User's Manual

8

μPD78F0730

User's Manual: Hardware

8-Bit Single-Chip Microcontroller

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

Renesas Electronics
www.renesas.com

Rev.3.00 Sep 2011

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.

NOTES FOR CMOS DEVICES

(1) VOLTAGE APPLICATION WAVEFORM AT INPUT PIN: Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between VIL (MAX) and VIH (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between VIL (MAX) and VIH (MIN).

(2) HANDLING OF UNUSED INPUT PINS: Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to VDD or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

(3) PRECAUTION AGAINST ESD: A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

(4) STATUS BEFORE INITIALIZATION: Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.

(5) POWER ON/OFF SEQUENCE: In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current. The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

(6) INPUT OF SIGNAL DURING POWER OFF STATE : Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

How to Use This Manual

Readers

This manual is intended for user engineers who wish to understand the functions of the μPD78F0730 and design and develop application systems and programs for this device.
The target product is as follows.

μPD78F0730

Purpose

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

Organization

The μPD78F0730 manual is separated into two parts: this manual and the instructions edition (common to the 78K/0 Series).

μPD78F0730 User's Manual (This Manual)	**78K/0 Series User's Manual Instructions**
• 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 check the details of a register when you know the register name:
  → See **APPENDIX C REGISTER INDEX**.
- To know details of the 78K/0 Series 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:	$\overline{\times\times\times}$ (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	···×××× or ××××B
	Decimal	···××××
	Hexadecimal	···××××H

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.
μPD78F0730 User's Manual	This manual
78K/0 Series Instructions User's Manual	U12326E
78K0/Kx2 Flash Memory Programming (Programmer) Application Note	U17739E
78K0/Kx2 Flash Memory Self Programming User's Manual	U17516E
78K0/Kx2 EEPROM™ Emulation Application Note	U17517E
78K0 Microcontrollers Self Programming Library Type01 User's Manual	U18274E
78K0 Microcontrollers EEPROM Emulation Library Type01 User's Manual	U18275E

Documents Related to Flash Memory Programming

Document Name	Document No.
PG-FP5 Flash Memory Programmer User's Manual	R20UT0008E
QB-MINI2 On-Chip Debug Emulator with Programming Function User's Manual	U18371E
QB-Programmer Programming GUI User's Manual	U18527E

Documents Related to Development Tools (Hardware)

Document Name	Document No.
QB-780731 In-Circuit Emulator User's Manual	U17804E
QB-MINI2 On-Chip Debug Emulator with Programming Function User's Manual	U18371E

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

Documents Related to Development Tools (Software)

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

Notes 1. This document is installed into the PC together with the tool when installing RA78K0 Ver. 4.01. For descriptions not included in "78K0 Assembler Package RA78K0 Ver. 4.01 Operating Precautions", refer to the user's manual of RA78K0 Ver. 3.80.

2. This document is installed into the PC together with the tool when installing CC78K0 Ver. 4.00. For descriptions not included in "78K0 C Compiler CC78K0 Ver. 4.00 Operating Precautions", refer to the user's manual of CC78K0 Ver. 3.70.

3. PM plus Ver. 5.20 is the integrated development environment included with RA78K0 Ver. 3.80.

4. PM+ Ver. 6.30 is the integrated development environment included with RA78K0 Ver. 4.01. Software tool (assembler, C compiler, debugger, and simulator) products of different versions can be managed.

Other Documents

Document Name	Document No.
RENESAS MICROCOMPUTER GENERAL CATALOG	R01CS0001E
78K MICROCONTROLLERS SELECTION GUIDE	U17652E
Semiconductor Device Mount Manual	**Note**
Quality Grades on NEC Semiconductor Devices	C11531E
NEC Semiconductor Device Reliability/Quality Control System	C10983E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E

Note See the "Semiconductor Device Mount Manual" website (http://www2.renesas.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.

All trademarks and registered trademarks are the property of their respective owners.
EEPROM is a trademark of Renesas Electronics Corporation.
Windows and Windows NT are registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
SuperFlash is a registered trademark of Silicon Storage Technology, Inc. in several countries including the United States and Japan.

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

CONTENTS

CHAPTER 1 OUTLINE 15

1.1 Features 15
1.2 Applications 15
1.3 Ordering Information 16
1.4 Pin Configuration (Top View) 16
1.5 Block Diagram 18
1.6 Outline of Functions 19

CHAPTER 2 PIN FUNCTIONS 21

2.1 Pin Function List 21
2.2 Description of Pin Functions 23
2.2.1 P00 and P01 (port 0) 23
2.2.2 P10 to P17 (port 1) 24
2.2.3 P30 to P33 (port 3) 25
2.2.4 P60 and P61 (port 6) 25
2.2.5 P120 to P122 (port 12) 26
2.2.6 $\overline{\text{RESET}}$ 26
2.2.7 REGC 27
2.2.8 USBM 27
2.2.9 USBP 27
2.2.10 USBPUC 27
2.2.11 USBREGC 27
2.2.12 VDD and EVDD 27
2.2.13 VSS and EVSS 27
2.2.14 FLMD0 27
2.3 Pin I/O Circuits and Recommended Connection of Unused Pins 28

CHAPTER 3 CPU ARCHITECTURE 31

3.1 Memory Space 31
3.1.1 Internal program memory space 34
3.1.2 Internal data memory space 36
3.1.3 Special function register (SFR) area 36
3.1.4 USB area 36
3.1.5 Data memory addressing 37
3.2 Processor Registers 38
3.2.1 Control registers 38
3.2.2 General-purpose registers 42
3.2.3 Special function registers (SFRs) 43
3.3 Instruction Address Addressing 48
3.3.1 Relative addressing 48
3.3.2 Immediate addressing 49
3.3.3 Table indirect addressing 50
3.3.4 Register addressing 51
3.4 Operand Address Addressing 52
3.4.1 Implied addressing 52

3.4.2 Register addressing 53
3.4.3 Direct addressing 54
3.4.4 Short direct addressing 55
3.4.5 Special function register (SFR) addressing 56
3.4.6 Register indirect addressing 57
3.4.7 Based addressing 58
3.4.8 Based indexed addressing 59
3.4.9 Stack addressing 60

CHAPTER 4 PORT FUNCTIONS 61

4.1 Port Functions 61
4.2 Port Configuration 62
4.2.1 Port 0 63
4.2.2 Port 1 65
4.2.3 Port 3 71
4.2.4 Port 6 73
4.2.5 Port 12 74
4.3 Registers Controlling Port Function 77
4.4 Port Function Operations 80
4.4.1 Writing to I/O port 80
4.4.2 Reading from I/O port 80
4.4.3 Operations on I/O port 80
4.5 Settings of Port Mode Register and Output Latch When Using Alternate Function 81

CHAPTER 5 CLOCK GENERATOR 82

5.1 Functions of Clock Generator 82
5.2 Configuration of Clock Generator 83
5.3 Registers Controlling Clock Generator 85
5.4 System Clock Oscillator 95
5.4.1 X1 oscillator 95
5.4.2 Internal high-speed oscillator 97
5.4.3 Internal low-speed oscillator 97
5.4.4 Prescaler 97
5.5 Clock Generator Operation 98
5.6 Controlling Clock 100
5.6.1 Controlling high-speed system clock 100
5.6.2 Example of controlling internal high-speed oscillation clock 103
5.6.3 Example of controlling internal low-speed oscillation clock 105
5.6.4 Example of controlling USB clock 106
5.6.5 Clocks supplied to CPU and peripheral hardware 107
5.6.6 CPU clock status transition diagram 108
5.6.7 Condition before changing CPU clock and processing after changing CPU clock 111
5.6.8 Time required for switchover of CPU clock and main system clock 112
5.6.9 Conditions before clock oscillation is stopped 113
5.6.10 Peripheral hardware and source clocks 113

CHAPTER 6 16-BIT TIMER/EVENT COUNTER 00 114

6.1 Functions of 16-Bit Timer/Event Counter 00 114
6.2 Configuration of 16-Bit Timer/Event Counter 00 115

6.3 Registers Controlling 16-Bit Timer/Event Counter 00 119
6.4 Operation of 16-Bit Timer/Event Counter 00 126
6.4.1 Interval timer operation 126
6.4.2 Square wave output operation 129
6.4.3 External event counter operation 132
6.4.4 Operation in clear & start mode entered by TI000 pin valid edge input 135
6.4.5 Free-running timer operation 148
6.4.6 PPG output operation 157
6.4.7 One-shot pulse output operation 160
6.4.8 Pulse width measurement operation 165
6.5 Special Use of TM00 173
6.5.1 Rewriting CR010 during TM00 operation 173
6.5.2 Setting LVS00 and LVR00 173
6.6 Cautions for 16-Bit Timer/Event Counter 00 175

CHAPTER 7 8-BIT TIMER/EVENT COUNTERS 50 AND 51 179

7.1 Functions of 8-Bit Timer/Event Counters 50 and 51 179
7.2 Configuration of 8-Bit Timer/Event Counters 50 and 51 179
7.3 Registers Controlling 8-Bit Timer/Event Counters 50 and 51 182
7.4 Operations of 8-Bit Timer/Event Counters 50 and 51 187
7.4.1 Operation as interval timer 187
7.4.2 Operation as external event counter 189
7.4.3 Square-wave output operation 190
7.4.4 PWM output operation 191
7.5 Cautions for 8-Bit Timer/Event Counters 50 and 51 195

CHAPTER 8 8-BIT TIMER H1 196

8.1 Functions of 8-Bit Timer H1 196
8.2 Configuration of 8-Bit Timer H1 196
8.3 Registers Controlling 8-Bit Timer H1 199
8.4 Operation of 8-Bit Timer H1 202
8.4.1 Operation as interval timer/square-wave output 202
8.4.2 Operation as PWM output 205
8.4.3 Carrier generator operation 211

CHAPTER 9 WATCHDOG TIMER 218

9.1 Functions of Watchdog Timer 218
9.2 Configuration of Watchdog Timer 219
9.3 Register Controlling Watchdog Timer 220
9.4 Operation of Watchdog Timer 221
9.4.1 Controlling operation of watchdog timer 221
9.4.2 Setting overflow time of watchdog timer 222
9.4.3 Setting window open period of watchdog timer 223

CHAPTER 10 SERIAL INTERFACE UART6 224

10.1 Functions of Serial Interface UART6 224
10.2 Configuration of Serial Interface UART6 225
10.3 Registers Controlling Serial Interface UART6 228

10.4 Operation of Serial Interface UART6 235
10.4.1 Operation stop mode 235
10.4.2 Asynchronous serial interface (UART) mode 236
10.4.3 Dedicated baud rate generator 248
10.5 Cautions for Serial Interface UART6 254

CHAPTER 11 SERIAL INTERFACE CSI10 255

11.1 Functions of Serial Interface CSI10 255
11.2 Configuration of Serial Interface CSI10 256
11.3 Registers Controlling Serial Interface CSI10 258
11.4 Operation of Serial Interface CSI10 261
11.4.1 Operation stop mode 261
11.4.2 3-wire serial I/O mode 262
11.5 Caution for Serial Interface CSI10 271

CHAPTER 12 USB FUNCTION CONTROLLER USBF 272

12.1 Overview 272
12.2 Configuration 273
12.3 Requests 275
12.3.1 Automatic requests 275
12.3.2 Other requests 282
12.4 Register Configuration 283
12.4.1 Control registers 283
12.4.2 Data hold registers 323
12.4.3 Request data registers 337
12.4.4 Peripheral control register 349
12.5 STALL Handshake or No Handshake 352
12.6 Register Values in Specific Status 353
12.7 FW Processing 355
12.7.1 Initialization processing 357
12.7.2 Interrupt servicing 360
12.7.3 USB main processing 361
12.7.4 Suspend/Resume processing 386
12.7.5 Processing after power application 389
12.8 External Circuit Configuration 392
12.8.1 Outline 392
12.8.2 USB connection example 393
12.9 Cautions for USB Function Controller USBF 394

CHAPTER 13 INTERRUPT FUNCTIONS 395

13.1 Interrupt Function Types 395
13.2 Interrupt Sources and Configuration 395
13.3 Registers Controlling Interrupt Functions 398
13.4 Interrupt Servicing Operations 405
13.4.1 Maskable interrupt acknowledgement 405
13.4.2 Software interrupt request acknowledgement 407
13.4.3 Multiple interrupt servicing 408
13.4.4 Interrupt request hold 411

CHAPTER 14 STANDBY FUNCTION 412

14.1 Standby Function and Configuration 412

14.1.1 Standby function 412

14.1.2 Registers controlling standby function 412

14.2 Standby Function Operation 415

14.2.1 HALT mode 415

14.2.2 STOP mode 419

CHAPTER 15 RESET FUNCTION 424

15.1 Register for Confirming Reset Source 432

CHAPTER 16 POWER-ON-CLEAR CIRCUIT 433

16.1 Functions of Power-on-Clear Circuit 433

16.2 Configuration of Power-on-Clear Circuit 434

16.3 Operation of Power-on-Clear Circuit 434

16.4 Cautions for Power-on-Clear Circuit 436

CHAPTER 17 LOW-VOLTAGE DETECTOR 438

17.1 Functions of Low-Voltage Detector 438

17.2 Configuration of Low-Voltage Detector 438

17.3 Registers Controlling Low-Voltage Detector 439

17.4 Operation of Low-Voltage Detector 442

17.4.1 When used as reset 443

17.4.2 When used as interrupt 445

17.5 Cautions for Low-Voltage Detector 447

CHAPTER 18 OPTION BYTE 450

18.1 Functions of Option Bytes 450

18.2 Format of Option Byte 451

CHAPTER 19 FLASH MEMORY 454

19.1 Internal Memory Size Switching Register 454

19.2 Internal Expansion RAM Size Switching Register 455

19.3 Writing with Flash Memory Programmer 456

19.4 Programming Environment 459

19.5 Communication Mode 459

19.6 Connection of Pins on Board 461

19.6.1 FLMD0 pin 461

19.6.2 Serial interface pins 461

19.6.3 $\overline{\text{RESET}}$ pin 463

19.6.4 Port pins 463

19.6.5 REGC pin 463

19.6.6 Other signal pins 463

19.6.7 Power supply 464

19.7 Programming Method 464

19.7.1 Controlling flash memory 464

19.7.2 Flash memory programming mode 465

19.7.3 Selecting communication mode 466
19.7.4 Communication commands 467
19.8 Security Settings 468
19.9 Flash Memory Programming by Self-Programming 470
19.9.1 Boot swap function 472

CHAPTER 20 ON-CHIP DEBUG FUNCTION 474

20.1 Connecting QB-MINI2 to μPD78F0730 474
20.2 Reserved Area Used by QB-MINI2 476

CHAPTER 21 INSTRUCTION SET 477

21.1 Conventions Used in Operation List 477
21.1.1 Operand identifiers and specification methods 477
21.1.2 Description of operation column 478
21.1.3 Description of flag operation column 478
21.2 Operation List 479
21.3 Instructions Listed by Addressing Type 487

CHAPTER 22 ELECTRICAL SPECIFICATIONS 490

CHAPTER 23 PACKAGE DRAWINGS 506

CHAPTER 24 RECOMMENDED SOLDERING CONDITIONS 507

CHAPTER 25 CAUTIONS FOR WAIT 508

25.1 Cautions for Wait 508
25.2 Peripheral Hardware That Generates Wait 508

APPENDIX A DEVELOPMENT TOOLS 509

A.1 Software Package 512
A.2 Language Processing Software 512
A.3 Control Software 513
A.4 Flash Memory Writing Tools 513
A.4.1 When using flash memory programmer FG-FP5 and FL-PR5 513
A.4.2 When using on-chip debug emulator with programming function QB-MINI2 513
A.5 Debugging Tools (Hardware) 514
A.5.1 When using in-circuit emulator QB-780731 514
A.5.2 When using on-chip debug emulator with programming function QB-MINI2 515
A.6 Debugging Tools (Software) 515

APPENDIX B NOTES ON TARGET SYSTEM DESIGN 516

APPENDIX C REGISTER INDEX 517

C.1 Register Index (In Alphabetical Order with Respect to Register Names) 517
C.2 Register Index (In Alphabetical Order with Respect to Register Symbol) 521

APPENDIX D REVISION HISTORY 525

D.1 Major Revisions in This Edition 525

D.2 Revision History of Preceding Editions 526

CHAPTER 1 OUTLINE

1.1 Features

- High speed instruction execution (0.125 μs: @ 16 MHz operation with high-speed system clock)
- General-purpose register: 8 bits × 32 registers (8 bits × 8 registers × 4 banks)
- ROM, RAM capacities

Item / Part Number	Program Memory (ROM)	Data Memory	
	Flash memory[Note]	Internal High-Speed RAM[Note]	Internal Expansion RAM[Note]
μPD78F0730	16 KB	1 KB	2 KB

Note The internal flash memory, internal high-speed RAM capacities, and internal expansion RAM capacities can be changed using the internal memory size switching register (IMS) and the internal expansion RAM size switching register (IXS). For IMS and IXS, see **19.1 Internal Memory Size Switching Register** and **19.2 Internal Expansion RAM Size Switching Register**.

- On-chip USB function controller (USBF)
- On-chip single-power-supply flash memory
- Self-programming (with boot swap function)
- On-chip debug function[Note]
- On-chip power-on-clear (POC) circuit and low-voltage detector (LVI)
- On-chip watchdog timer (operable with the internal low-speed oscillation clock)
- I/O ports: 19 (N-ch open drain: 2)
- Timer: 5 channels
  - 16-bit timer/event counter: 1 channel
  - 8-bit timer/event counter: 2 channels
  - 8-bit timer: 1 channel
  - Watchdog timer: 1 channel
- Serial interface: 3 channels
  - UART: 1 channel
  - CSI: 1 channel
  - USB: 1 channel
- Power supply voltage: V_{DD} = 4.0 to 5.5 V
- Operating ambient temperature: T_A = -40 to +85°C

Note The μPD78F0730 has an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. NEC Electronics is not liable for problems occurring when the on-chip debug function is used.

