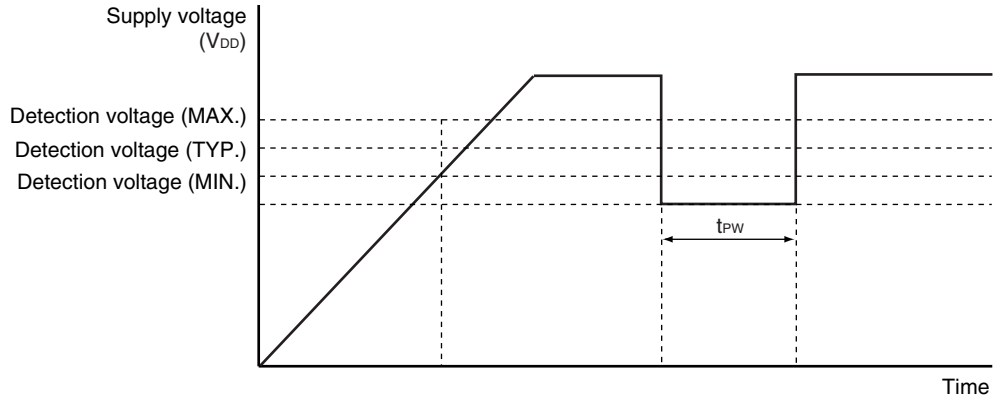
**(a) UART6 (dedicated baud rate generator output)**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate					312.5	kbps

**POC Circuit Characteristics ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $V_{SS} = 0\text{ V}$ )**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Detection voltage	$V_{POC}$		1.7	1.8	1.9	V
Minimum pulse width	$t_{PW}$		200			$\mu\text{s}$

**POC Circuit Timing**



**Note** Internal reset signal by POC circuit will be generated in case that the state of  $V_{DD} < V_{POC}$  is kept more than  $200\ \mu\text{s}$ .  $200\ \mu\text{s}$  is a guaranteed value, so it may generate internal reset signal in case of  $T_{PW} < 200\ \mu\text{s}$ . POC detection voltage may become below than operation voltage range ( $V_{DD} = 1.8$  to  $3.6\text{V}$ ), but CPU does not overrun until POC detection voltage. Please notice that it may stop the X1 oscillation before POC reset generation, if the oscillator which is not guaranteed its operation in low voltage.

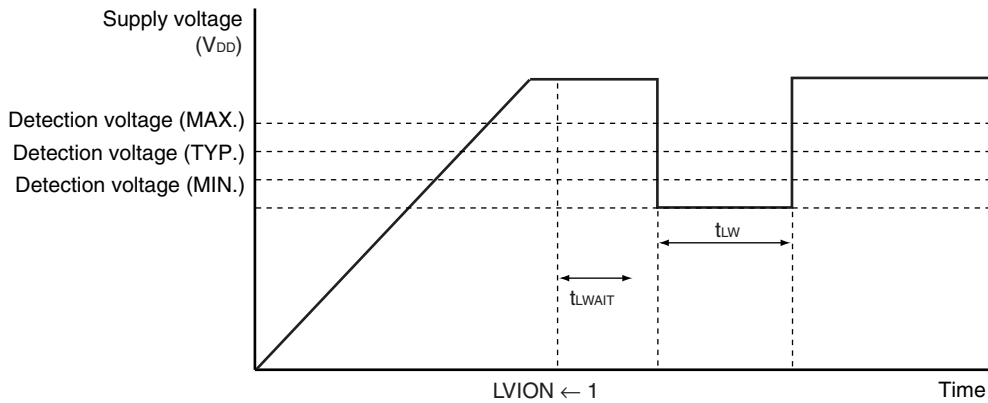
**LVI Circuit Characteristics ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $1.8\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
Detection voltage	Supply voltage level	$V_{LV15}$		3.37	3.47	3.57	V
		$V_{LV16}$		3.22	3.32	3.42	V
		$V_{LV17}$		3.06	3.16	3.26	V
		$V_{LV18}$		2.91	3.01	3.11	V
		$V_{LV19}$		2.75	2.85	2.95	V
		$V_{LV110}$		2.60	2.70	2.80	V
		$V_{LV111}$		2.45	2.55	2.65	V
		$V_{LV112}$		2.29	2.39	2.49	V
		$V_{LV113}$		2.14	2.24	2.34	V
		$V_{LV114}$		2.00	2.08	2.15	V
		$V_{LV115}$		1.83	1.93	2.03	V
External input pin <sup>Note 1</sup>	EXLVI	EXLVI < $V_{DD}$		1.21		V	
Minimum pulse width	$t_{LW}$		200			$\mu\text{s}$	
Operation stabilization wait time <sup>Note 2</sup>	$t_{LWAIT}$				10	$\mu\text{s}$	

- 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**  $V_{LV(n-1)} > V_{LVn}$ : n = 6 to 15

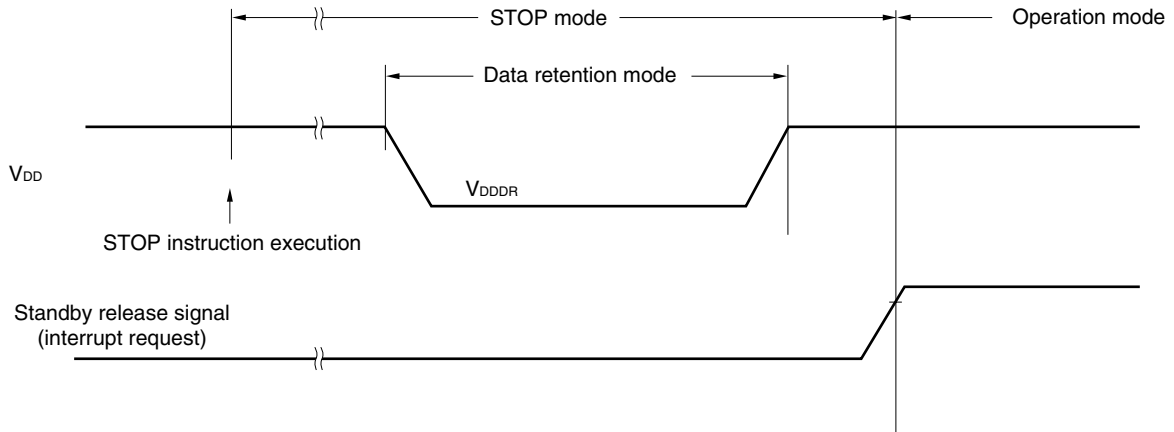
**LVI Circuit Timing**





**Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics (T<sub>A</sub> = -40 to +85°C)**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention supply voltage	V <sub>DDDR</sub>		1.3		3.6	V