1.2 Applications

- USB – serial conversion

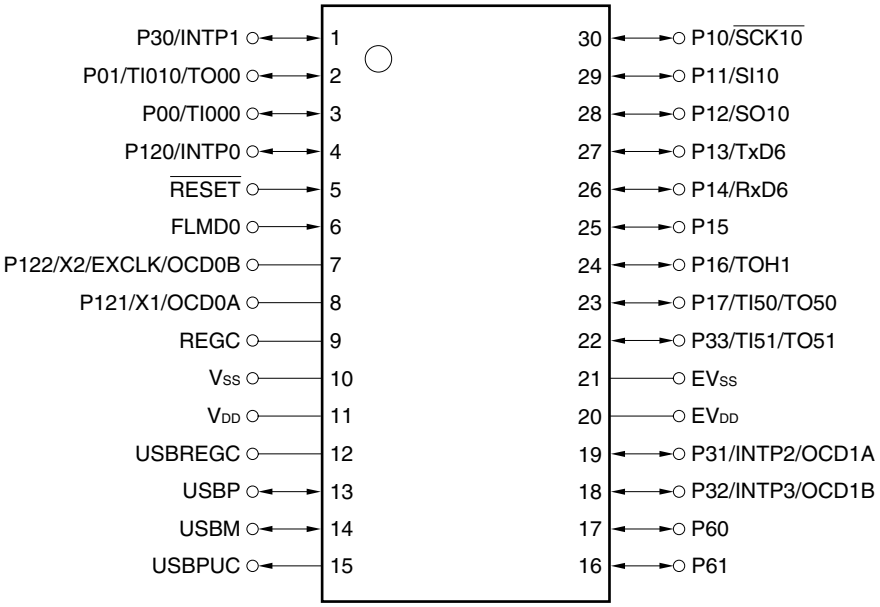
1.3 Ordering Information

- **Flash memory version**

Part Number	Package
µPD78F0730MC-CAB-AX	30-pin plastic SSOP (7.62 mm)

1.4 Pin Configuration (Top View)

- 30-pin plastic SSOP (7.62 mm)

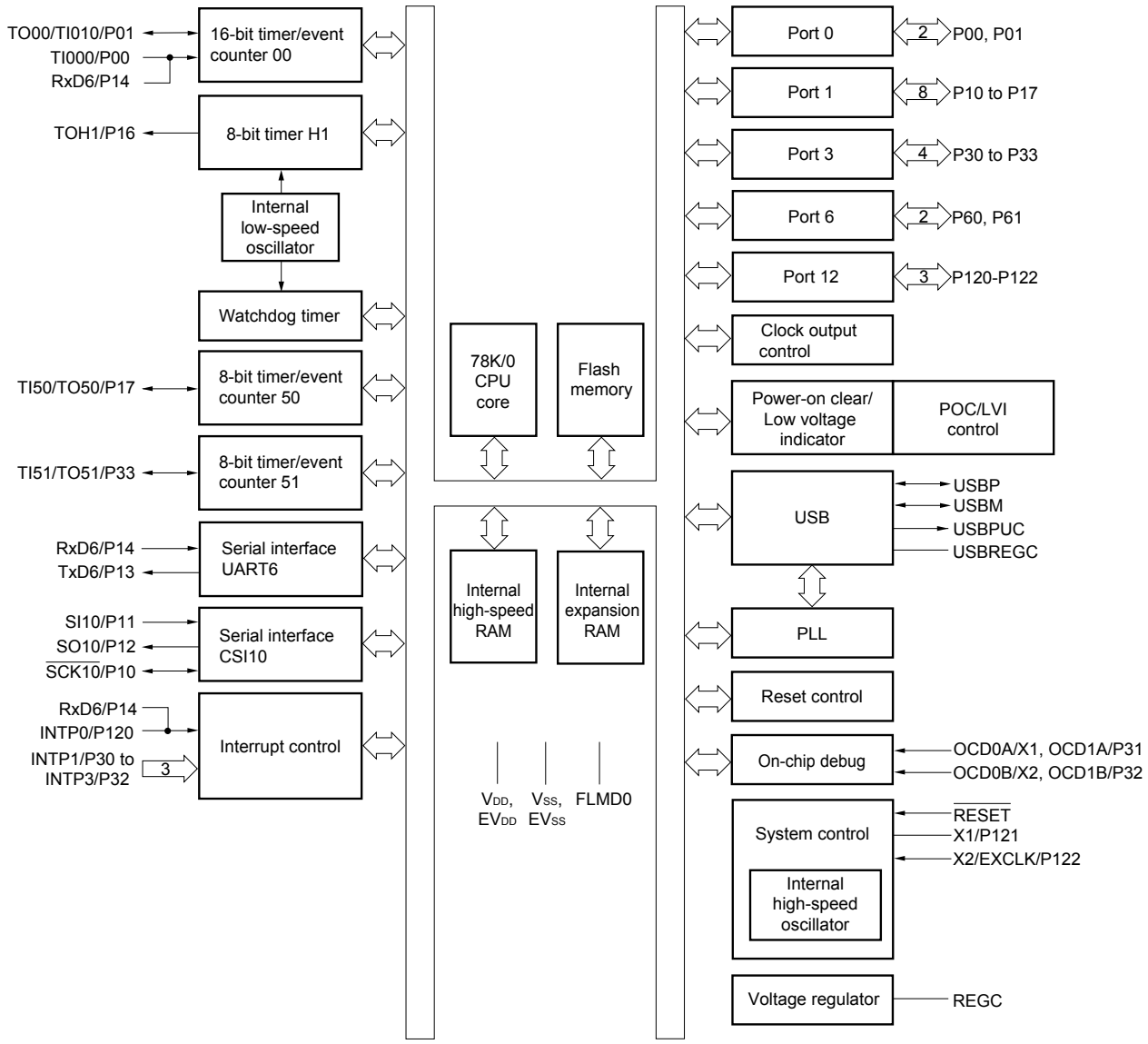


Caution Connect the REGC and USBREGC pins to V_{SS} via a capacitor (0.47 to 1 µF: recommended).

Pin Identification

EV_{DD}:	Power supply for port	$\overline{SCK10}$:	Serial clock input/output
EV_{SS}:	Ground for port	SI10:	Serial data input
EXCLK:	External clock input (main system clock)	SO10:	Serial data output
		TI000, TI010:	Timer input
FLMD0:	Flash programming mode	TI50, TI51:	Timer input
INTP0 to INTP3:	External interrupt input	TO00:	Timer output
OCD0A, OCD0B:	On chip debug input/output	TO50, TO51:	Timer output
OCD1A, OCD1B:	On chip debug input/output	TOH1:	Timer output
P00, P01:	Port 0	TxD6:	Transmit data
P10 to P17:	Port 1	USBM:	USB port (–)
P30 to P33:	Port 3	USBP:	USB port (+)
P60, P61:	Port 6	USBPUC:	USB pull-up resistor control
P120 to P122:	Port 12	USBREGC:	USB regulator capacitance
REGC	Regulator capacitance	V_{DD}:	Power supply
$\overline{RESET}$:	Reset	V_{SS}:	Ground
RxD6:	Receive data	X1, X2:	Crystal oscillator (main system clock)

1.5 Block Diagram



1.6 Outline of Functions

(1/2)

Item		μPD78F0730
Internal memory	Flash memory (self-programming supported)[Note]	16 KB
	High-speed RAM[Note]	1 KB
	Expansion RAM[Note]	2 KB
Memory space		64 KB
Main system clock (oscillation frequency)	High-speed system clock	X1 (crystal/ceramic) oscillation, external main system clock input (EXCLK) 12 or 16 MHz: V_{DD} = 4.0 to 5.5 V
	Internal high-speed oscillation clock	Internal oscillation 16 MHz (TYP.): V_{DD} = 4.0 to 5.5 V
Internal low-speed oscillation clock (for TMH1, WDT)		Internal oscillation 240 kHz (TYP.): V_{DD} = 4.0 to 5.5 V
USB clock		X1 (crystal/ceramic) oscillation, external main system clock input (EXCLK) 12/2 or 16/4 MHz: V_{DD} = 4.0 to 5.5 V (multiplied by 8 or 12 by PLL function)
General-purpose registers		8 bits × 32 registers (8 bits × 8 registers × 4 banks)
Minimum instruction execution time		0.125 μs (high-speed system clock: @ f_{XH} = 16 MHz operation)
		0.125 μs (internal high-speed oscillation clock: @ f_{RH} = 16 MHz (TYP.) operation)
Instruction set		• 8-bit operation, 16-bit operation • Multiply/divide (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulate (set, reset, test, and Boolean operation) • BCD adjust, etc.
I/O ports		Total: 19 CMOS I/O: 17 N-ch open-drain I/O (6 V withstanding voltage): 2
Timers		• 16-bit timer/event counter: 1 channel • 8-bit timer/event counter: 2 channels • 8-bit timer: 1 channel • Watchdog timer: 1 channel
	Timer outputs	4 (PWM output: 3, PPG output: 1)
Serial interface		• UART: 1 channel • 3-wire serial I/O: 1 channel • USB: 1 channel
Vectored interrupt sources	Internal	14
	External	4
Reset		• Reset using $\overline{\text{RESET}}$ pin • Internal reset by watchdog timer • Internal reset by power-on-clear • Internal reset by low-voltage detector

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

(2/2)

Item	µPD78F0730
On-chip debug function	Provided
Power supply voltage	V_{DD} = 4.0 to 5.5 V
Operating ambient temperature	T_A = -40 to +85°C
Package	30-pin plastic SSOP (7.62 mm)

An outline of the timer is shown below.

		16-Bit Timer/ Event Counter 00	8-Bit Timer/ Event Counters 50 and 51		8-Bit Timer H1	Watchdog Timer
		TM00	TM50	TM51	TMH1	
Function	Interval timer	1 channel	1 channel	1 channel	1 channel	–
	External event counter	1 channel	1 channel	1 channel	–	–
	PPG output	1 output	–	–	–	–
	PWM output	–	1 output	1 output	1 output	–
	Pulse width measurement	2 inputs	–	–	–	–
	Square-wave output	1 output	1 output	1 output	1 output	–
	Carrier generator	–	–	–	1 output[Nore 2]	–
	Watchdog timer	–	–	–	–	1 channel
Interrupt source		2	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

There are two types of pin I/O buffer power supplies: EV_{DD} and V_{DD}. The relationship between these power supplies and the pins is shown below.

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

Power Supply	Corresponding Pins
EV_{DD}	Port pins other than P121 and P122
V_{DD}	• P121 and P122 • Pins other than port

(1) Port functions

Function Name	I/O	Function	After Reset	Alternate Function
P00	I/O	Port 0. 2-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting.	Input port	TI000
P01				TI010/TO00
P10	I/O	Port 1. 8-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting.	Input port	$\overline{SCK10}$
P11				SI10
P12				SO10
P13				TxD6
P14				RxD6
P15				–
P16				TOH1
P17				TI50/TO50
P30	I/O	Port 3. 4-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting.	Input port	INTP1
P31				INTP2/OCD1A
P32				INTP3/OCD1B
P33				TI51/TO51
P60	I/O	Port 6. 2-bit I/O port. Output of P60 and P61 is N-ch open-drain output (6 V tolerance). Input/output can be specified in 1-bit units.	Input port	–
P61				–
P120	I/O	Port 12. 3-bit I/O port. Input/output can be specified in 1-bit units. Only for P120, use of an on-chip pull-up resistor can be specified by a software setting.	Input port	INTP0
P121				X1/OCD0A
P122				X2/EXCLK/OCD0B

(2) Non-port functions

Function Name	I/O	Function	After Reset	Alternate Function
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
INTP1				P30
INTP2				P31/OCD1A
INTP3				P32/OCD1B
REGC	–	Connecting regulator output (2.5 V) stabilization capacitance for internal operation. Connect to V_{SS} via a capacitor (0.47 to 1 µF: recommended).	–	–
$\overline{\text{RESET}}$	Input	System reset input	–	–
RxD6	Input	Serial data input to UART6	Input port	P14
$\overline{\text{SCK10}}$	I/O	Clock input/output for CSI10	Input port	P10
SI10	Input	Serial data input to CSI10	Input port	P11
SO10	Output	Serial data output from CSI10	Input port	P12
TI000	Input	External count clock input to 16-bit timer/event counter 00 Capture trigger input to capture registers (CR000, CR010) of 16-bit timer/event counter 00	Input port	P00
TI010	Input	Capture trigger input to capture register (CR000) of 16-bit timer/event counter 00	Input port	P01/TO00
TI50	Input	External count clock input to 8-bit timer/event counter 50	Input port	P17/TO50
TI51		External count clock input to 8-bit timer/event counter 51		P33/TO51
TO00	Output	16-bit timer/event counter 00 output	Input port	P01/TI010
TO50	Output	8-bit timer/event counter 50 output	Input port	P17/TI50
TO51		8-bit timer/event counter 51 output		P33/TI51
TOH1		8-bit timer H1 output		P16
TxD6	Output	Serial data output from UART6	Input port	P13
USBM	I/O	USB data input/output (–)	Input port	–
USBP	I/O	USB data input/output (+)	Input port	–
USBPUC	Output	USB pull-up resistor control pin	Low level output	–
USBREGC	–	Regulator output (3.3 V) stabilization capacitance for USB. Connect to V_{SS} via a capacitor (0.47 to 1 µF: recommended).	–	–
X1	–	Connecting resonator for main system clock	Input port	P121/OCD0A
X2	–		Input port	P122/EXCLK/OCD0B
EXCLK	Input	External clock input for main system clock	Input port	P122/X2/OCD0B
V_{DD}	–	Positive power supply (P121 and P122 and except for ports)	–	–
EV_{DD}	–	Positive power supply for ports (other than P121 and P122)	–	–
V_{SS}	–	Ground potential (P121 and P122 and except for ports)	–	–
EV_{SS}	–	Ground potential for ports (other than P121 and P122)	–	–
OCD0A	Input	Connection for on-chip debug mode setting pins	Input port	P121/X1
OCD1A				P31/INTP2
OCD0B	–			P122/X2/EXCLK
OCD1B				P32/INTP3

2.2 Description of Pin Functions

2.2.1 P00 and P01 (port 0)

P00 and P01 function as a 2-bit I/O port. These pins also function as timer I/O.

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

(1) Port mode

P00 and P01 function as a 2-bit I/O port. P00 and P01 can be set to input or output port in 1-bit units using port mode register 0 (PM0). Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 0 (PU0).

(2) Control mode

P00 and P01 function as timer I/O.

(a) TI000

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

(b) TI010

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

(c) TO00

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

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 serial interface data I/O, clock I/O, and timer I/O.

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).

(2) Control mode

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

(a) SI10

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

(b) SO10

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

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

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

(d) 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

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

(g) TO50

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

(h) TOH1

This is the timer output pin of 8-bit timer H1.

2.2.3 P30 to P33 (port 3)

P30 to P33 function as a 4-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 P33 function as a 4-bit I/O port. P30 to P33 can be set to input or output port in 1-bit units using port mode register 3 (PM3). Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 3 (PU3).

(2) Control mode

P30 to P33 function as external interrupt request input and timer I/O.

(a) INTP1 to INTP3

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

(b) TI51

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

(c) TO51

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

Cautions 1. In the µPD78F0730, be sure to pull the P31/INTP2/OCD1A pin down before a reset release to prevent malfunction.

2. When writing the flash memory with a flash memory programmer, connect P31/INTP2/OCD1A as follows.

• P31/INTP2/OCD1A: Connect to EV_{SS} via a resistor (10 kΩ: recommended).

The above connection is not necessary when writing the flash memory by means of self programming.

Remark Only for the µPD78F0730, P31 and P32 can be used as on-chip debug mode setting pins (OCD1A, OCD1B) when the on-chip debug function is used. For how to connect an on-chip debug emulator (QB-MINI2), see **CHAPTER 20 ON-CHIP DEBUG FUNCTION.**

2.2.4 P60 and P61 (port 6)

P60 and P61 function as a 2-bit I/O port.

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

(1) Port mode

P60 and P61 function as a 2-bit I/O port. P60 and P61 can be set to input port or output port in 1-bit units using port mode register 6 (PM6).

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

2.2.5 P120 to P122 (port 12)

P120 to P122 function as a 3-bit I/O port. These pins also function as pins for external interrupt request input, connecting resonator for main system clock, and external clock input for main system clock. The following operation modes can be specified in 1-bit units.

(1) Port mode

P120 to P122 function as a 3-bit I/O port. P120 to P122 can be set to input or output port using port mode register 12 (PM12). Only for P120, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 12 (PU12).

(2) Control mode

P120 to P122 function as pins for external interrupt request input, connecting resonator for main system clock, and external clock input for main system clock.

(a) INTP0

This functions as an external interrupt request input (INTP0) for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.

(b) X1, X2

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

(c) EXCLK

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

Caution When writing the flash memory with a flash memory programmer, connect P121/X1/OCD0A as follows.
• P121/X1/OCD0A: When using this pin as a port, connect it to V_{SS} via a resistor (10 kΩ: recommended) (in the input mode) or leave it open (in the output mode).
The above connection is not necessary when writing the flash memory by means of self programming.

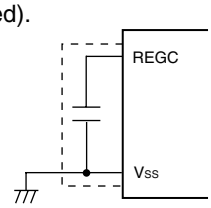
Remark Only for the μPD78F0730, X1 and X2 can be used as on-chip debug mode setting pins (OCD0A, OCD0B) when the on-chip debug function is used. For how to connect an on-chip debug emulator (QB-MINI2), see **CHAPTER 20 ON-CHIP DEBUG FUNCTION.**

2.2.6 $\overline{\text{RESET}}$

This is the active-low system reset input pin.

2.2.7 REGC

This is the pin for connecting regulator output (2.5 V) stabilization capacitance for internal operation. Connect this pin to V_{SS} via a capacitor (0.47 to 1.0 μF: recommended).



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

2.2.8 USBM

This is the pin for inputting/outputting data (–) to USB ports.

2.2.9 USBP

This is the pin for inputting/outputting data (+) to USB ports.

2.2.10 USBPUC

This is the pin for controlling pull-up resistors connected to USB ports.

2.2.11 USBREGC

This is the pin for connecting regulator output (3.3 V) stabilization capacitance for USB ports. Connect this pin to V_{SS} via a capacitor (0.47 to 1.0 μF: recommended).

2.2.12 V_{DD} and EV_{DD}

V_{DD} is the positive power supply pin for P121, P122 and other than ports.

EV_{DD} is the positive power supply pin for ports other than P121 and P122.

2.2.13 V_{SS} and EV_{SS}

V_{SS} is the ground potential pin for P121, P122 and other than ports.

EV_{SS} is the ground potential pin for ports other than P121 and P122.

2.2.14 FLMD0

This is a pin for setting flash memory programming mode.

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

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

To rewrite the data of the flash memory on-board, or to execute on-chip debug, connect this pin to V_{SS} via a resistor (10 kΩ: recommended).

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 2-2. Pin I/O Circuit Types

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P00/TI000	5-AH	I/O	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P01/TI010/TO00			
P10/$\overline{SCK10}$			
P11/SI10			
P12/SO10	5-AG		
P13/TxD6			
P14/RxD6	5-AH		
P15	5-AG		
P16/TOH1			
P17/TI50/TO50	5-AH		
P30/INTP1			
P31/INTP2/OCD1A Note 1			
P32/INTP3/OCD1B			
P33/TI51/TO51			
P60	13-AD		Input: Connect to EV_{SS}. Output: Leave this pin open at low-level output after clearing the output latch of the port to 0.
P61			
P121/X1/OCD0A Note 2, 3	37		Input: Independently connect to V_{DD} or V_{SS} via a resistor. Output: Leave open.
P122/X2/EXCLK/OCD0B Note 3			
USBM	24-A		Connect to EV_{SS}.
USBP	24-A		
USBPUC	3-C	Output	Leave open.
FLMD0	38	–	Connect to EV_{SS} or V_{SS}. Note 4
RESET	2	Input	Connect directly to V_{DD} or via a resistor.

Notes 1. When writing the flash memory with a flash memory programmer, connect P31/INTP2/OCD1A as follows.
• P31/INTP2/OCD1A: Connect to EV_{SS} via a resistor (10 kΩ: recommended).
The above connection is not necessary when writing the flash memory by means of self programming.

2. When writing the flash memory with a flash memory programmer, connect P121/X1/OCD0A as follows.
• P121/X1/OCD0A: When using this pin as a port, connect it to V_{SS} via a resistor (10 kΩ: recommended) (in the input mode) or leave it open (in the output mode).
The above connection is not necessary when writing the flash memory by means of self programming.

3. Use recommended connection above in I/O port mode (see **Figure 5-2 Format of Clock Operation Mode Select Register (OSCCTL)**) when these pins are not used.

4. FLMD0 is a pin that is used to write data to the flash memory. To rewrite the data of the flash memory on-board, or to execute on-chip debug, connect this pin to V_{SS} via a resistor (10 kΩ: recommended).

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

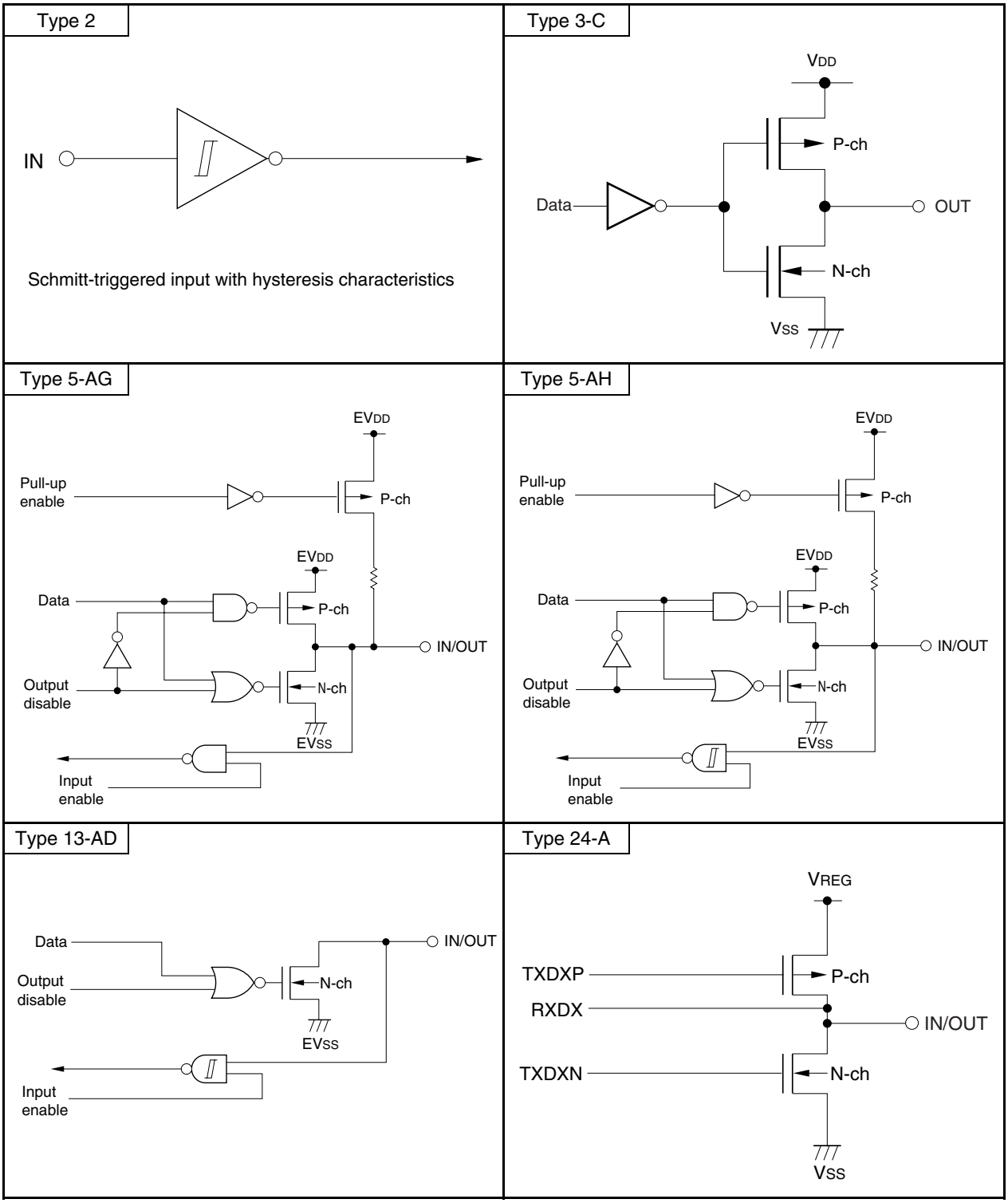
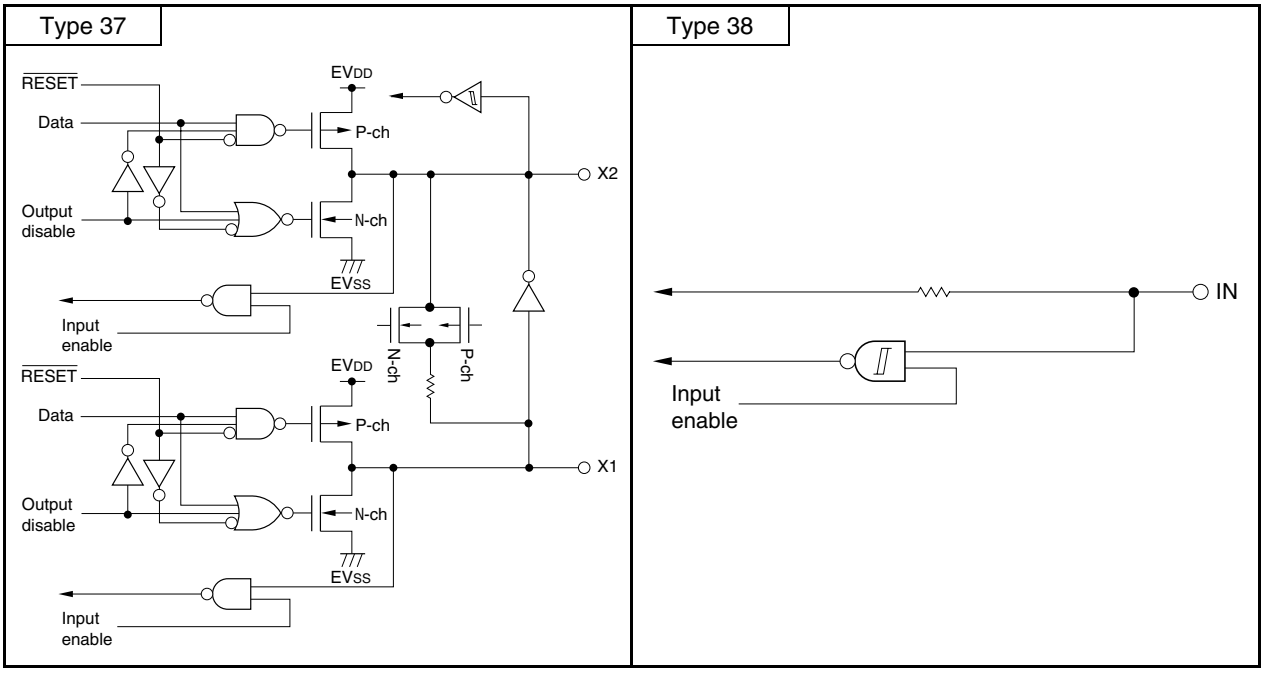


Figure 2-1. Pin I/O Circuit List (2/2)



CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Space

The μPD78F0730 can access a 64 KB memory space. Figure 3-1 shows the memory map.

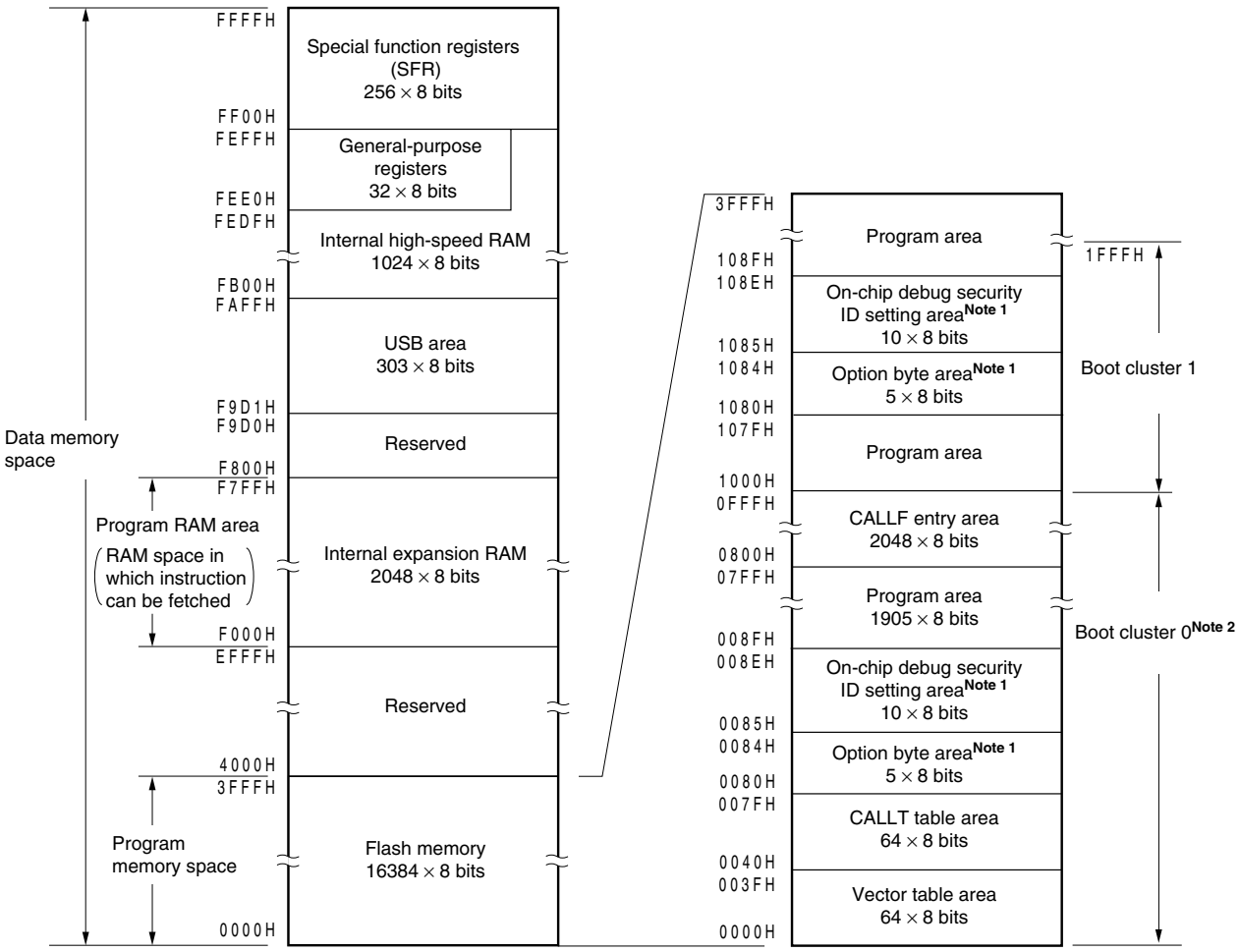
Cautions 1. Regardless of the internal memory capacity, the initial values of the internal memory size switching register (IMS) and internal expansion RAM size switching register (IXS) are fixed (IMS = CFH, IXS = 0CH). Therefore, set the value as indicated below.

2. To set the memory size, set IMS and then IXS. Set the memory size so that the internal ROM and internal expansion RAM areas do not overlap.

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

Flash Memory Version (μPD78F0730)	IMS	IXS	ROM Capacity	Internal High-Speed RAM Capacity	Internal Expansion RAM Capacity
μPD78F0730	C4H	08H	16 KB	1 KB	2 KB

Figure 3-1. Memory Map

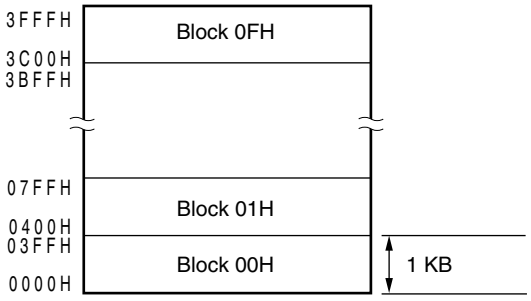


Notes 1. When boot swap is not used: Set the option bytes to 0080H to 0084H, and the on-chip debug security IDs to 0085H to 008EH.

When boot swap is used: Set the option bytes to 0080H to 0084H and 1080H to 1084H, and the on-chip debug security IDs to 0085H to 008EH and 1085H to 108EH.

2. Writing boot cluster 0 can be prohibited depending on the setting of security (see **19.8 Security Setting**).

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
0000H to 03FFH	00H
0400H to 07FFH	01H
0800H to 0BFFH	02H
0C00H to 0FFFH	03H
1000H to 13FFH	04H
1400H to 17FFH	05H
1800H to 1BFFH	06H
1C00H to 1FFFH	07H
2000H to 23FFH	08H
2400H to 27FFH	09H
2800H to 2BFFH	0AH
2C00H to 2FFFH	0BH
3000H to 33FFH	0CH
3400H to 37FFH	0DH
3800H to 3BFFH	0EH
3C00H to 3FFFH	0FH

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).

The µPD78F0730 incorporates internal ROM (flash memory), as shown below.

Table 3-3. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
µPD78F0730	Flash memory	16,384 × 8 bits (0000H to 3FFFH)

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	$\overline{\text{RESET}}$ input, POC, LVI, WDT	0018H	INTCSI10
0004H	INTLVI	001AH	INTTMH1
0006H	INTP0	001CH	INTUSB2
0008H	INTP1	001EH	INTTM50
000AH	INTP2	0020H	INTTM000
000CH	INTP3	0022H	INTTM010
000EH	INTUSB0	0024H	INTRSUM
0010H	INTUSB1	002AH	INTTM51
0012H	INTSRE6	003EH	BRK
0014H	INTSR6		
0016H	INTST6		

(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 and 1080H to 1084H can be used as an option byte area. Set the option byte at 0080H to 0084H when the boot swap is not used, and at 0080H to 0084H and 1080H to 1084H when the boot swap is used. For details, see **CHAPTER 18 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 and 1085H to 108EH can be used as an on-chip debug security ID setting area. Set the on-chip debug security ID of 10 bytes at 0085H to 008EH when the boot swap is not used and at 0085H to 008EH and 1085H to 108EH when the boot swap is used. For details, see **CHAPTER 20 ON-CHIP DEBUG FUNCTION**.

3.1.2 Internal data memory space

The μPD78F0730 incorporates the following RAMs.

(1) Internal high-speed RAM

Table 3-5. Internal High-Speed RAM Capacity

Part Number	Internal High-Speed RAM
μPD78F0730	1,024 × 8 bits (FB00H to FEFFH)

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

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

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

(2) Internal expansion RAM

Table 3-6. Internal Expansion RAM Capacity

Part Number	Internal Expansion RAM
μPD78F0730	2,048 × 8 bits (F000H to F7FFH)

The internal expansion RAM can also be used as a normal data area similar to the internal high-speed RAM, as well as a program area in which instructions can be written and executed.