**Detection Voltage for Data Memory Retention at Battery Exchange and Power Supply Decrease (T<sub>A</sub> = -40 to +85°C)**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Detection voltage for RAM retention	V <sub>LD</sub>	LVDET0 bit "1"→"0"	1.30	1.40	1.50	V

<R>

**Flash Memory Programming Characteristics (T<sub>A</sub> = -40 to +85°C, 2.0 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V)**

• **Basic characteristics**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
V <sub>DD</sub> supply current	I <sub>DD</sub>	f <sub>XP</sub> = 4 MHz (TYP.)		4.5	11.0	mA
Erase time <sup>Note 1, 2</sup>	All block	T <sub>eraca</sub>		20	200	ms
	Block unit	T <sub>erasa</sub>		20	200	ms
Write time (in 8-bit units) <sup>Note 1</sup>	T <sub>wrwa</sub>			10	100	μs
Number of rewrites per chip	C <sub>erwr</sub>	1 erase + 1 write after erase = 1 rewrite <sup>Note 3</sup>	1000			Times

**Notes 1.** Characteristic of the flash memory.

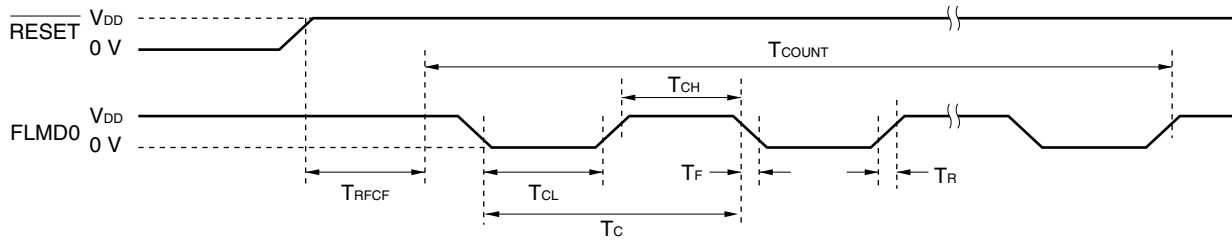
**2.** The prewrite time before erasure and the erase verify time (writeback time) are not included.

**3.** When a product is first written after shipment, "erase → write" and "write only" are both taken as one rewrite.

**Remark** f<sub>XP</sub>: Main system clock oscillation frequency

• Serial write operation characteristics

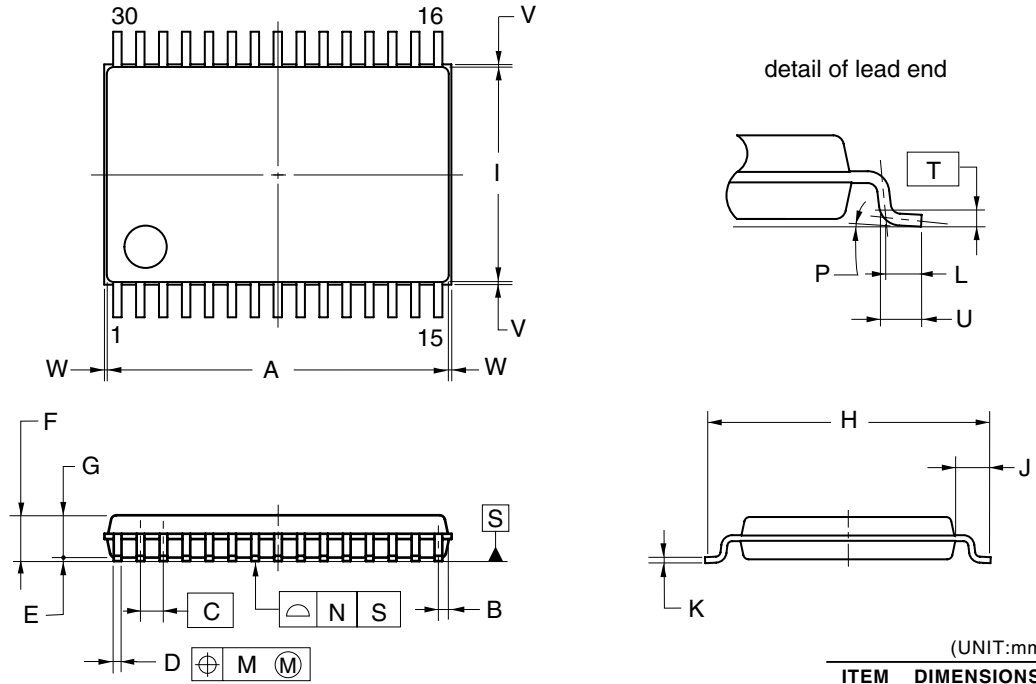
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Count start time from RESET↑ to FLMD0	$T_{RFCF}$		4.1		17.1	ms
Count execution time	$T_{COUNT}$		10.8		13.2	ms
FLMD0 counter high level width / low level width	$T_{CH}/ T_{CL}$		$T_C \times 0.45$			$\mu s$
FLMD0 counter rise time / fall time	$T_R/ T_F$		12.5			$\mu s$



## CHAPTER 22 PACKAGE DRAWINGS

- $\mu$  PD179F110MC-CAB-AX, 179F111MC-CAB-AX, 179F112MC-CAB-AX, 179F113MC-CAB-AX, 179F114MC-CAB-AX

### 30-PIN PLASTIC SSOP (7.62mm (300))



**NOTE**

Each lead centerline is located within 0.13 mm of its true position (T.P.) at maximum material condition.

(UNIT:mm)

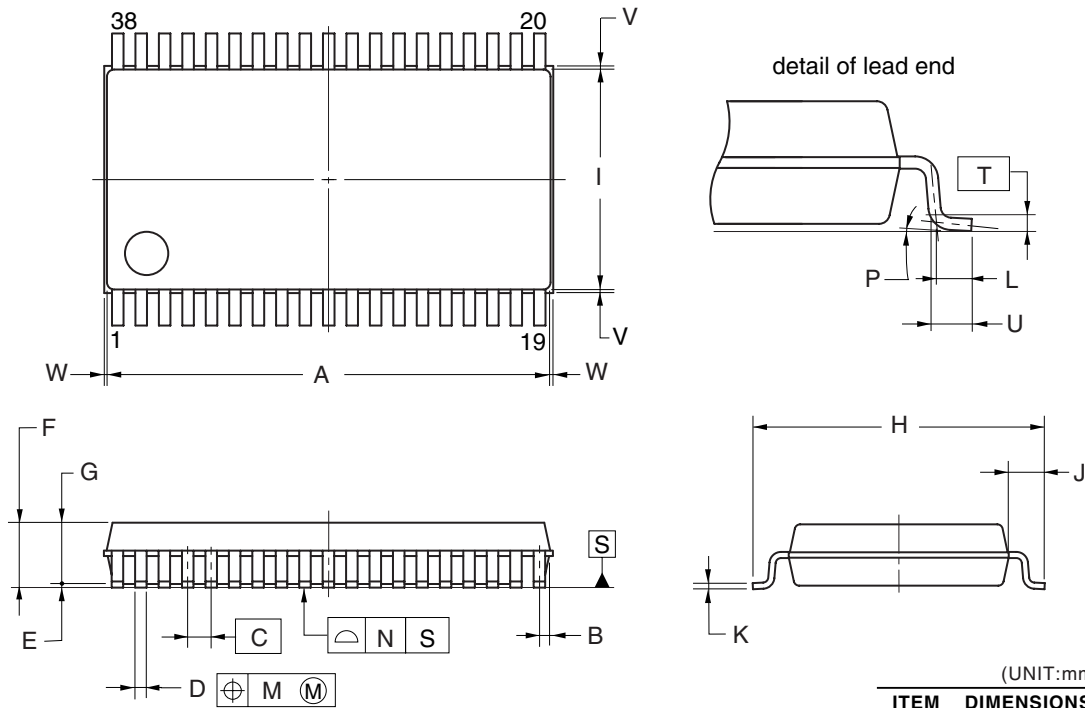
ITEM	DIMENSIONS
A	9.70±0.10
B	0.30
C	0.65 (T.P.)
D	0.22 <sup>+0.10</sup> <sub>-0.05</sub>
E	0.10±0.05
F	1.30±0.10
G	1.20
H	8.10±0.20
I	6.10±0.10
J	1.00±0.20
K	0.15 <sup>+0.05</sup> <sub>-0.01</sub>
L	0.50
M	0.13
N	0.10
P	3° <sup>+5°</sup> <sub>-3°</sub>
T	0.25(T.P.)
U	0.60±0.15
V	0.25 MAX.
W	0.15 MAX.

**P30MC-65-CAB**

© NEC Electronics Corporation 2005

- $\mu$ PD179F122MC-GAA-AX, 179F123MC-GAA-AX, 179F124MC-GAA-AX

**38-PIN PLASTIC SSOP (7.62mm (300))**



**NOTE**

Each lead centerline is located within 0.10 mm of its true position (T.P.) at maximum material condition.

(UNIT:mm)

ITEM	DIMENSIONS
A	12.30±0.10
B	0.30
C	0.65 (T.P.)
D	0.30 <sup>+0.10</sup> <sub>-0.05</sub>
E	0.125±0.075
F	2.00 MAX.
G	1.70±0.10
H	8.10±0.20
I	6.10±0.10
J	1.00±0.20
K	0.15 <sup>+0.10</sup> <sub>-0.05</sub>
L	0.50
M	0.10
N	0.10
P	3° <sup>+5°</sup> <sub>-3°</sub>
T	0.25(T.P.)
U	0.60±0.15
V	0.25 MAX.
W	0.15 MAX.

**P38MC-65-GAA**

## APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for the development of systems that employ the  $\mu$ PD179F11x, 179F12x microcontrollers.

Figure A-1 shows the development tool configuration.

- **Support for PC98-NX series**

Unless otherwise specified, products supported by IBM PC/AT™ compatibles are compatible with PC98-NX series computers. When using PC98-NX series computers, refer to the explanation for IBM PC/AT compatibles.