The internal expansion RAM cannot 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 from FF00H to FFFFH (see **Table 3-7 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 USB area

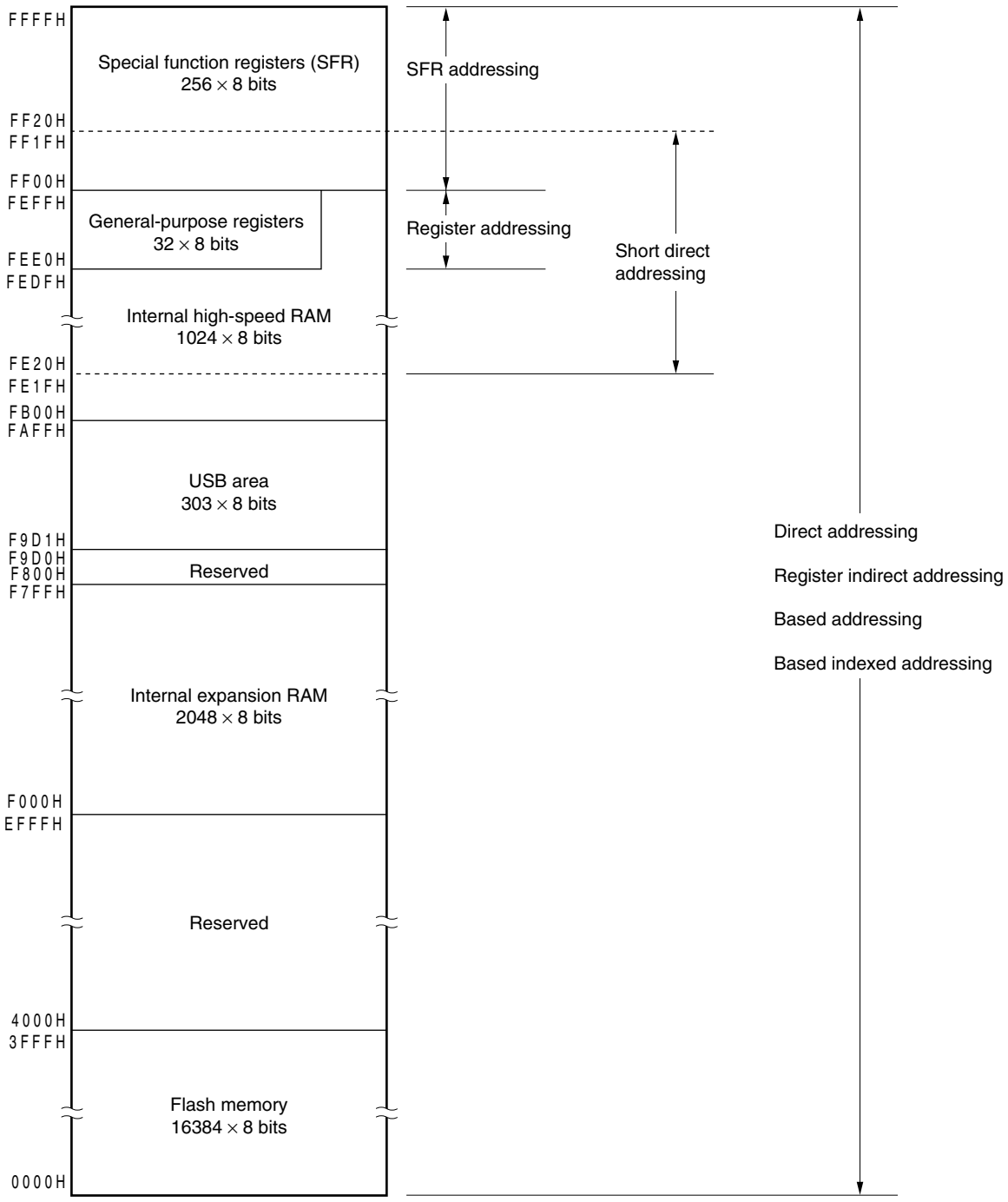
Some registers for USB (UF0DD0 to UF0DD17 and UF0CIE0 to UF0CIE255) are allocated in the area from F9D1H to FAFFH (see **12.4.3 Request data registers**).

3.1.5 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 µPD78F0730, 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. Figure 3-2 shows correspondence between data memory and addressing. For details of each addressing mode, see **3.4 Operand Address Addressing**.

Figure 3-2. Correspondence Between Data Memory and Addressing



3.2 Processor Registers

The μPD78F0730 incorporates 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 3-3. Format of Program Counter

	15															0
PC	PC15	PC14	PC13	PC12	PC11	PC10	PC9	PC8	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

(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 3-4. Format of Program Status Word

	7							0
PSW	IE	Z	RBS1	AC	RBS0	0	ISP	CY

(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 acknowledgement 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 acknowledgement and is set (1) upon EI instruction execution.

(b) Zero flag (Z)

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

(c) Register bank select flags (RBS0 and RBS1)

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

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

(d) Auxiliary carry flag (AC)

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

(e) In-service priority flag (ISP)

This flag manages the priority of acknowledgeable maskable vectored interrupts. When this flag is 0, low-level vectored interrupt requests specified by a priority specification flag register (PR0L, PR0H, PR1L, PR1H) (see **13.3 (3) Priority specification flag registers (PR0L, PR0H, PR1L, PR1H)**) can not be acknowledged. Actual request acknowledgement 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 3-5. Format of Stack Pointer

	15															0
SP	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0

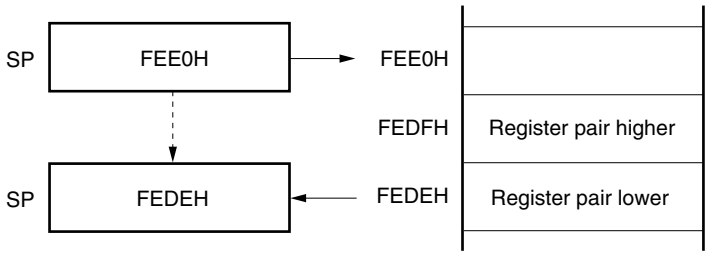
The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restored) from the stack memory.

Each stack operation saves/restores data as shown in Figures 3-6 and 3-7.

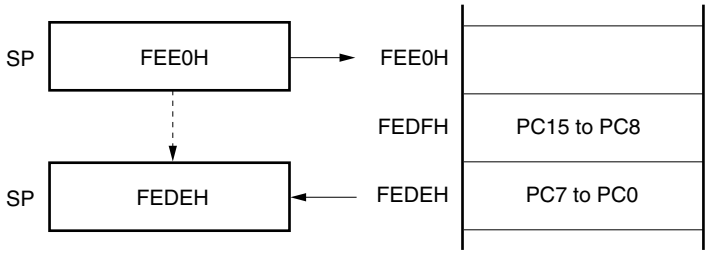
Caution Since reset signal generation makes the SP contents undefined, be sure to initialize the SP before using the stack.

Figure 3-6. 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)

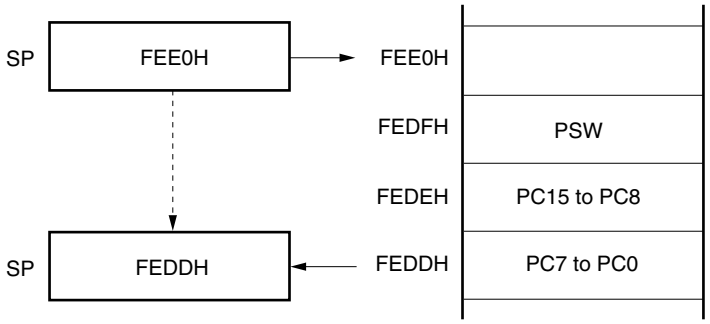
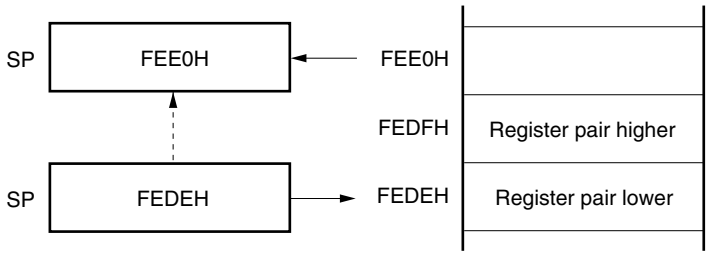
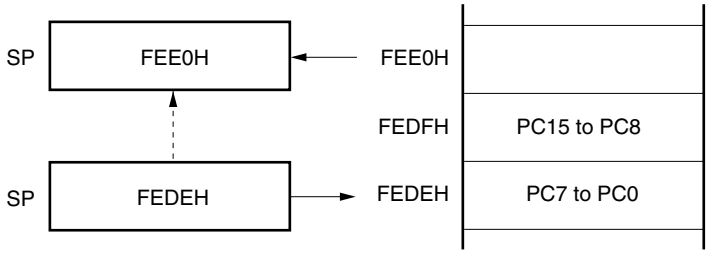


Figure 3-7. Data to Be Restored from Stack Memory

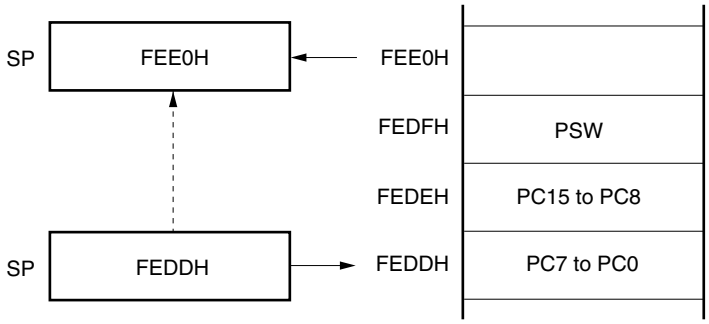
(a) POP rp instruction (when SP = FEDEH)



(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).

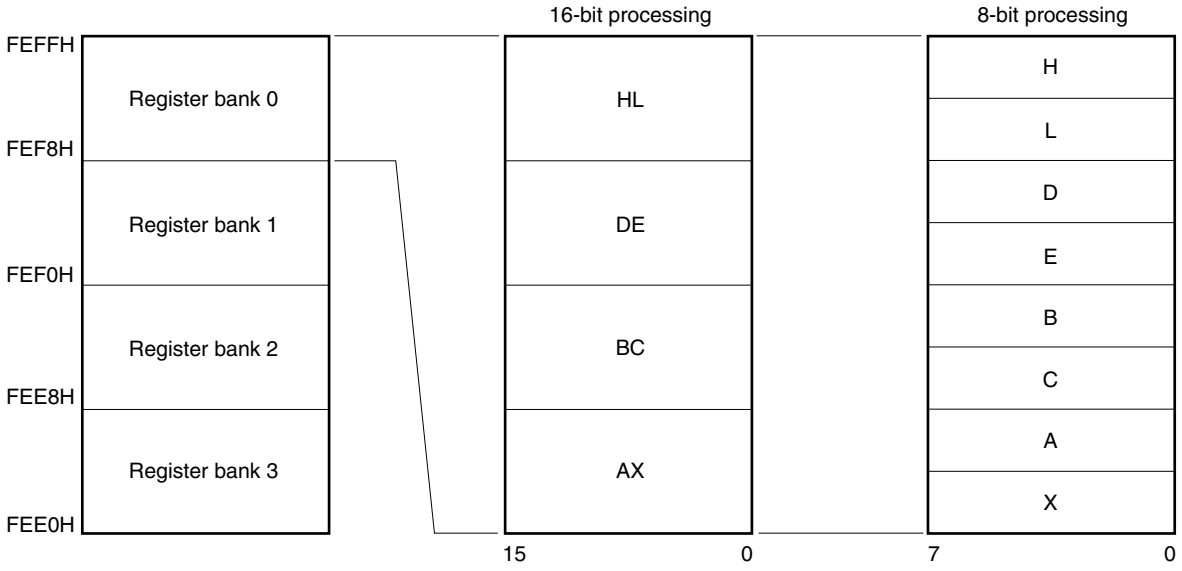
Each register can be used as an 8-bit register, and two 8-bit registers can also be used in a pair as a 16-bit register (AX, BC, DE, and HL).

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).

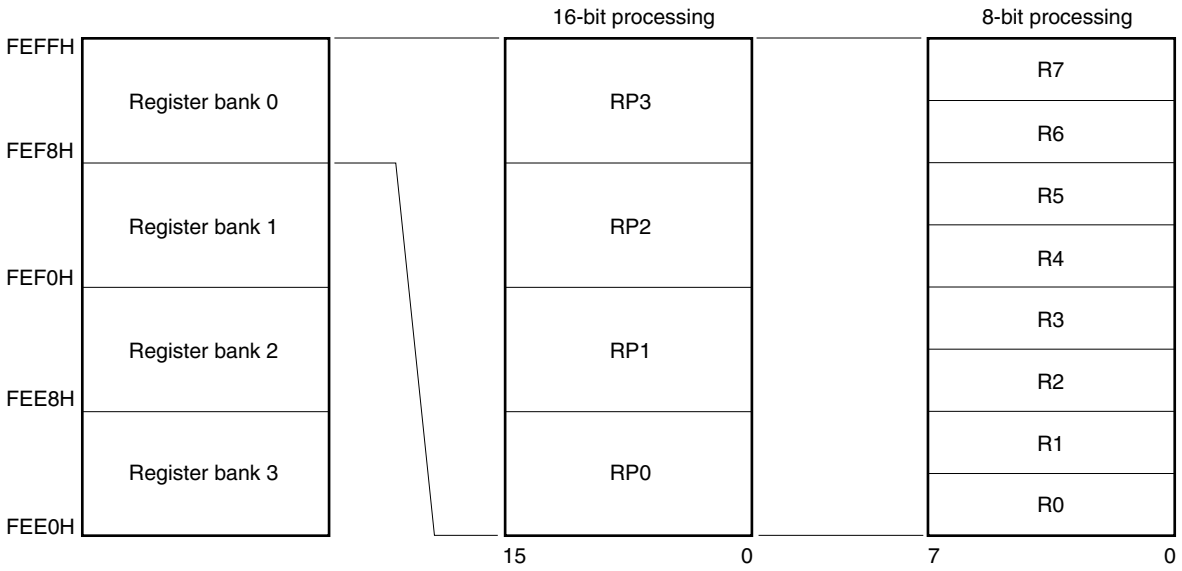
Register banks to be used for instruction execution are set by the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interrupts for each bank.

Figure 3-8. Configuration of General-Purpose Registers

(a) Function name



(b) Absolute name



3.2.3 Special function registers (SFRs)

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

SFRs are allocated to the FF00H to FFFFH area.

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

Each manipulation bit unit can be specified as follows.

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

Table 3-7 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, 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 3-7. Special Function Register List (1/4)

Address	Special Function Register (SFR) Name	Symbol	R/W	Manipulatable Bit Unit			After Reset
				1 Bit	8 Bits	16 Bits	
FF00H	Port register 0	P0	R/W	√	√	–	00H
FF01H	Port register 1	P1	R/W	√	√	–	00H
FF02H	UF0 EP0 read register	UF0E0R	R	–	√	–	Undefined
FF03H	Port register 3	P3	R/W	√	√	–	00H
FF06H	Port register 6	P6	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
FF0DH	UF0 bulk out 1 register	UF0BO1	R	–	√	–	Undefined
FF0EH	UF0 bulk in 1 register	UF0BI1	W	–	√	–	Undefined
FF0FH	Serial I/O shift register 10	SIO10	R	–	√	–	00H
FF10H	16-bit timer counter 00	TM00	R	–	–	√	0000H
FF11H							
FF12H	16-bit timer capture/compare register 000	CR000	R/W	–	–	√	0000H
FF13H							
FF14H	16-bit timer capture/compare register 010	CR010	R/W	–	–	√	0000H
FF15H							
FF16H	8-bit timer counter 50	TM50	R	–	√	–	00H
FF17H	8-bit timer compare register 50	CR50	R/W	–	√	–	00H
FF18H	UF0 EP0 setup register	UF0E0ST	R	–	√	–	00H
FF19H	UF0 EP0 write register	UF0E0W	W	–	√	–	Undefined
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
FF23H	Port mode register 3	PM3	R/W	√	√	–	FFH
FF26H	Port mode register 6	PM6	R/W	√	√	–	FFH
FF27H	UF0 INT status 0 register	UF0IS0	R	–	√	–	00H
FF28H	UF0 INT status 1 register	UF0IS1	R	–	√	–	00H
FF29H	UF0 INT status 2 register	UF0IS2	R	–	√	–	00H
FF2AH	UF0 INT status 3 register	UF0IS3	R	–	√	–	00H
FF2BH	UF0 INT status 4 register	UF0IS4	R	–	√	–	00H
FF2CH	Port mode register 12	PM12	R/W	√	√	–	FFH
FF2DH	UF0 GPR register	UF0GPR	R/W	–	√	–	00H
FF2EH	UF0 mode control register	UF0MODC	R/W	√	√	–	00H
FF2FH	UF0 mode status register	UF0MODS	R	√	√	–	00H
FF30H	Pull-up resistor option register 0	PU0	R/W	√	√	–	00H
FF31H	Pull-up resistor option register 1	PU1	R/W	√	√	–	00H

Table 3-7. Special Function Register List (2/4)

Address	Special Function Register (SFR) Name	Symbol	R/W	Manipulatable Bit Unit			After Reset
				1 Bit	8 Bits	16 Bits	
FF33H	Pull-up resistor option register 3	PU3	R/W	√	√	–	00H
FF37H	UF0 INT mask 0 register	UF0IM0	R/W	–	√	–	00H
FF38H	UF0 INT mask 1 register	UF0IM1	R/W	–	√	–	00H
FF39H	UF0 INT mask 2 register	UF0IM2	R/W	–	√	–	00H
FF3AH	UF0 INT mask 3 register	UF0IM3	R/W	–	√	–	00H
FF3BH	UF0 INT mask 4 register	UF0IM4	R/W	–	√	–	00H
FF3CH	Pull-up resistor option register 12	PU12	R/W	√	√	–	00H
FF41H	8-bit timer compare register 51	CR51	R/W	–	√	–	00H
FF43H	8-bit timer mode control register 51	TMC51	R/W	√	√	–	00H
FF48H	External interrupt rising edge enable register	EGP	R/W	√	√	–	00H
FF49H	External interrupt falling edge enable register	EGN	R/W	√	√	–	00H
FF4AH	UF0 INT clear 0 register	UF0IC0	W	–	√	–	FFH
FF4BH	UF0 INT clear 1 register	UF0IC1	W	–	√	–	FFH
FF4CH	UF0 INT clear 2 register	UF0IC2	W	–	√	–	FFH
FF4DH	UF0 INT clear 3 register	UF0IC3	W	–	√	–	FFH
FF4EH	UF0 INT clear 4 register	UF0IC4	W	–	√	–	FFH
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
FF60H	UF0 EP0NAK register	UF0E0N	R/W	–	√	–	00H
FF61H	UF0 EP0NAKALL register	UF0E0NA	R/W	–	√	–	00H
FF62H	UF0 EPNAK register	UF0EN	R/W	–	√	–	00H
FF63H	UF0 EPNAK mask register	UF0ENM	R/W	–	√	–	00H
FF64H	UF0 SNDSIE register	UF0SDS	R/W	–	√	–	00H
FF65H	UF0 CLR request register	UF0CLR	R	–	√	–	00H
FF66H	UF0 SET request register	UF0SET	R	–	√	–	00H
FF67H	UF0 EP status 0 register	UF0EPS0	R	–	√	–	00H
FF68H	UF0 EP status 1 register	UF0EPS1	R	√	√	–	00H
FF69H	UF0 EP status 2 register	UF0EPS2	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
FF70H	UF0 active interface number register	UF0AIFN	R/W	–	√	–	01H
FF71H	UF0 active alternative setting register	UF0AAS	R/W	–	√	–	1FH
FF72H	UF0 alternative setting status register	UF0ASS	R	–	√	–	FFH

Table 3-7. Special Function Register List (3/4)