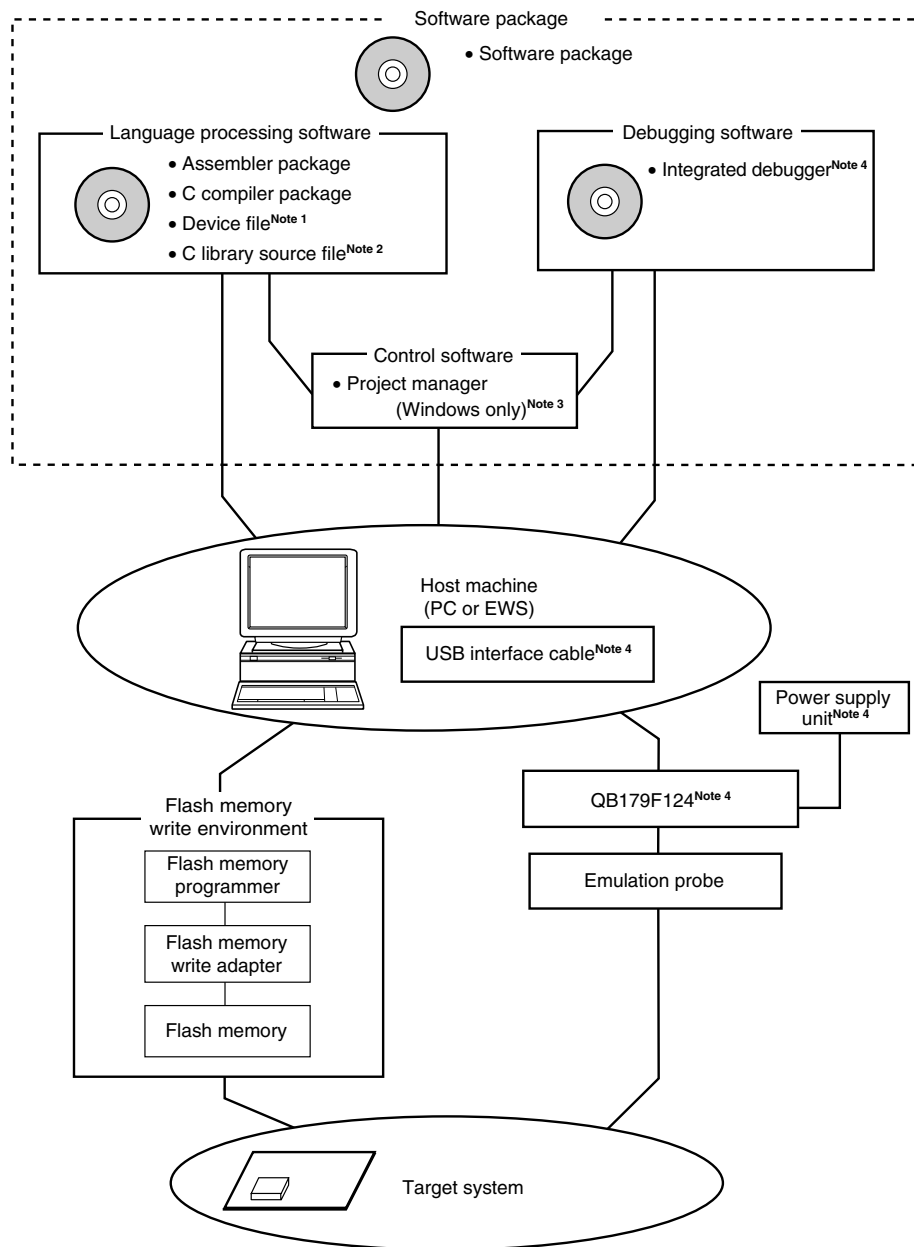
- **Windows™**

Unless otherwise specified, “Windows” means the following OSs.

- Windows 98
- Windows NT™
- Windows 2000
- Windows XP

Figure A-1. Development Tool Configuration (1/2)

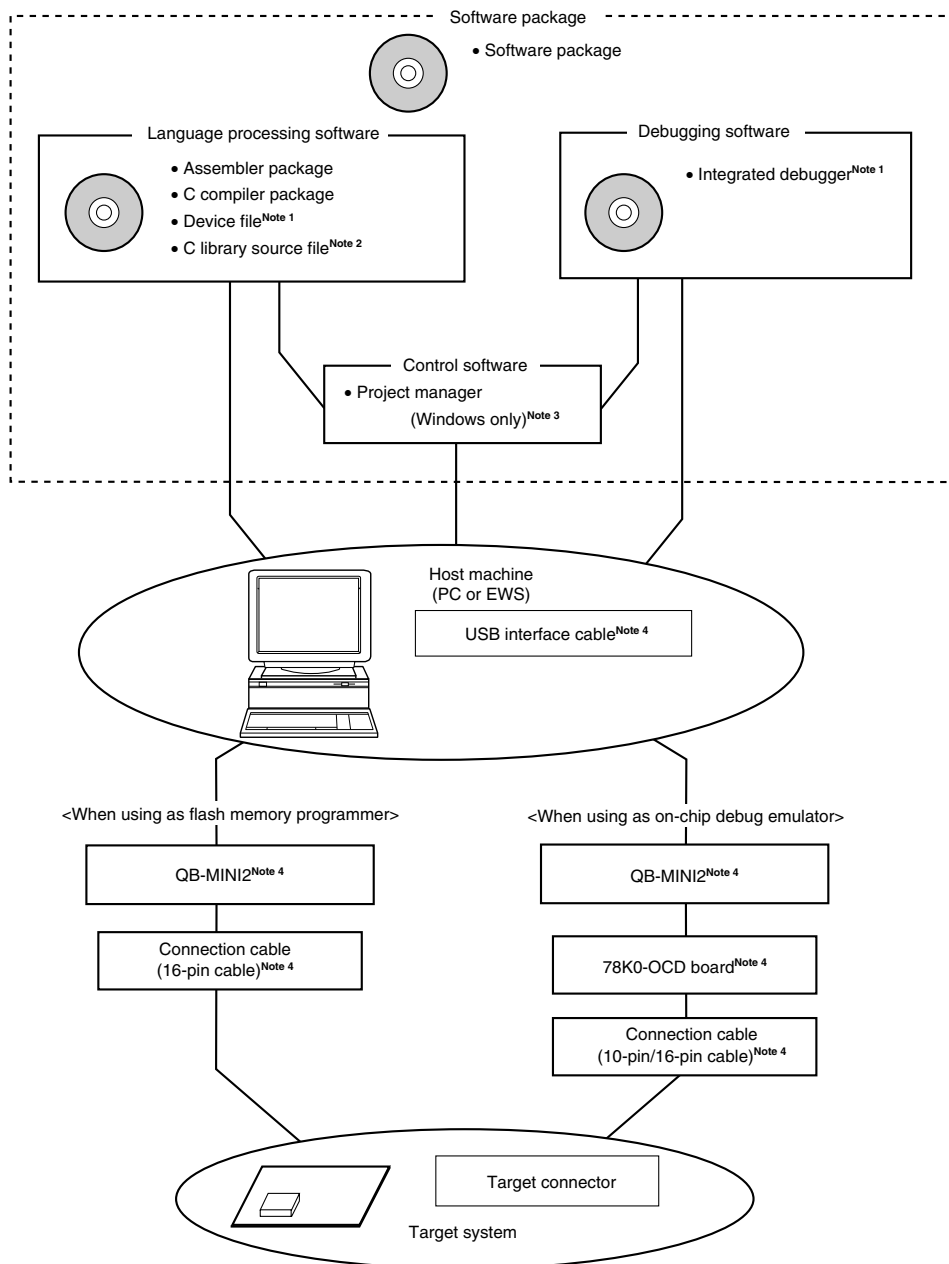
## (1) When using the in-circuit emulator QB-179F124