Address	Special Function Register (SFR) Name	Symbol	R/W	Manipulatable Bit Unit			After Reset
				1 Bit	8 Bits	16 Bits	
FF73H	UF0 endpoint 1 interface mapping register	UF0E1IM	R/W	–	√	–	00H
FF74H	UF0 endpoint 2 interface mapping register	UF0E2IM	R/W	–	√	–	00H
FF75H	UF0 data end register	UF0DEND	R/W	–	√	–	00H
FF76H	UF0 EP0 length register	UF0E0L	R	–	√	–	00H
FF77H	UF0 bulk out 1 length register	UF0BO1L	R	–	√	–	00H
FF78H	UF0 descriptor length register	UF0DSCL	R/W	–	√	–	00H
FF79H	UF0 FIFO clear 0 register	UF0FIC0	W	–	√	–	00H
FF7AH	UF0 FIFO clear 1 register	UF0FIC1	W	–	√	–	00H
FF80H	Serial operation mode register 10	CSIM10	R/W	√	√	–	00H
FF81H	Serial clock selection register 10	CSIC10	R/W	√	√	–	00H
FF84H	Transmit buffer register 10	SOTB10	R/W	√	√	–	00H
FF8BH	USB function 0 buffer control register	UF0BC	R/W	–	√	–	00H
FF8CH	Timer clock selection register 51	TCL51	R/W	√	√	–	00H
FF90H	UF0 address register	UF0ADRS	R	–	√	–	00H
FF91H	UF0 configuration register	UF0CNF	R	–	√	–	00H
FF92H	UF0 interface 0 register	UF0IF0	R	–	√	–	00H
FF93H	UF0 interface 1 register	UF0IF1	R	–	√	–	00H
FF94H	UF0 interface 2 register	UF0IF2	R	–	√	–	00H
FF95H	UF0 interface 3 register	UF0IF3	R	–	√	–	00H
FF96H	UF0 interface 4 register	UF0IF4	R	–	√	–	00H
FF99H	Watchdog timer enable register	WDTE	R/W	–	√	–	Note 1 1AH/9AH
FF9AH	UF0 device status register	UF0DSTL	R/W	–	√	–	00H
FF9CH	UF0 EP0 status register	UF0E0SL	R/W	–	√	–	00H
FF9DH	UF0 EP1 status register	UF0E1SL	R/W	–	√	–	00H
FF9EH	UF0 EP2 status register	UF0E2SL	R/W	–	√	–	00H
FF9FH	Clock operation mode select register	OSCCTL	R/W	√	√	–	00H
FFA0H	Internal oscillation mode register	RCM	R/W	√	√	–	80H Note 2
FFA1H	Main clock mode register	MCM	R/W	√	√	–	00H
FFA2H	Main OSC control register	MOC	R/W	√	√	–	80H
FFA3H	Oscillation stabilization time counter status register	OSTC	R	√	√	–	00H
FFA4H	Oscillation stabilization time select register	OSTS	R/W	–	√	–	05H
FFA6H	PLL control register	PLLC	R/W	√	√	–	00H
FFA7H	USB clock control register	UCKC	R/W	√	√	–	00H
FFACH	Reset control flag register	RESF	R	–	√	–	00H Note 3
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

Notes 1. The reset value of WDTE is determined by setting of option byte.

2. The value of this register is 00H immediately after a reset release but automatically changes to 80H after oscillation accuracy stabilization of high-speed internal oscillator has been waited.

3. The reset value of RESF vary depending on the reset source.

Table 3-7. Special Function Register List (4/4)

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

Notes 1. The reset values of LVIM and LVIS vary depending on the reset source.

2. Regardless of the internal memory capacity, the initial values of the internal memory size switching register (IMS) and internal expansion RAM size switching register (IXS) are fixed (IMS = CFH, IXS = 0CH). Therefore, be sure to set IMS to C4H and IXS to 08H.

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

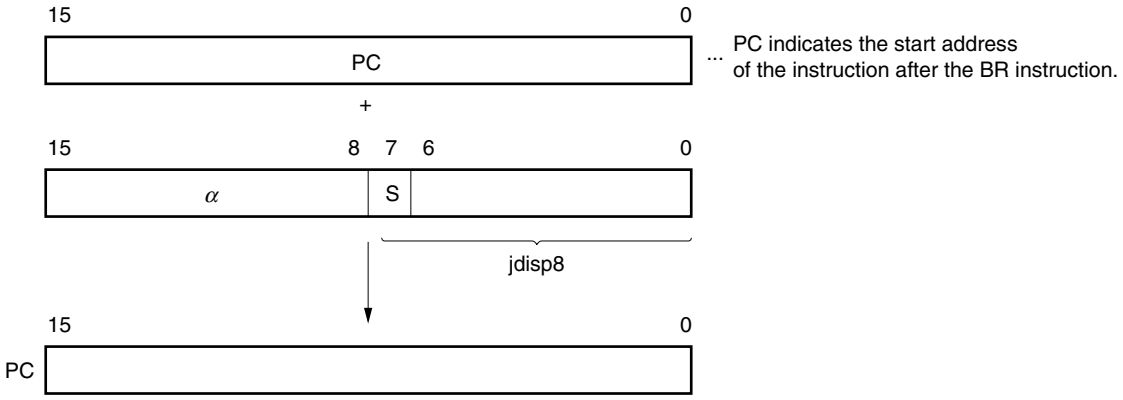
[Function]

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.

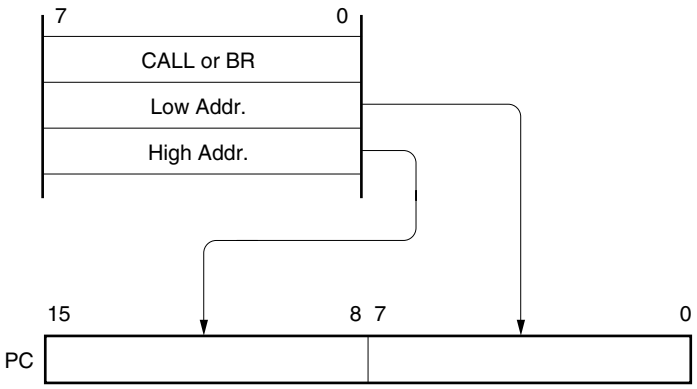
This function is carried out when the CALL !addr16 or BR !addr16 or CALLF !addr11 instruction is executed.

CALL !addr16 and BR !addr16 instructions can be branched to the entire memory space.

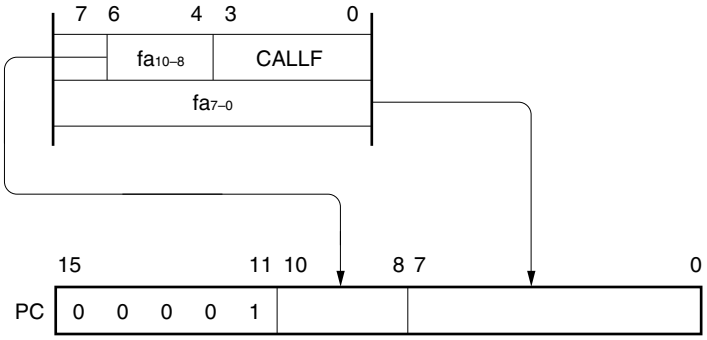
The CALLF !addr11 instruction is branched to the 0800H to 0FFFH area.

[Illustration]

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



In the case of CALLF !addr11 instruction



3.3.3 Table indirect addressing

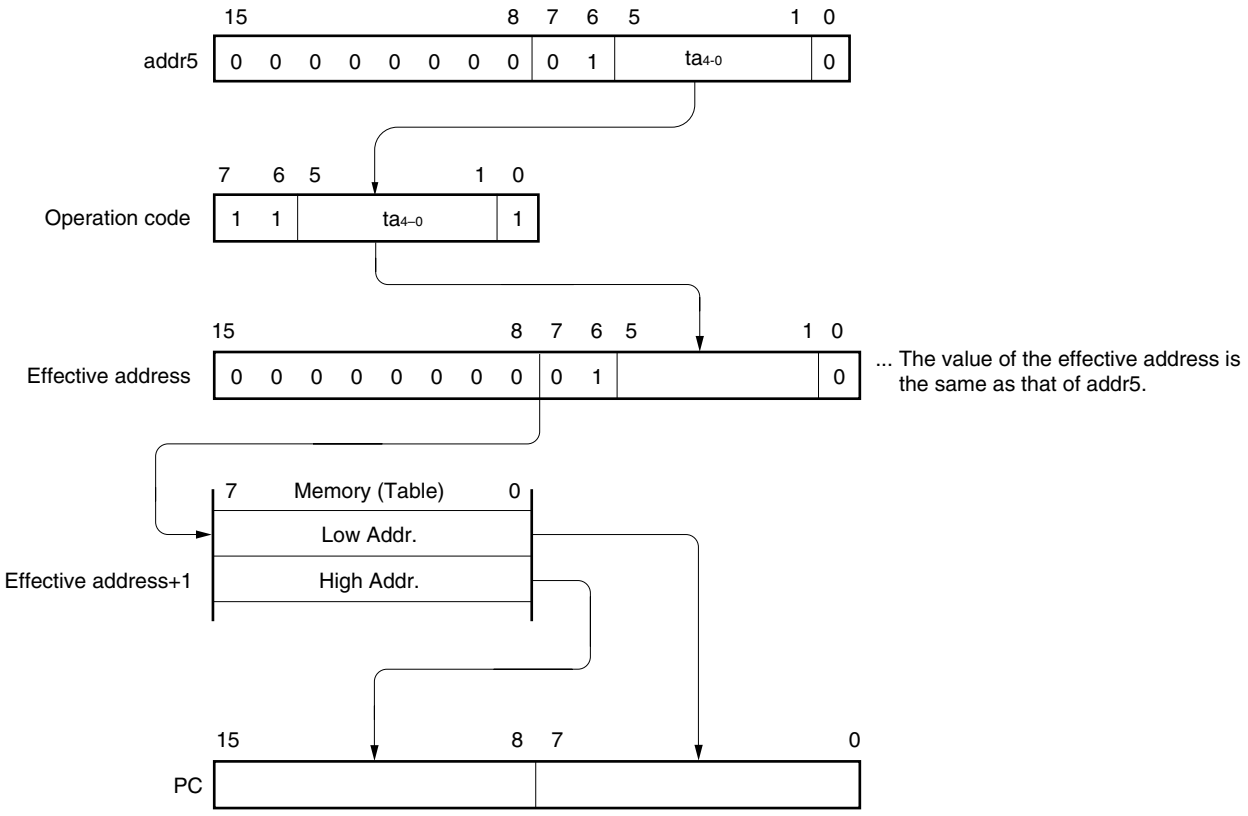
[Function]

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 stored in the memory table from 40H to 7FH, and allows branching to the entire memory space.

[Illustration]



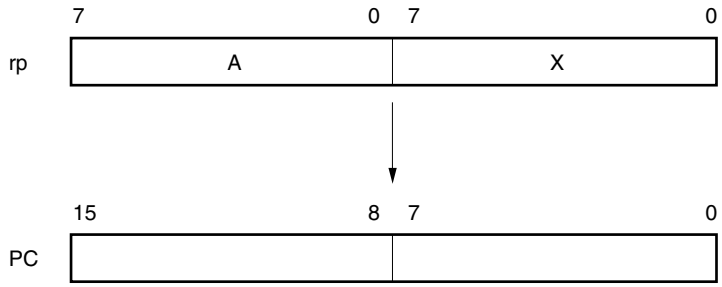
3.3.4 Register addressing

[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 µPD78F0730 instruction words, the following instructions employ implied addressing.

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

[Operand format]

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

[Description example]

In the case of MULU X

With an 8-bit × 8-bit multiply instruction, the product of 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.

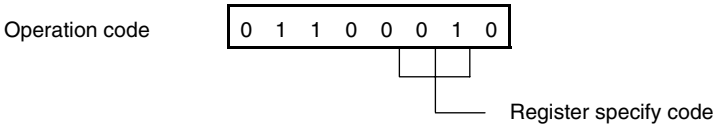
[Operand format]

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

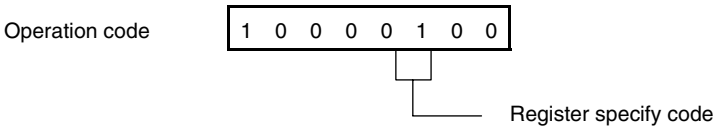
'r' and 'rp' can be described by absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

[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]

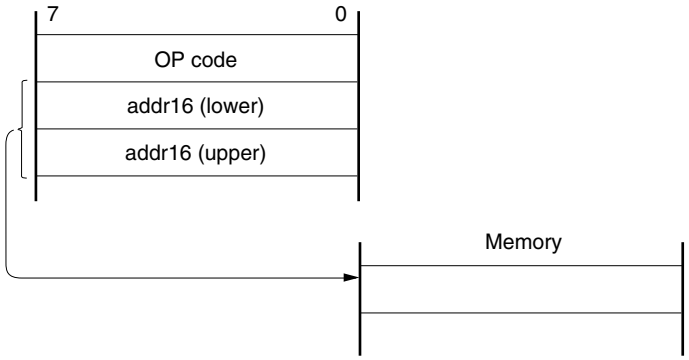
Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

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

Operation code	1 0 0 0 1 1 1 0	OP code
	0 0 0 0 0 0 0 0	00H
	1 1 1 1 1 1 1 0	FEH

[Illustration]



3.4.4 Short direct addressing

[Function]

The memory to be manipulated in the fixed space is directly addressed with 8-bit data in an instruction word.

This addressing is applied to the 256-byte space FE20H to FF1FH. Internal high-speed RAM and special function registers (SFRs) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

The SFR area (FF00H to FF1FH) where short direct addressing is applied is a part of the overall SFR area. Ports that are frequently accessed in a program and compare and capture registers of the timer/event counter are mapped in this area, allowing SFRs to be manipulated with a small number of bytes and clocks.

When 8-bit immediate data is at 20H to FFH, bit 8 of an effective address is set to 0. When it is at 00H to 1FH, bit 8 is set to 1. See the **[Illustration]** shown below.

[Operand format]

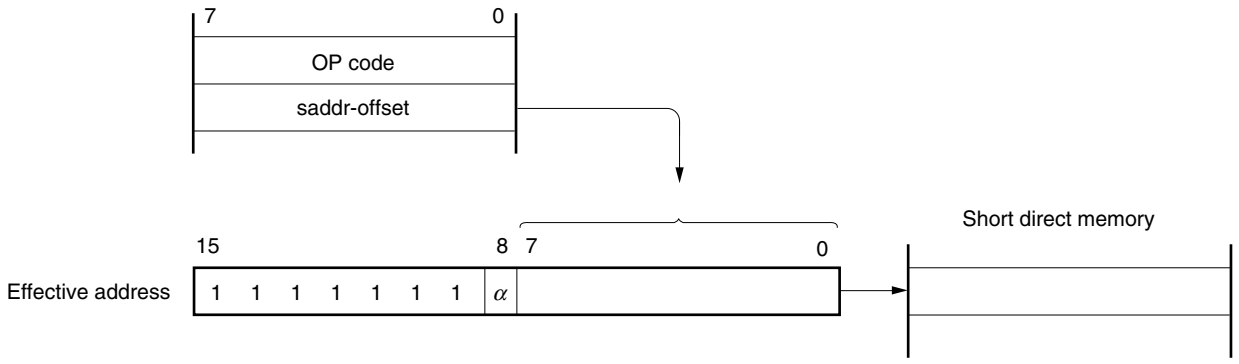
Identifier	Description
saddr	Immediate data that indicate label or FE20H to FF1FH
saddrp	Immediate data that indicate label or FE20H to FF1FH (even address only)

[Description example]

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

Operation code	1 1 1 1 0 0 1 0	OP code
	0 0 1 1 0 0 0 0	30H (saddr-offset)

[Illustration]



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

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

3.4.5 Special function register (SFR) addressing

[Function]

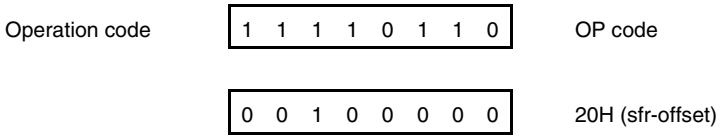
A memory-mapped special function register (SFR) is addressed with 8-bit immediate data in an instruction word. This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, the SFRs mapped at FF00H to FF1FH can be accessed with short direct addressing.