- Notes**
1. Download the device file (DF179124) for the  $\mu$ PD179F11x, 179F12x microcontrollers from the download site for development tools (<http://www.necel.com/micro/ods/eng/index.html>).
  2. The C library source file is not included in the software package.
  3. The project manager PM+ is included in the assembler package. The PM+ is only used for Windows.
  4. The QB-179F124 is supplied with the integrated debugger ID78K0-QB, a USB interface cable, a power supply unit, the on-chip debug emulator QB-MINI2, connection cables (10-pin and 16-pin cables), and the 78K0-OCD board. Any other products are sold separately. Download the software for operating the QB-MINI2 from the download site for development tools (<http://www.necel.com/micro/ods/eng/index.html>) when using the QB-MINI2.

Figure A-1. Development Tool Configuration (2/2)

## (2) When using the on-chip debug emulator with programming function QB-MINI2



- Notes**
1. Download the device file (DF179124) for the  $\mu$ PD179F11x, 179F12x microcontrollers and the integrated debugger ID78K0-QB from the download site for development tools (<http://www.necel.com/micro/ods/eng/index.html>).
  2. The C library source file is not included in the software package.
  3. The project manager PM+ is included in the assembler package. The PM+ is only used for Windows.
  4. The on-chip debug emulator QB-MINI2 is supplied with a USB interface cable, connection cables (10-pin and 16-pin cables), and the 78K0-OCD board. Any other products are sold separately. Download the software for operating the QB-MINI2 from the download site for development tools (<http://www.necel.com/micro/ods/eng/index.html>).

## A.1 Software Package

SP78K0 78K0 microcontroller software package	Development tools (software) common to the 78K0 microcontrollers and $\mu$ PD179F11x, 179F12x microcontrollers are combined in this package.
	Part number: $\mu$ SxxxxSP78K0

**Remark** xxxx in the part number differs depending on the host machine and OS used.

$\mu$ SxxxxSP78K0

xxxx	Host Machine	OS	Supply Medium
AB17	PC-9800 series,	Windows (Japanese version)	CD-ROM
BB17	IBM PC/AT compatibles	Windows (English version)	

## A.2 Language Processing Software

RA78K0 Assembler package	This assembler converts programs written in mnemonics into object codes executable with a microcontroller. This assembler is also provided with functions capable of automatically creating symbol tables and branch instruction optimization. This assembler should be used in combination with a device file (DF179124) (sold separately). <b>&lt;Precaution when using RA78K0 in PC environment&gt;</b> This assembler package is a DOS-based application. It can also be used in Windows, however, by using the Project Manager (included in assembler package) on Windows.
	Part number: $\mu$ SxxxxRA78K0
CC78K0 C compiler package	This compiler converts programs written in C language into object codes executable with a microcontroller. This compiler should be used in combination with an assembler package and device file (both sold separately). <b>&lt;Precaution when using CC78K0 in PC environment&gt;</b> This C compiler package is a DOS-based application. It can also be used in Windows, however, by using the Project Manager (included in assembler package) on Windows.
	Part number: $\mu$ SxxxxCC78K0
DF179124 <sup>Note 1</sup> Device file	This file contains information peculiar to the device. This device file should be used in combination with a tool (RA78K0, CC78K0, and ID78K0-QB) (all sold separately). The corresponding OS and host machine differ depending on the tool to be used.
	Part number: $\mu$ SxxxxDF179124
CC78K0-L <sup>Note 2</sup> C library source file	This is a source file of the functions that configure the object library included in the C compiler package. This file is required to match the object library included in the C compiler package to the user's specifications.
	Part number: $\mu$ SxxxxCC78K0-L

- Notes**
1. The DF179124 can be used in common with the RA78K0, CC78K0, and ID78K0-QB. Download the DF179124 from the download site for development tools (<http://www.necel.com/micro/ods/eng/index.html>).
  2. The CC78K0-L is not included in the software package (SP78K0).



**Remark** xxxx in the part number differs depending on the host machine and OS used.

μSxxxxRA78K0

μSxxxxCC78K0

μSxxxxCC78K0-L

xxxx	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	
3P17	HP9000 series 700™	HP-UX™ (Rel. 10.10)	
3K17	SPARCstation™	SunOS™ (Rel. 4.1.4) Solaris™ (Rel. 2.5.1)	

μSxxxxDF179124

xxxx	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	3.5-inch 2HD FD
BB13		Windows (English version)	

### A.3 Control Software

PM+ Project manager	This is control software designed to enable efficient user program development in the Windows environment. All operations used in development of a user program, such as starting the editor, building, and starting the debugger, can be performed from the project manager. <b>&lt;Caution&gt;</b> The project manager is included in the assembler package (RA78K0). It can only be used in Windows.
------------------------	---

## A.4 Flash Memory Writing Tools

### <R> A.4.1 When using flash memory programmer PG-FP4, FL-PR4, PG-FP5 and FL-PR5

PG-FP4, FL-PR4, PG-FP5, FL-PR5 Flash memory programmer	Flash memory programmer dedicated to microcontrollers with on-chip flash memory.
FA-179F114MC-CAB-MX FA-179F124MC-GAA-MX Flash memory writing adapter	Flash memory writing adapter used connected to the flash memory programmer. <ul style="list-style-type: none"> <li>FA-179F114MC-CAB-MX: For 30-pin plastic SSOP (MC-CAB type)</li> <li>FA-179F124MC-GAA-MX: For 38-pin plastic SSOP (MC-GAA type)</li> </ul>

**Remarks 1.** FL-PR4, FL-PR5, FA-179F114MC-CAB-MX, and FA-179F124MC-GAA-MX are products of Naito Densai Machida Mfg. Co., Ltd.

TEL: +81-42-750-4172 Naito Densai Machida Mfg. Co., Ltd.

**2.** Use the latest version of the flash memory programming adapter.

### A.4.2 When using on-chip debug emulator with programming function QB-MINI2

QB-MINI2 On-chip debug emulator with programming function	This is a flash memory programmer dedicated to microcontrollers with on-chip flash memory. It is available also as on-chip debug emulator which serves to debug hardware and software when developing application systems using the $\mu$ PD179F11x, 179F12x microcontrollers. When using this as flash memory programmer, it should be used in combination with a connection cable (16-pin cable) and a USB interface cable that is used to connect the host machine.
Target connector specifications	16-pin general-purpose connector (2.54 mm pitch)

**Remarks 1.** The QB-MINI2 is supplied with a USB interface cable, connection cables (10-pin and 16-pin cables), and the 78K0-OCD board. The connection cable (10-pin cable) and the 78K0-OCD board are used only when using the on-chip debug function.

**2.** Download the software for operating the QB-MINI2 from the download site for development tools (<http://www.necel.com/micro/ods/eng/index.html>).

## A.5 Debugging Tools (Hardware)

### A.5.1 When using in-circuit emulator QB-179F124

QB-179F124 In-circuit emulator	This in-circuit emulator serves to debug hardware and software when developing application systems using the $\mu$ PD179F11x, 179F12x microcontrollers. It supports to the integrated debugger (ID78K0-QB). This emulator should be used in combination with a power supply unit and emulation probe, and the USB is used to connect this emulator to the host machine.
QB-144-CA-01 Check pin adapter	This check pin adapter is used in waveform monitoring using the oscilloscope, etc.
QB-80-EP-01T Emulation probe	This emulation probe is flexible type and used to connect the in-circuit emulator and target system.
QB-30MC-EA-02T, QB-38MC-EA-02T Exchange adapter	This exchange adapter is used to perform pin conversion from the in-circuit emulator to target connector. <ul style="list-style-type: none"> <li>• QB-30MC-EA-02T: For 30-pin plastic SSOP (MC-CAB type)</li> <li>• QB-38MC-EA-02T: For 38-pin plastic SSOP (MC-GAA type)</li> </ul>
QB-30MC-YS-01T, QB-38MC-YS-01T Space adapter	This space adapter is used to adjust the height between the target system and in-circuit emulator. <ul style="list-style-type: none"> <li>• QB-30MC-YS-01T: For 30-pin plastic SSOP (MC-CAB type)</li> <li>• QB-38MC-YS-01T: For 38-pin plastic SSOP (MC-GAA type)</li> </ul>
QB-30MC-YQ-01T, QB-38MC-YQ-01T YQ connector	This YQ connector is used to connect the target connector and exchange adapter. <ul style="list-style-type: none"> <li>• QB-30MC-YQ-01T: For 30-pin plastic SSOP (MC-CAB type)</li> <li>• QB-38MC-YQ-01T: For 38-pin plastic SSOP (MC-GAA type)</li> </ul>
QB-30MC-HQ-01T, QB-38MC-HQ-01T Mount adapter	This mount adapter is used to mount the target device with socket. <ul style="list-style-type: none"> <li>• QB-30MC-HQ-01T: For 30-pin plastic SSOP (MC-CAB type)</li> <li>• QB-38MC-HQ-01T: For 38-pin plastic SSOP (MC-GAA type)</li> </ul>
QB-30MC-NQ-01T, QB-38MC-NQ-01T Target connector	This target connector is used to mount on the target system. <ul style="list-style-type: none"> <li>• QB-30MC-NQ-01T: For 30-pin plastic SSOP (MC-CAB type)</li> <li>• QB-38MC-NQ-01T: For 38-pin plastic SSOP (MC-GAA type)</li> </ul>

**Remarks 1.** The QB-179F124 is supplied with the integrated debugger ID78K0-QB, a USB interface cable, a power supply unit, the on-chip debug emulator QB-MINI2, connection cables (10-pin and 16-pin cables), and the 78K0-OCD board.

Download the software for operating the QB-MINI2 from the download site for development tools (<http://www.necel.com/micro/ods/eng/index.html>) when using the QB-MINI2.

**2.** The packed contents differ depending on the part number, as follows.

Packed Contents Part Number	In-Circuit Emulator	Emulation Probe	Exchange Adapter	YQ Connector	Target Connector
QB-179F124-ZZZ	QB-179F124	None			
QB-179F124-T30MC		QB-80-EP-01T	QB-30MC-EA-02T	QB-30MC-YQ-01T	QB-30MC-NQ-01T
QB-179F124-T38MC			QB-38MC-EA-02T	QB-38MC-YQ-01T	QB-38MC-NQ-01T

**A.5.2 When using on-chip debug emulator with programming function QB-MINI2**

QB-MINI2 On-chip debug emulator with programming function	This on-chip debug emulator serves to debug hardware and software when developing application systems using the $\mu$ PD179F11x, 179F12x microcontrollers. It is available also as flash memory programmer dedicated to microcontrollers with on-chip flash memory. When using this as on-chip debug emulator, it should be used in combination with a connection cable (10-pin cable or 16-pin cable), a USB interface cable that is used to connect the host machine, and the 78K0-OCD board.
Target connector specifications	10-pin general-purpose connector (2.54 mm pitch) or 16-pin general-purpose connector (2.54 mm pitch)

- Remarks**
1. The QB-MINI2 is supplied with a USB interface cable, connection cables (10-pin and 16-pin cables), and the 78K0-OCD board. The connection cable (10-pin cable) and the 78K0-OCD board are used only when using the on-chip debug function.
  2. Download the software for operating the QB-MINI2 from the download site for development tools (<http://www.necel.com/micro/ods/eng/index.html>).