[Operand format]

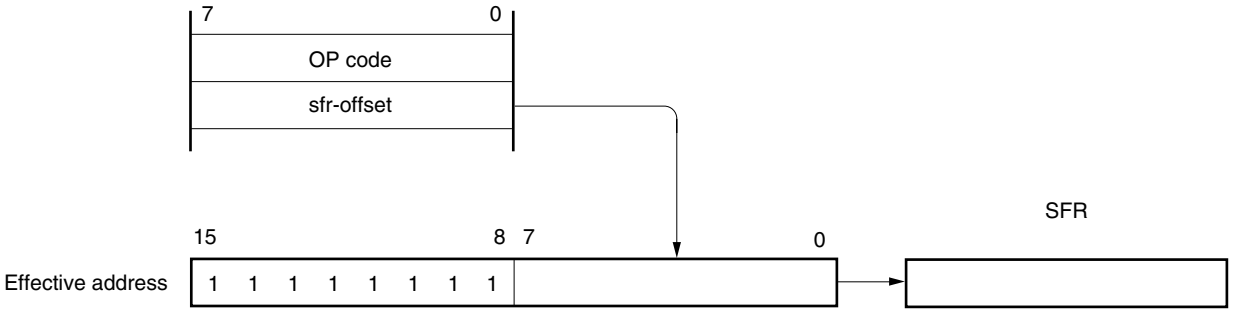
Identifier	Description
sfr	Special function register name
sfrp	16-bit manipulatable special function register name (even address only)

[Description example]

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



[Illustration]



3.4.6 Register indirect addressing

[Function]

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

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

[Operand format]

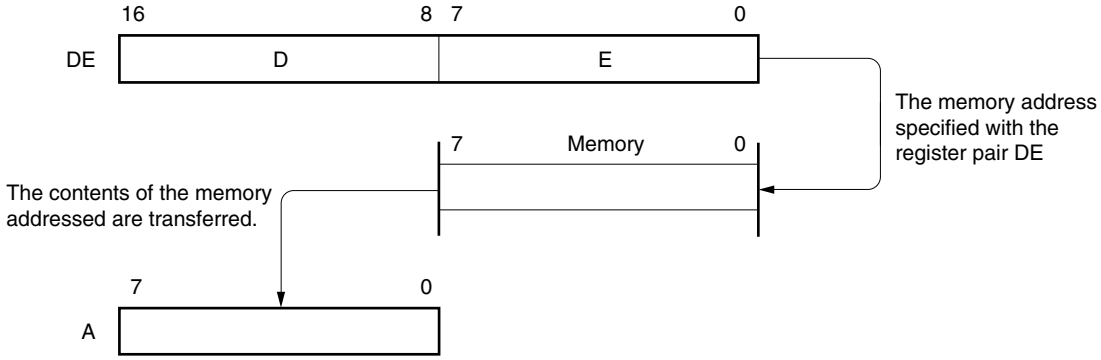
Identifier	Description
–	[DE], [HL]

[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]

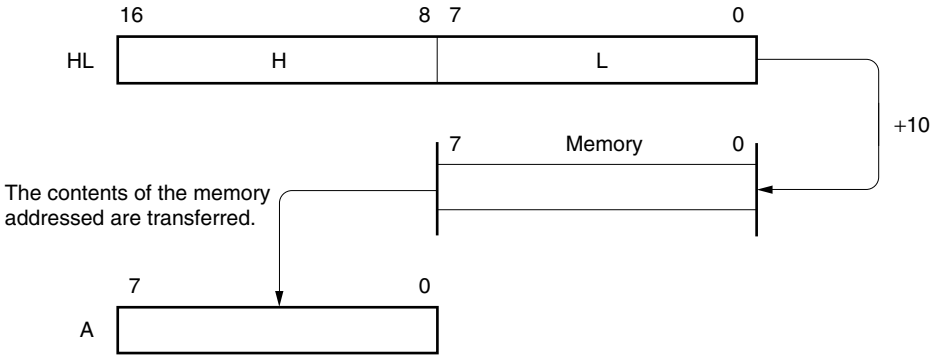
MOV A, [HL + 10H]; when setting byte to 10H

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]

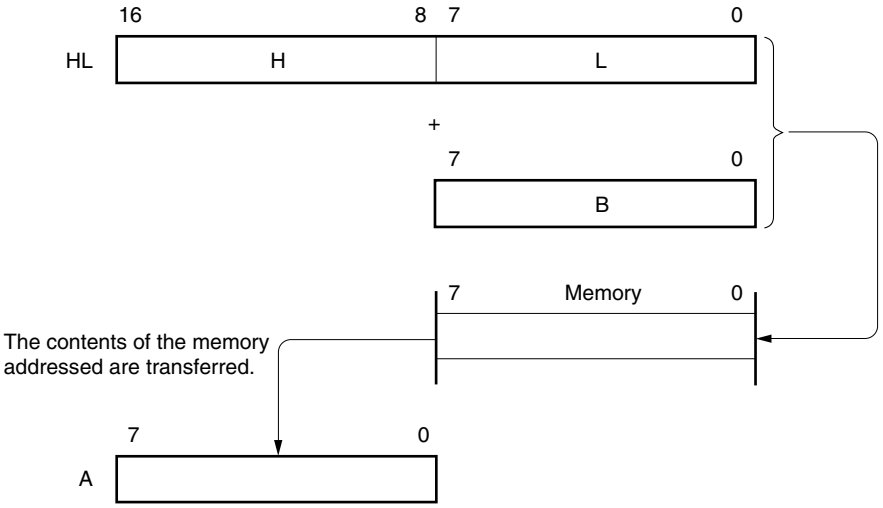
Identifier	Description
–	[HL + B], [HL + C]

[Description example]

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

Operation code	1	0	1	0	1	0	1	1

[Illustration]



3.4.9 Stack addressing

[Function]

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

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

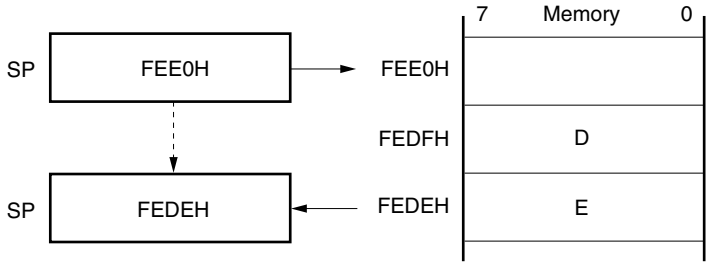
With stack addressing, only the internal high-speed RAM area can be accessed.

[Description example]

PUSH DE; when saving DE register

Operation code	1	0	1	1	0	1	0	1

[Illustration]



CHAPTER 4 PORT FUNCTIONS

4.1 Port Functions

There are two types of pin I/O buffer power supplies: EV_{DD} and V_{DD}. The relationship between these power supplies and the pins is shown below.

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

Power Supply	Corresponding Pins
EV_{DD}	Port pins other than P121 and P122
V_{DD}	• P121 and P122 • Non-port pins

The μPD78F0730 is provided with the ports shown in Figure 4-1, which enable variety of control operations. The functions of each port are shown in Table 4-2.

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

Figure 4-1. Port Types

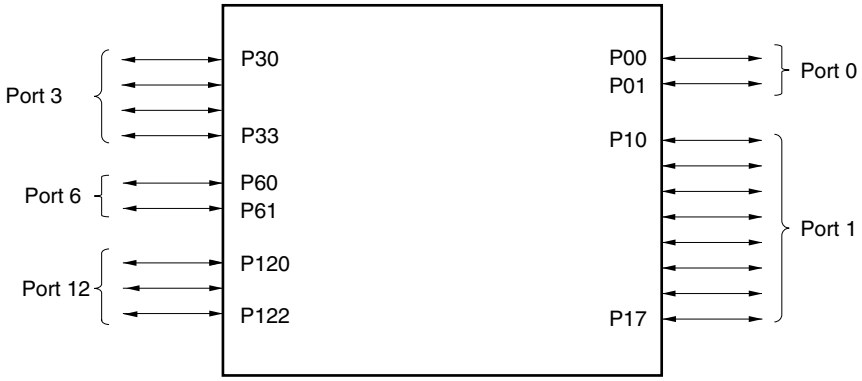


Table 4-2. Port Functions

Function Name	I/O	Function	After Reset	Alternate Function
P00	I/O	Port 0. 2-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting.	Input port	TI000
P01				TI010/TO00
P10	I/O	Port 1. 8-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting.	Input port	$\overline{SCK10}$
P11				SI10
P12				SO10
P13				TxD6
P14				RxD6
P15				–
P16				TOH1
P17				TI50/TO50
P30	I/O	Port 3. 4-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting.	Analog input	INTP1
P31				INTP2/OCD1A
P32				INTP3/OCD1B
P33				TI51/TO51
P60	I/O	Port 6. 2-bit I/O port. Output of P60 and P61 is N-ch open-drain output (6 V tolerance). Input/output can be specified in 1-bit units.	Input port	–
P61				–
P120	I/O	Port 12. 3-bit I/O port. Input/output can be specified in 1-bit units. Only for P120, use of an on-chip pull-up resistor can be specified by a software setting.	Input port	INTP0
P121				X1/OCD0A
P122				X2/EXCLK/OCD0B

4.2 Port Configuration

Ports include the following hardware.

Table 4-3. Port Configuration

Item	Configuration
Control registers	Port mode register (PM0, PM1, PM3, PM6, PM12) Port register (P0, P1, P3, P6, P12) Pull-up resistor option register (PU0, PU1, PU3, PU12)
Port	Total: 19 (CMOS I/O: 17, N-ch open drain I/O: 2)
Pull-up resistor	Total: 15

4.2.1 Port 0

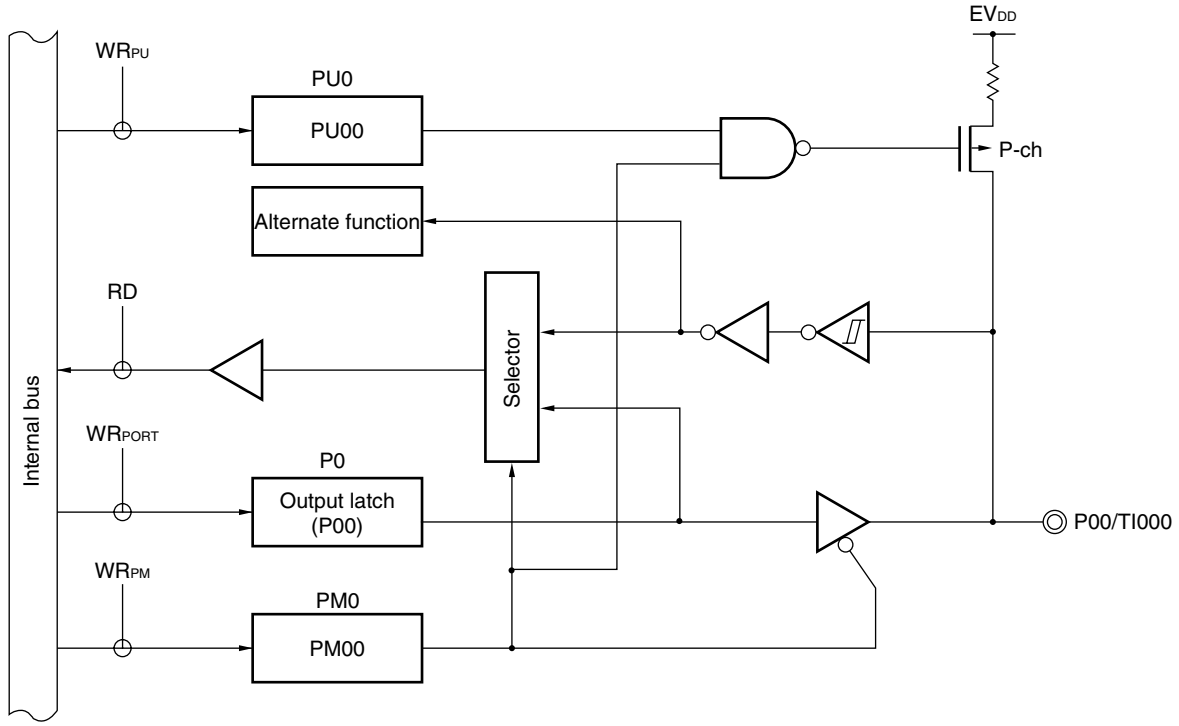
Port 0 is a 2-bit I/O port with an output latch. Port 0 can be set to the input mode or output mode in 1-bit units using port mode register 0 (PM0). When the P00 and P01 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 0 (PU0).

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

Reset signal generation sets port 0 to input mode.

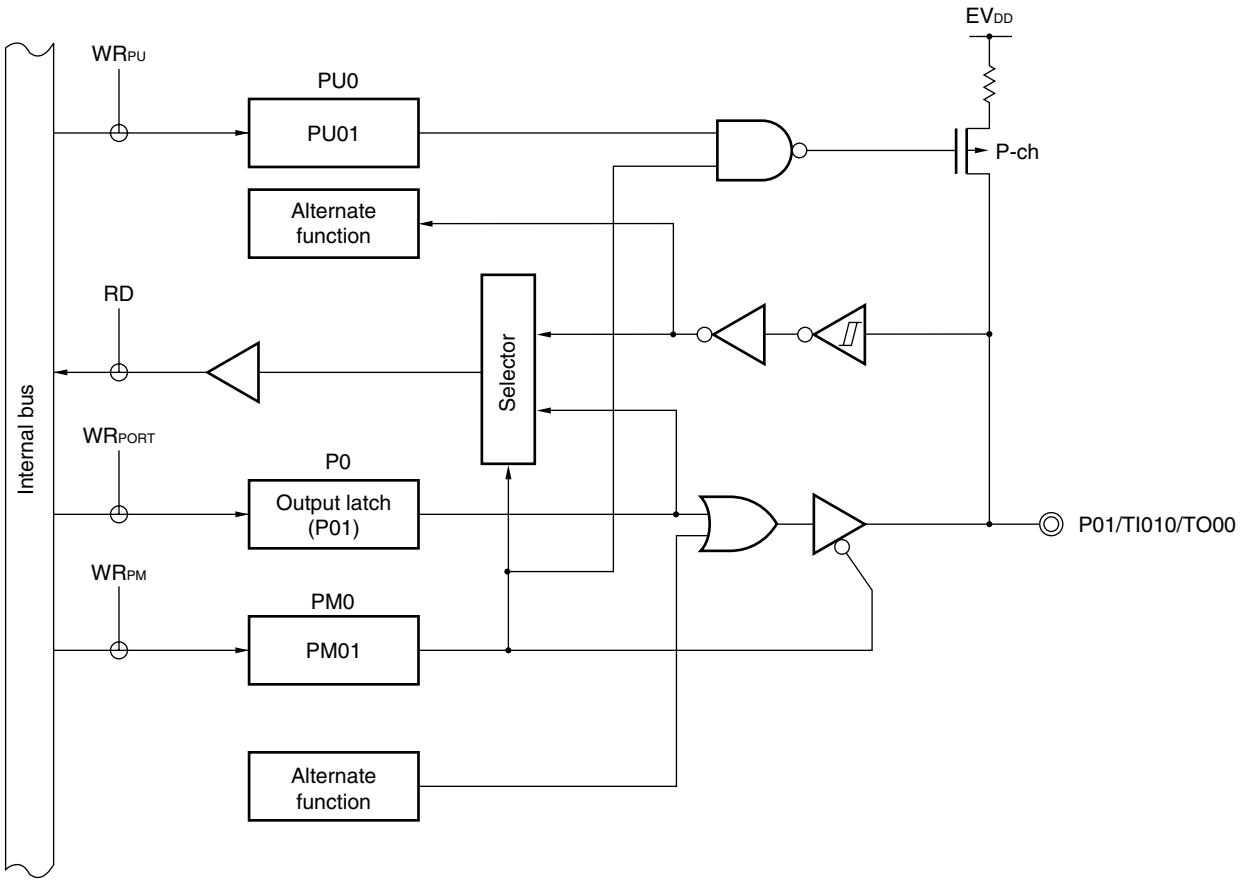
Figures 4-2 and 4-3 show block diagrams of port 0.

Figure 4-2. Block Diagram of P00



P0: Port register 0
PU0: Pull-up resistor option register 0
PM0: Port mode register 0
RD: Read signal
WR××: Write signal

Figure 4-3. Block Diagram of P01



P0: Port register 0
PU0: Pull-up resistor option register 0
PM0: Port mode register 0
RD: Read signal
WR××: Write signal

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).

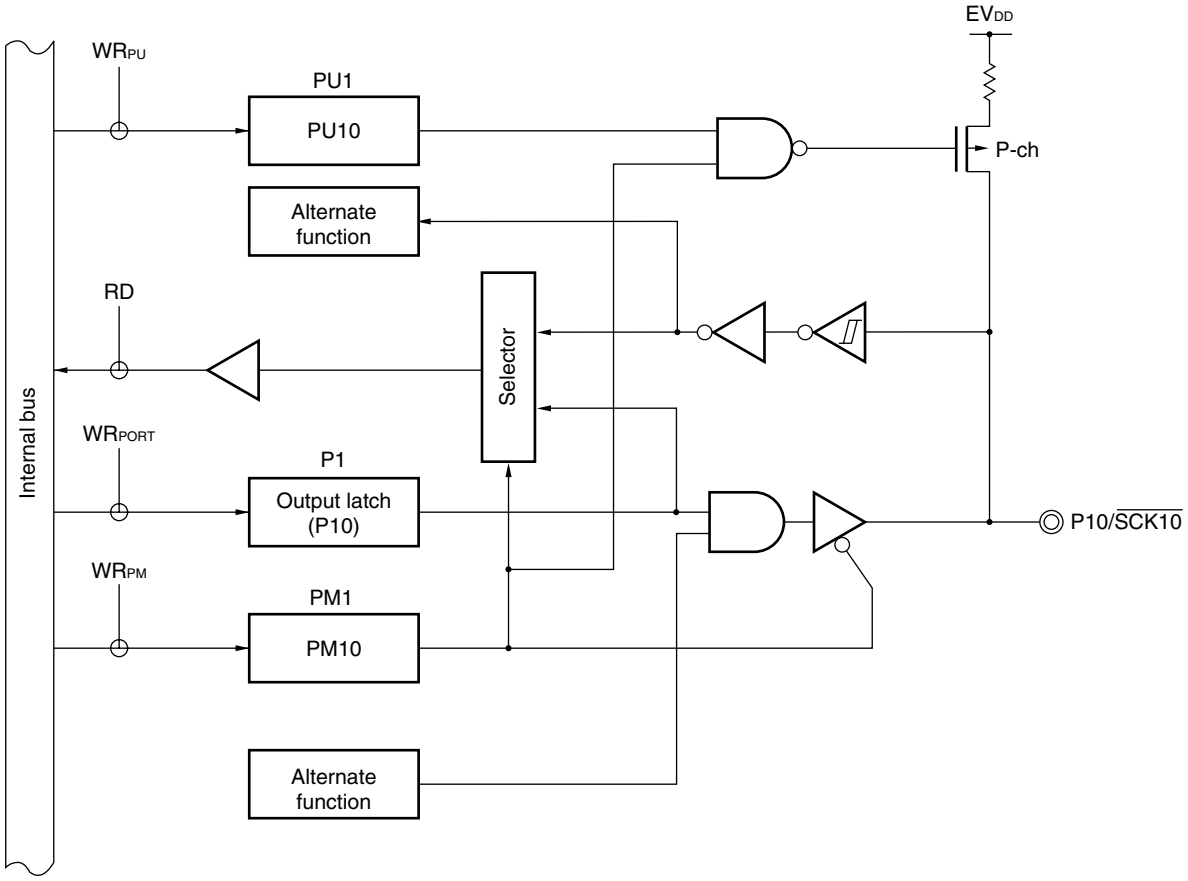
This port can also be used for serial interface data I/O, clock I/O, and timer I/O.

Reset signal generation sets port 1 to input mode.

Figures 4-4 to 4-9 show block diagrams of port 1.