**A.6 Debugging Tools (Software)**

ID78K0-QB Integrated debugger	This debugger supports the in-circuit emulators for the 78K0 microcontrollers and $\mu$ PD179F11x, 179F12x microcontrollers. The ID78K0-QB is a Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. It should be used in combination with the device file (sold separately).
	Part number: $\mu$ SxxxxID78K0-QB

**Remark** xxxx in the part number differs depending on the host machine and OS used.

$\mu$ SxxxxID78K0-QB

xxxx	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	

## APPENDIX B REVISION HISTORY

### B.1 Major Revisions in This Edition

(1/2)

Page	Description
<b>CHAPTER 1 OUTLINE</b>	
p. 16	Modification of <b>1.3 Ordering Information</b>
p. 20	Modification of Internal low-speed oscillation clock (for WDT) in <b>1.8 Outline of Functions</b>
<b>CHAPTER 2 PIN FUNCTIONS</b>	
p. 28	Modification of <b>Table 2-1. Pin I/O Circuit Types</b>
<b>CHAPTER 3 CPU ARCHITECTURE</b>	
p. 55	Modification of <b>3.3.3 Table indirect addressing</b>
<b>CHAPTER 4 PORT FUCTIONS</b>	
p. 68	Modification of <b>Figure 4-2. Block Diagram of P00 and P03</b>
p. 83	Modification of <b>4.3 (3) Pull-up resistor option registers (PU0 to PU2, PU3, and PU12)</b>
p. 86	Modification of <b>4.5 Settings of Port Mode Register, Output Latch, Pull-up Resistor Option Register, and Port Output Mode Register When Using Alternate Function</b>
<b>CHAPTER 5 CLOCK GENERATOR</b>	
p. 88	Modification of <b>(2) Internal low-speed oscillation clock (clock for watchdog timer) in 5.1 Functions of Clock Generator.</b>
p. 90	Modification of <b>Figure 5-1. Block Diagram of Clock Generator</b>
p. 92	Deletion of <b>Caution 1</b> from <b>Figure 5-2. Format of Clock Operation Mode Select Register (OSCCTL)</b>
p. 99	Addition of <b>Caution</b> to <b>Figure 5-9. Example of External Circuit of X1 Oscillator</b>
p. 101	Modification of <b>5.4.3 Internal low-speed oscillator</b>
p. 107	Addition of <b>5.6.3 Example of controlling internal low-speed oscillation clock</b>
p. 111	Addition of <b>Caution</b> to <b>Table 5-4. CPU Clock Transition and SFR Register Setting Examples (2/2)</b>
<b>CHAPTER 8 8-BIT TIMERS H0 AND H1</b>	
p. 205	Modification of <b>Figure 8-6. Format of 8-bit Timer H Mode Register 1 (TMHMD1)</b>
<b>CHAPTER 9 WATCHDOG TIMER</b>	
p. 228	Modification of <b>9.4.1 Controlling operation of watchdog timer</b> in <b>Caution 5</b>
p. 228	Modification of <b>Table 9-3. Setting of Overflow Time of Watchdog Timer (when <math>2.1\text{ V} \leq V_{DD} \leq 3.6\text{ V}</math>)</b>
p. 229	Addition of <b>Table 9-4. Setting of Overflow Time of Watchdog Timer (when <math>1.8\text{ V} \leq V_{DD} &lt; 2.1\text{ V}</math>)</b>
p. 230	Modification of <b>Caution</b> and <b>Remark</b> of <b>Table 9-5. Setting Window Open Period of Watchdog Timer</b>
<b>CHAPTER 13 STANDBY FUNCTION</b>	
p. 283	Deletion of <b>Caution 1</b> from <b>13.1.1 Standby function</b>
p. 284	Modification of <b>Figure 13-1. Format of Oscillation Stabilization Time Counter Status Register (OSTC)</b>
p. 285	Modification of <b>Figure 13-2. Format of Oscillation Stabilization Time Select Register (OSTS)</b>
p. 290	Addition of <b>Caution 4</b> to <b>Table 13-3. Operating Statuses in STOP Mode</b>
<b>CHAPTER 17 OPTION BYTE</b>	
p. 324	Modification of <b>Figure 17-1. Format of Option Byte (1/2)</b>

Page	Description
<b>CHAPTER 18 FLASH MEMORY</b>	
p. 327	Modification of <b>Note of Table 18-2. Example of wiring Between <math>\mu</math>PD179F11x microcontroller and Dedicated Flash Memory Programmer</b>
p. 328	Modification of <b>Remark of Figure 18-2. Example of Wiring Adapter for Flash Memory Writing (30-Pin Products)</b>
p. 329	Modification of <b>Figure 18-3. Environment for Writing Program to Flash Memory</b>
p. 329	Modification of <b>Figure 18-4. Communication with Dedicated Flash Memory Programmer</b>
p. 330	Modification of <b>Note of Table 18-3. Pin Connection</b>
p. 333	Modification of <b>18.5.6 Other signal pins</b>
p. 334	Modification of <b>Figure 18-9. Flash Memory Manipulation Procedure</b>
p. 335	Modification of <b>Figure 18-11. Communication Commands</b>
p. 336	Modification of <b>Table 18-7. Flash Memory Control Commands</b>
p. 337	Modification of <b>18.7 Security Settings</b>
p. 337	Modification of <b>Table 18-9. Relationship Between Enabling Security Function and Command</b>
p. 338	Modification of <b>Table 18-10. Setting Security in Each Programming Mode</b>
<b>CHAPTER 20 INSTRUCTION SET</b>	
p. 353	Modification of <b>20.2 Operation List</b>
<b>CHAPTER 21 ELECTRICAL SPECIFICATIONS</b>	
p. 367	Modification of <b>Flash Memory Programming Characteristics (<math>T_A = -40</math> to <math>+85^\circ\text{C}</math>, <math>2.0\text{ V} \leq V_{DD} \leq 3.6\text{ V}</math>, <math>V_{SS} = 0\text{ V}</math>)</b>
<b>APPENDIX A DEVELOPMENT TOOLS</b>	
p. 376	Modification of <b>A.4.1 When using flash memory programmer PG-FP4, FL-PR4, PG-FP5 and FL-PR5</b>
<b>APPENDIX B REVISION HISTORY</b>	
p. 380	Addition of <b>B.2 Revision History up to Previous Editions</b>

<R>

## B.2 Revision History up to Previous Editions

Here is the revision history of the preceding editions. Chapter indicates the chapter of each edition.

Edition	Description	Applied to:
2nd edition	Addition of description to <b>2.2.5 P120 to P123 (port 12)</b>	<b>CHAPTER 2 PIN FUNCTIONS</b>
	Modification of <b>Table 3-6. Special Function Register List</b>	<b>CHAPTER 3 CPU ARCHITECTURE</b>
	Modification of <b>Table 4-2. Port Configuration</b>	<b>CHAPTER 4 PORT FUNCTIONS</b>
	Modification of <b>Figure 4-14. Block Diagram of P123</b>	
	Addition of description to <b>4.3 Registers Controlling Port Function</b>	
	Modification of <b>Figure 4-17. Format of Pull-up Resistor Option Register</b>	
	Addition of <b>4.3 (5) Reset pin mode register (RSTMASK)</b>	
	Addition of description to <b>12.1 Functions of Key Interrupt</b>	<b>CHAPTER 12 KEY INTERRUPT FUNCTION</b>
	Modification of <b>Table 14-2. Hardware Statuses After Reset Acknowledgment</b>	<b>CHAPTER 14 RESET FUNCTIONS</b>
	Addition of description to <b>16.1 Functions of Low-Voltage Detector</b>	<b>CHAPTER 16 LOW-VOLTAGE DETECTOR</b>
	Addition of <b>16.6 RAM Data Retention Detector</b>	
	Modification of <b>Table 18-7. Flash Memory Control Commands</b>	<b>CHAPTER 18 FLASH MEMORY</b>
	Modification of <b>18.7 Security Settings</b>	
	Addition of <b>18.8 Flash Memory Programming by Self-Programming</b>	
	Modification of <b>Figure 19-1. Connection Example of QB-MINI2 and <math>\mu</math>PD179F11x, 179F12x microcontrollers (When OCD0A/X1 and OCD0B/X2 Are Used)</b>	<b>CHAPTER 19 ON-CHIP DEBUG FUNCTION</b>
	Modification of <b>Figure 19-2. Connection Example of QB-MINI2 and <math>\mu</math>PD179F11x, 179F12x microcontroller (When OCD1A/P04 and OCD1B/P05 Are Used)</b>	
Modification of <b>CHAPTER 21 ELECTRICAL SPECIFICATIONS</b>	<b>CHAPTER 21 ELECTRICAL SPECIFICATIONS</b>	

*For further information,  
please contact:*

**NEC Electronics Corporation**  
1753, Shimonumabe, Nakahara-ku,  
Kawasaki, Kanagawa 211-8668,  
Japan  
Tel: 044-435-5111  
<http://www.necel.com/>

**[America]**

**NEC Electronics America, Inc.**  
2880 Scott Blvd.  
Santa Clara, CA 95050-2554, U.S.A.  
Tel: 408-588-6000  
800-366-9782  
<http://www.am.necel.com/>

**[Europe]**

**NEC Electronics (Europe) GmbH**  
Arcadiastrasse 10  
40472 Düsseldorf, Germany  
Tel: 0211-65030  
<http://www.eu.necel.com/>

**Hanover Office**

Podbielskistrasse 166 B  
30177 Hannover  
Tel: 0 511 33 40 2-0

**Munich Office**

Werner-Eckert-Strasse 9  
81829 München  
Tel: 0 89 92 10 03-0

**Stuttgart Office**

Industriestrasse 3  
70565 Stuttgart  
Tel: 0 711 99 01 0-0

**United Kingdom Branch**

Cygnus House, Sunrise Parkway  
Linford Wood, Milton Keynes  
MK14 6NP, U.K.  
Tel: 01908-691-133

**Succursale Française**

9, rue Paul Dautier, B.P. 52  
78142 Velizy-Villacoublay Cédex  
France  
Tel: 01-3067-5800

**Sucursal en España**

Juan Esplandiu, 15  
28007 Madrid, Spain  
Tel: 091-504-2787

**Tyskland Filial**

Täby Centrum  
Entrance S (7th floor)  
18322 Täby, Sweden  
Tel: 08 638 72 00

**Filiale Italiana**

Via Fabio Filzi, 25/A  
20124 Milano, Italy  
Tel: 02-667541

**Branch The Netherlands**

Steijgerweg 6  
5616 HS Eindhoven  
The Netherlands  
Tel: 040 265 40 10

**[Asia & Oceania]**

**NEC Electronics (China) Co., Ltd**  
7th Floor, Quantum Plaza, No. 27 ZhiChunLu Haidian  
District, Beijing 100083, P.R.China  
Tel: 010-8235-1155  
<http://www.cn.necel.com/>

**Shanghai Branch**

Room 2509-2510, Bank of China Tower,  
200 Yincheng Road Central,  
Pudong New Area, Shanghai, P.R.China P.C:200120  
Tel:021-5888-5400  
<http://www.cn.necel.com/>

**Shenzhen Branch**

Unit 01, 39/F, Excellence Times Square Building,  
No. 4068 Yi Tian Road, Futian District, Shenzhen,  
P.R.China P.C:518048  
Tel:0755-8282-9800  
<http://www.cn.necel.com/>

**NEC Electronics Hong Kong Ltd.**

Unit 1601-1613, 16/F., Tower 2, Grand Century Place,  
193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong  
Tel: 2886-9318  
<http://www.hk.necel.com/>

**NEC Electronics Taiwan Ltd.**

7F, No. 363 Fu Shing North Road  
Taipei, Taiwan, R. O. C.  
Tel: 02-8175-9600  
<http://www.tw.necel.com/>

**NEC Electronics Singapore Pte. Ltd.**

238A Thomson Road,  
#12-08 Novena Square,  
Singapore 307684  
Tel: 6253-8311  
<http://www.sg.necel.com/>

**NEC Electronics Korea Ltd.**

11F., Samik Lavied'or Bldg., 720-2,  
Yeoksam-Dong, Kangnam-Ku,  
Seoul, 135-080, Korea  
Tel: 02-558-3737  
<http://www.kr.necel.com/>