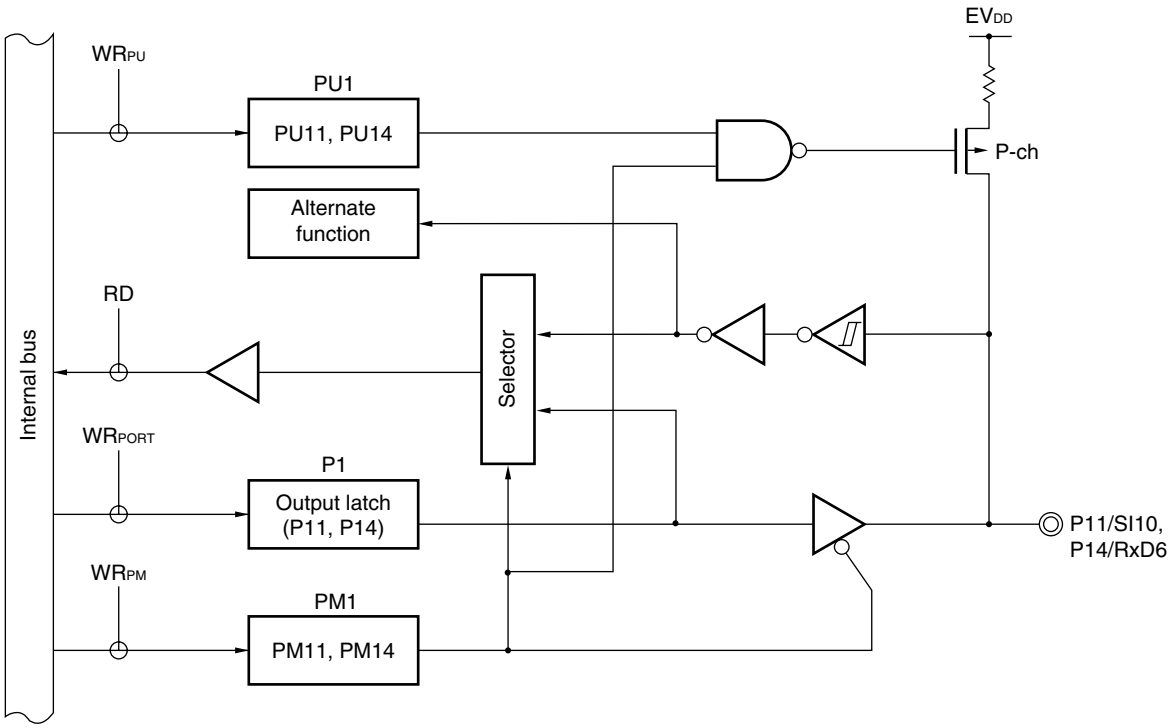
Caution To use P10/$\overline{\text{SCK10}}$ and P12/SO10 as general-purpose ports, set serial operation mode register 10 (CSIM10) and serial clock selection register 10 (CSIC10) to the default status (00H).

Figure 4-4. Block Diagram of P10



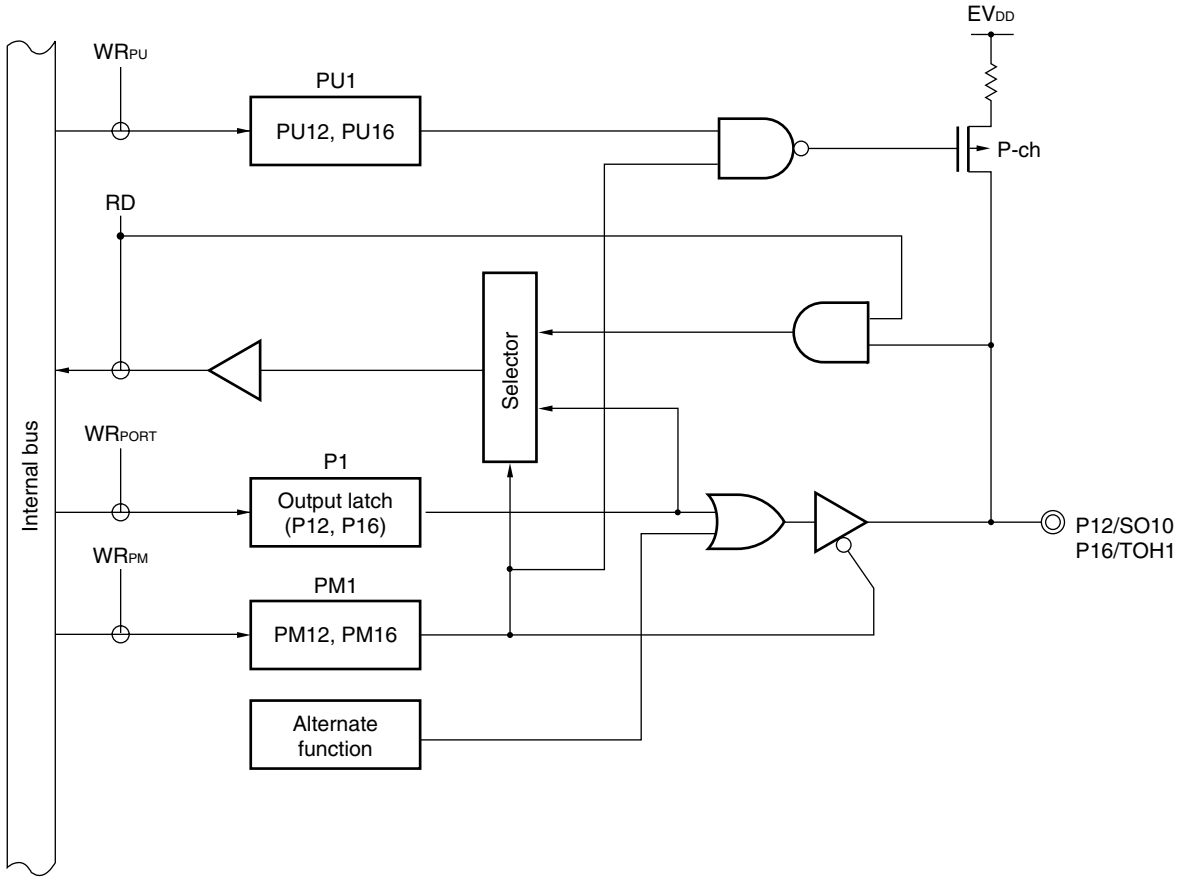
P1: Port register 1
PU1: Pull-up resistor option register 1
PM1: Port mode register 1
RD: Read signal
WR××: Write signal

Figure 4-5. Block Diagram of P11 and P14



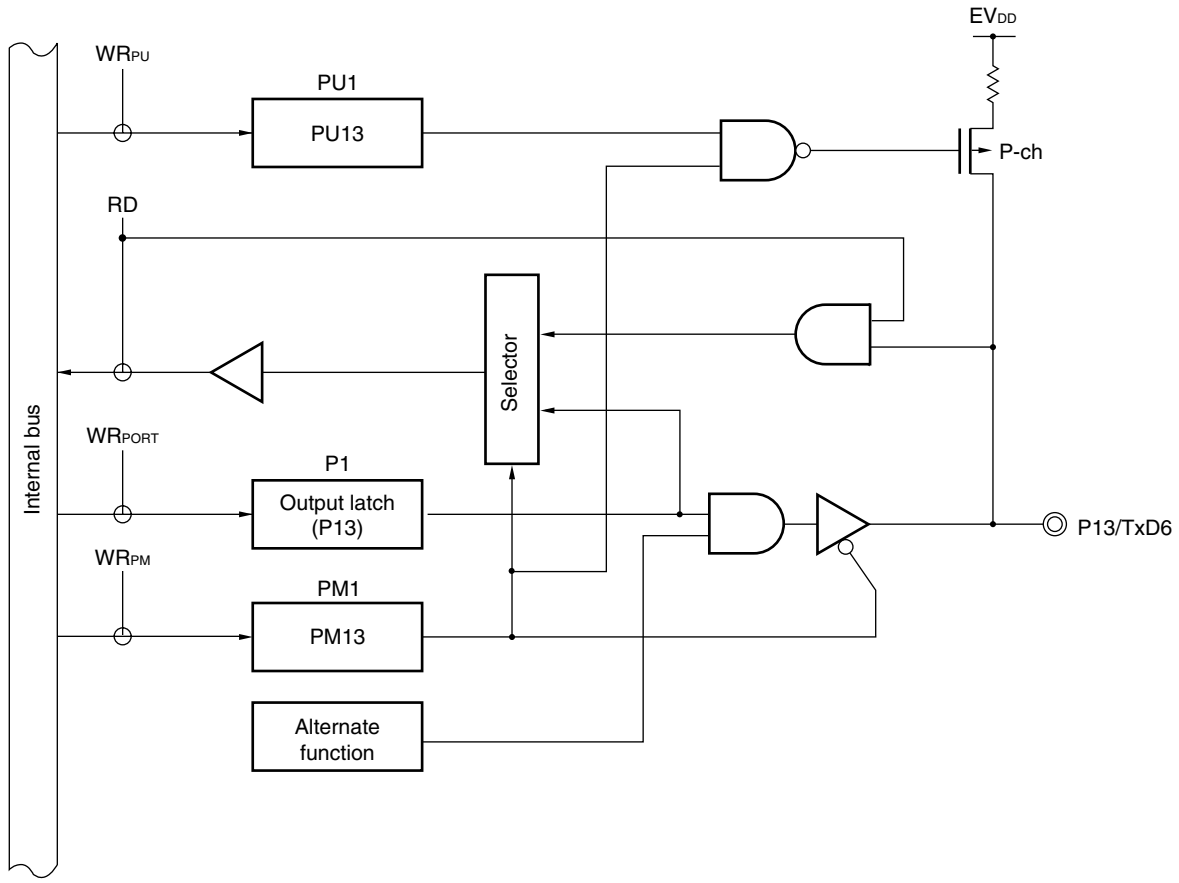
P1: Port register 1
PU1: Pull-up resistor option register 1
PM1: Port mode register 1
RD: Read signal
WR××: Write signal

Figure 4-6. Block Diagram of P12 and P16



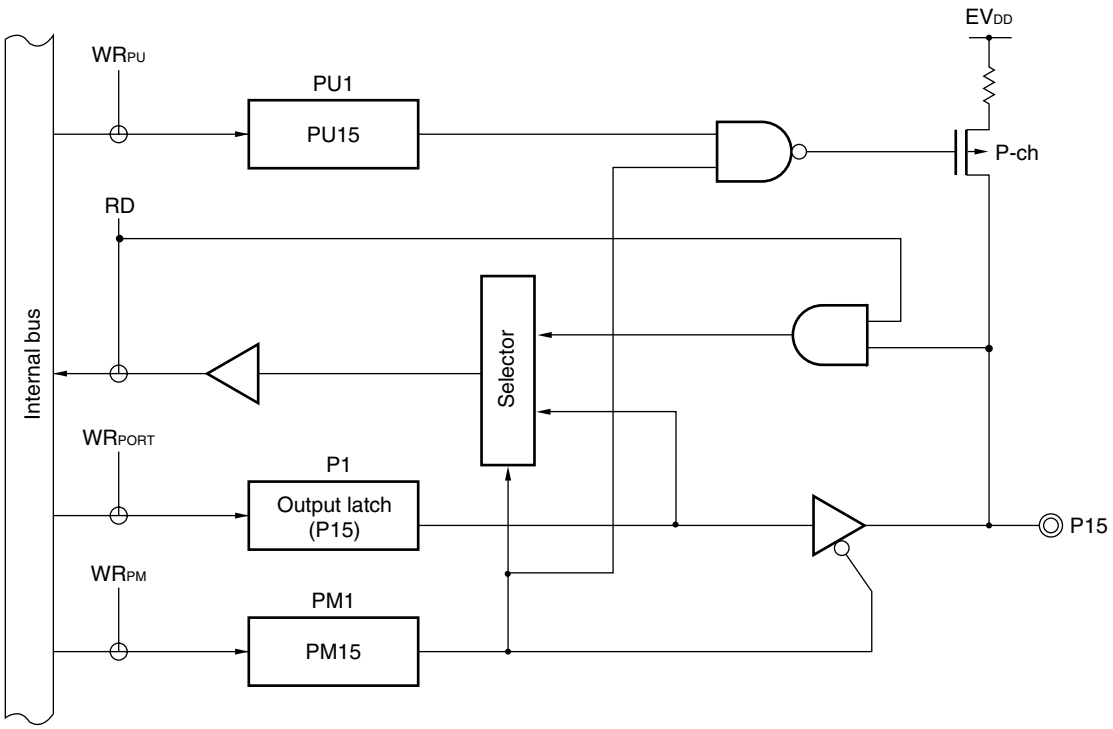
P1: Port register 1
PU1: Pull-up resistor option register 1
PM1: Port mode register 1
RD: Read signal
WR××: Write signal

Figure 4-7. Block Diagram of P13



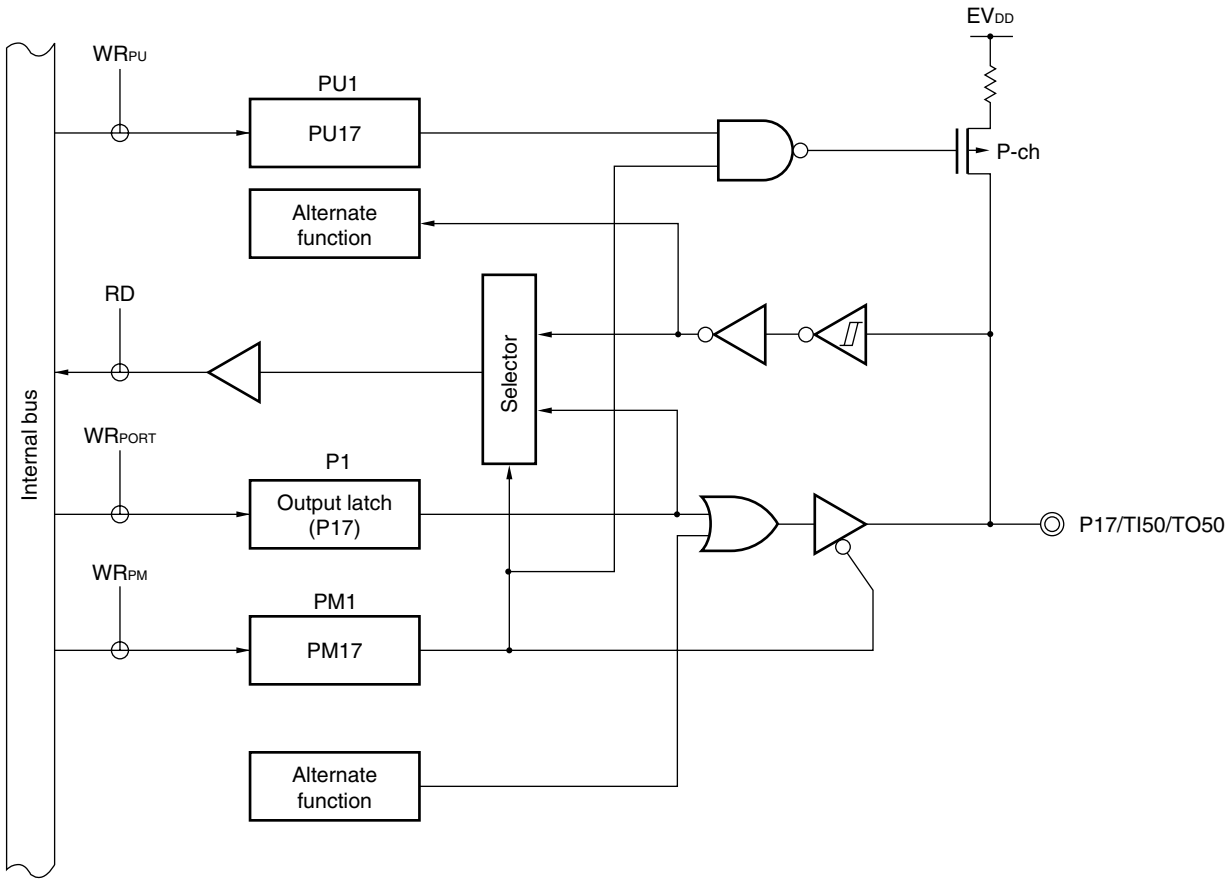
P1: Port register 1
PU1: Pull-up resistor option register 1
PM1: Port mode register 1
RD: Read signal
WR××: Write signal

Figure 4-8. Block Diagram of P15



P1: Port register 1
PU1: Pull-up resistor option register 1
PM1: Port mode register 1
RD: Read signal
WR××: Write signal

Figure 4-9. Block Diagram of P17



P1: Port register 1
PU1: Pull-up resistor option register 1
PM1: Port mode register 1
RD: Read signal
WR××: Write signal

4.2.3 Port 3

Port 3 is a 4-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 3 (PM3). When the P30 to P33 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 3 (PU3).

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

Reset signal generation sets port 3 to input mode.

Figures 4-10 and 4-11 show block diagrams of port 3.

Cautions 1. Be sure to pull the P31 pin down before a reset release to prevent malfunction.

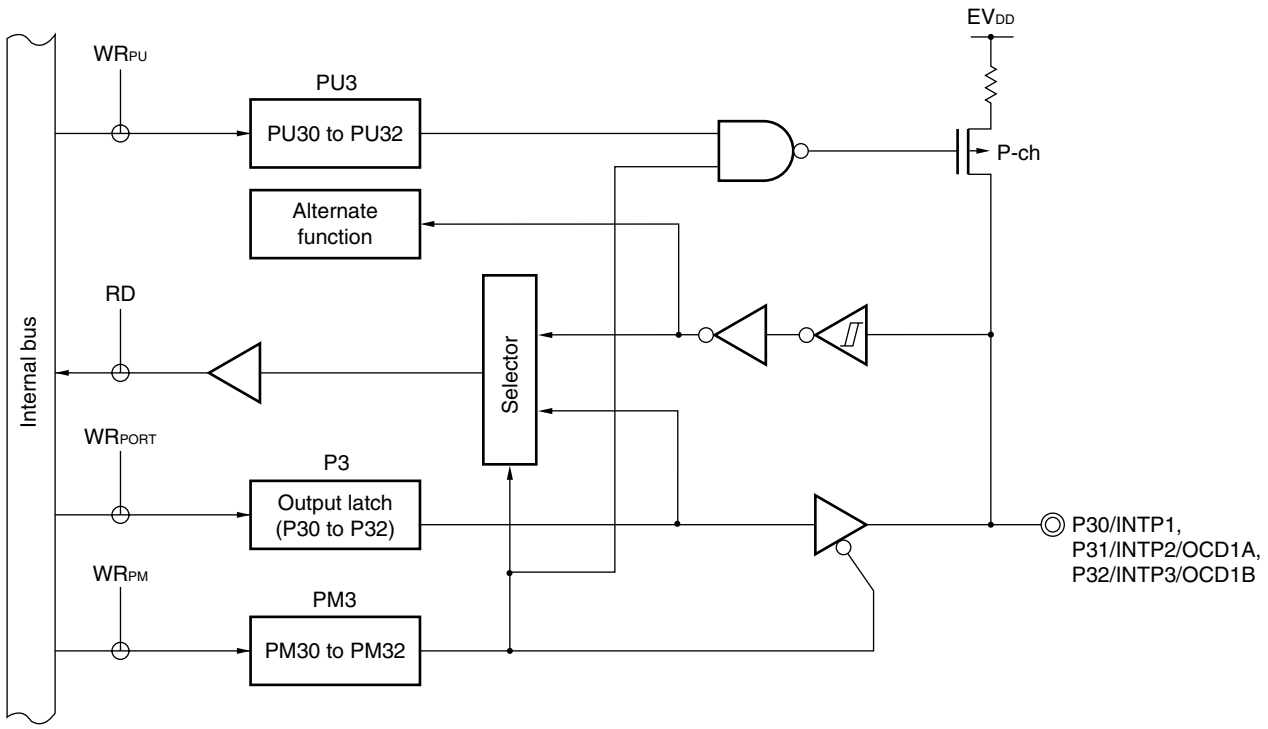
2. When writing the flash memory with a flash memory programmer, connect P31/INTP2/OCD1A as follows.

• P31/INTP2/OCD1A: Connect to EV_{SS} via a resistor (10 kΩ: recommended).

The above connection is not necessary when writing the flash memory by means of self programming.

Remark The P31 and P32 pins of the µPD78F0730 can be used as on-chip debug mode setting pins (OCD1A, OCD1B) when the on-chip debug function is used. For details, see **CHAPTER 20 ON-CHIP DEBUG FUNCTION.**

Figure 4-10. Block Diagram of P30 to P32



P3: Port register 3

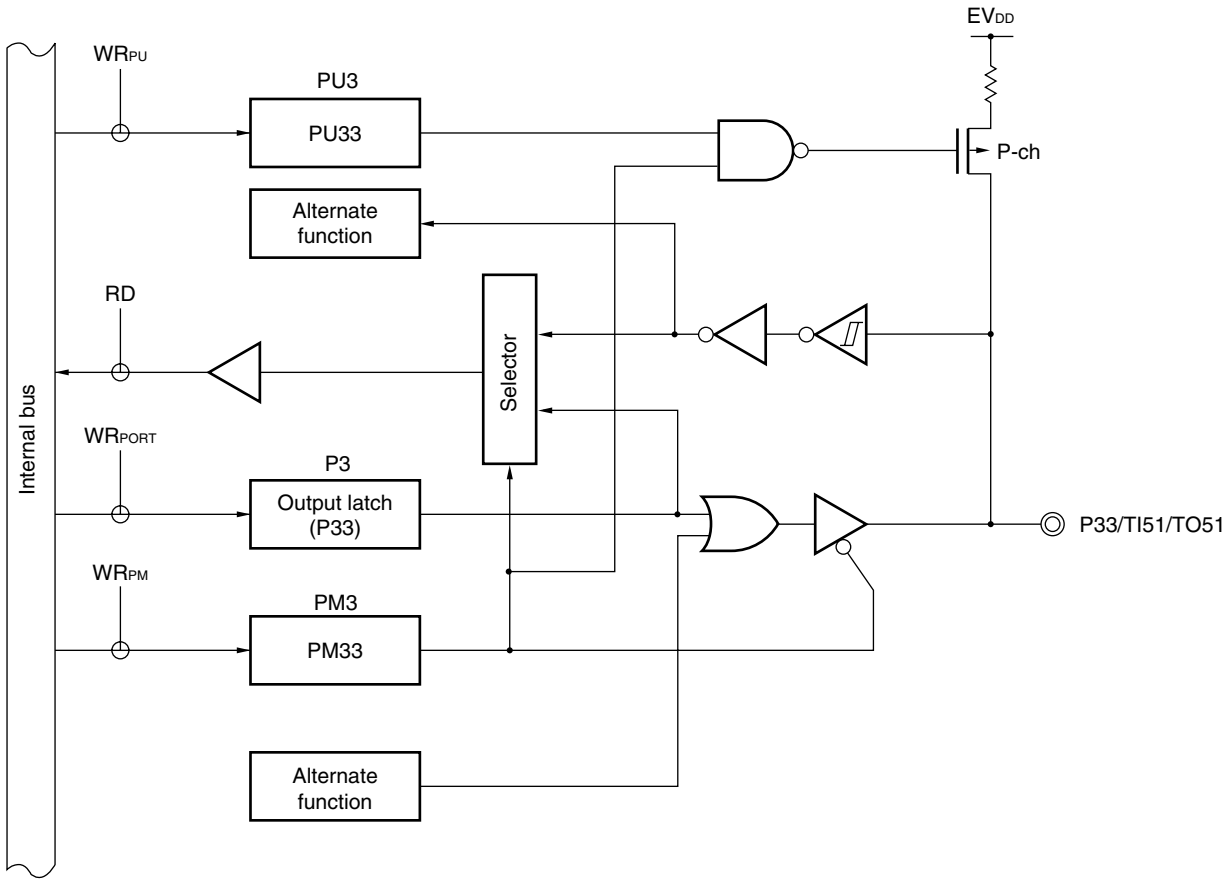
PU3: Pull-up resistor option register 3

PM3: Port mode register 3

RD: Read signal

WR××: Write signal

Figure 4-11. Block Diagram of P33



P3: Port register 3
PU3: Pull-up resistor option register 3
PM3: Port mode register 3
RD: Read signal
WR××: Write signal

4.2.4 Port 6

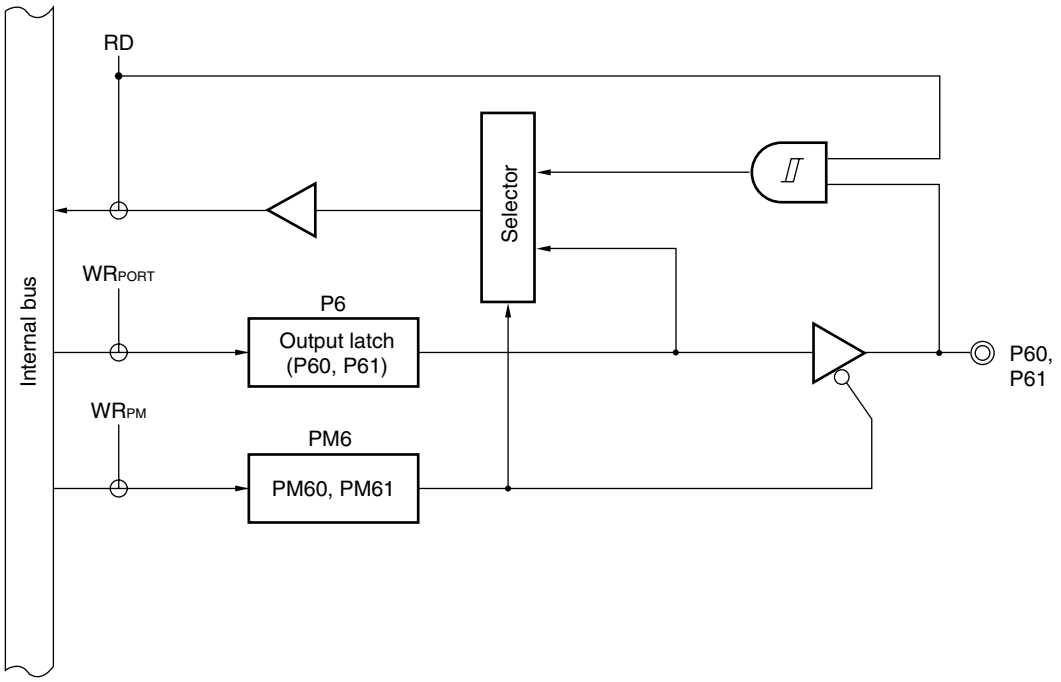
Port 6 is a 2-bit I/O port with an output latch. Port 6 can be set to the input mode or output mode in 1-bit units using port mode register 6 (PM6).

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

Reset signal generation sets port 6 to input mode.

Figure 4-12 shows a block diagram of port 6.

Figure 4-12. Block Diagram of P60 and P61



P6: Port register 6
PM6: Port mode register 6
RD: Read signal
WR××: Write signal

4.2.5 Port 12

Port 12 is a 3-bit I/O port with an output latch. Port 12 can be set to the input mode or output mode in 1-bit units using port mode register 12 (PM12). When used as an input port only for P120, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 12 (PU12).

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

Reset signal generation sets port 12 to input mode.

Figures 4-13 and 4-14 show block diagrams of port 12.

Cautions 1. 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.

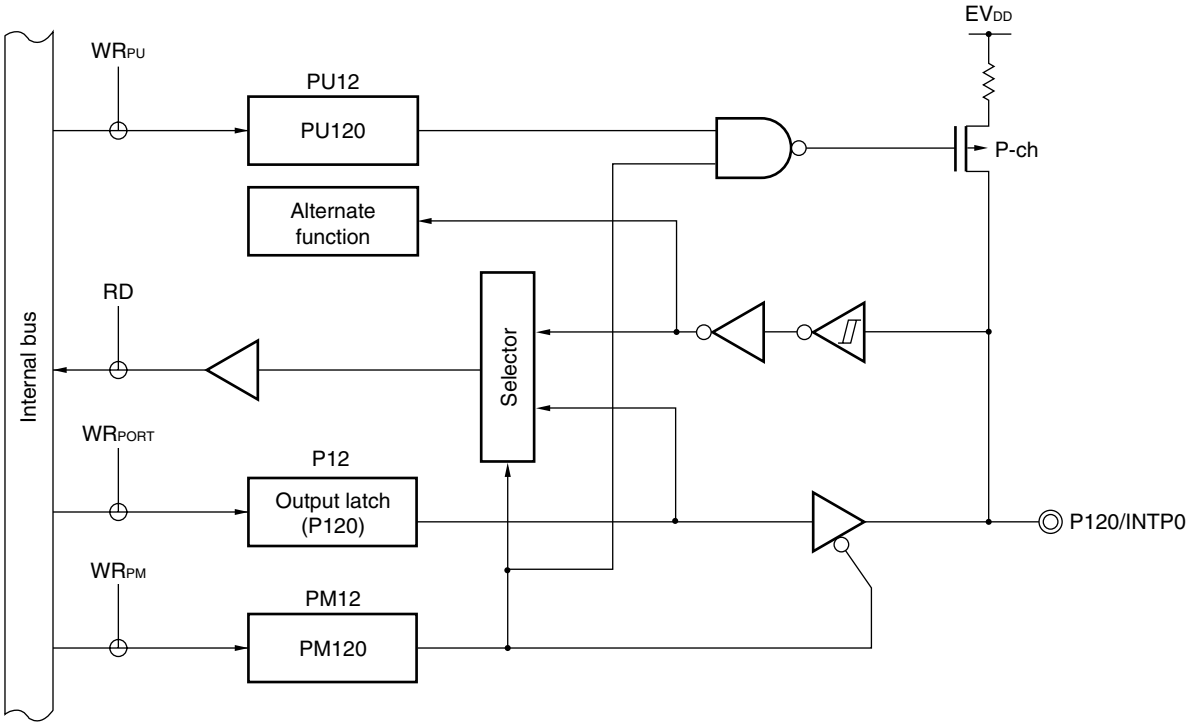
2. When writing the flash memory with a flash memory programmer, connect P121/X1/OCD0A as follows.

• P121/X1/OCD0A: When using this pin as a port, connect it to V_{SS} via a resistor (10 kΩ: recommended) (in the input mode) or leave it open (in the output mode).

The above connection is not necessary when writing the flash memory by means of self programming.

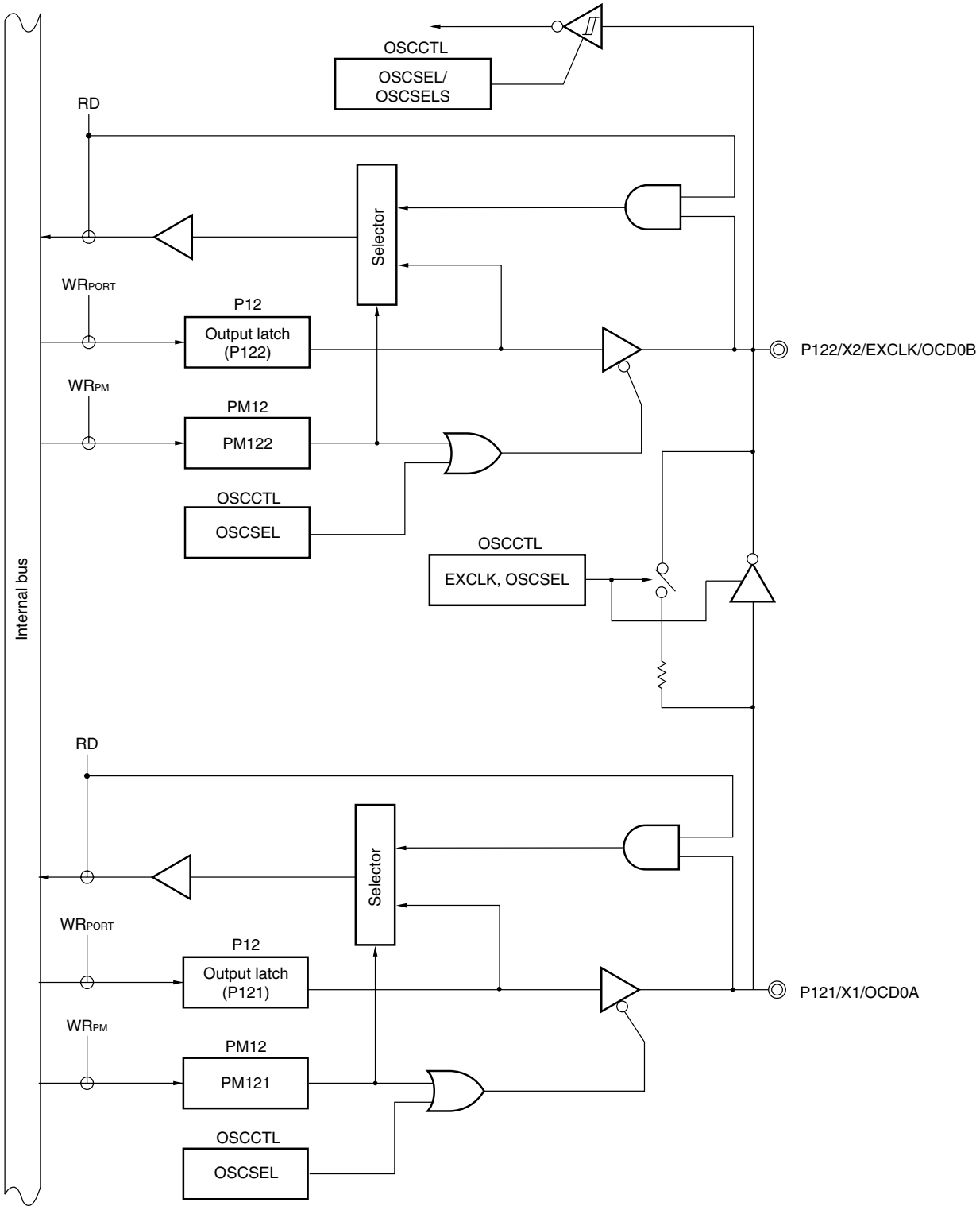
Remark The X1 and X2 pins of the µPD78F0730 can be used as on-chip debug mode setting pins (OCD0A, OCD0B) when the on-chip debug function is used. For details, see **CHAPTER 20 ON-CHIP DEBUG FUNCTION.**

Figure 4-13. Block Diagram of P120



P12: Port register 12
PU12: Pull-up resistor option register 12
PM12: Port mode register 12
RD: Read signal
WR××: Write signal

Figure 4-14. Block Diagram of P121 and P122



P12: Port register 12
PM12: Port mode register 12
OSCCTL: Clock operation mode select register
RD: Read signal
WR××: Write signal

4.3 Registers Controlling Port Function

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

- Port mode registers (PM0, PM1, PM3, PM6, PM12)
- Port registers (P0, P1, P3, P6, P12)
- Pull-up resistor option registers (PU0, PU1, PU3, PU12)

(1) Port mode registers (PM0, PM1, PM3, PM6, and 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 and Output Latch When Using Alternate Function**.

Figure 4-15. Format of Port Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	1	1	1	1	1	1	PM01	PM00	FF20H	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FF21H	FFH	R/W
PM3	1	1	1	1	PM33	PM32	PM31	PM30	FF23H	FFH	R/W
PM6	1	1	1	1	1	1	PM61	PM60	FF26H	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, 1, 3, 6, 12; n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

(2) Port registers (P0, P1, P3, P6, 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.

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
P3	0	0	0	0	P33	P32	P31	P30	FF03H	00H (output latch)	R/W
P6	0	0	0	0	0	0	P61	P60	FF06H	00H (output latch)	R/W
P12	0	0	0	0	0	P122	P121	P120	FF0CH	00H (output latch)	R/W

Pmn	m = 0, 1, 3, 6, 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

(3) Pull-up resistor option registers (PU0, PU1, PU3, and PU12)

These registers specify whether the on-chip pull-up resistors of P00 and P01, P10 to P17, P30 to P33, or P120 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, 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, and PU12.

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

Reset signal generation sets these registers to 00H.

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	0	0	0	0	0	0	PU01	PU00	FF30H	00H	R/W
PU1	PU17	PU16	PU15	PU14	PU13	PU12	PU11	PU10	FF31H	00H	R/W
PU3	0	0	0	0	PU33	PU32	PU31	PU30	FF33H	00H	R/W
PU12	0	0	0	0	0	0	0	PU120	FF3CH	00H	R/W

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

4.4 Port Function Operations

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

Caution In the case of 1-bit memory manipulation instruction, although a single bit is manipulated, the port is accessed as an 8-bit unit. Therefore, on a port with a mixture of input and output pins, the output latch contents for pins specified as input are undefined, even for bits other than the manipulated bit.

4.4.1 Writing to I/O port

(1) Output mode

A value is written to the output latch by a transfer instruction, and the output latch contents are output from the pin.
Once data is written to the output latch, it is retained until data is written to the output latch again.
The data of the output latch is cleared when a reset signal is generated.

(2) Input mode

A value is written to the output latch by a transfer instruction, but since the output buffer is off, the pin status does not change.
Once data is written to the output latch, it is retained until data is written to the output latch again.
The data of the output latch is cleared when a reset signal is generated.

4.4.2 Reading from I/O port

(1) Output mode

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

(2) Input mode

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

4.4.3 Operations on I/O port

(1) Output mode

An operation is performed on the output latch contents, and the result is written to the output latch. The output latch contents are output from the pins.
Once data is written to the output latch, it is retained until data is written to the output latch again.
The data of the output latch is cleared when a reset signal is generated.

(2) Input mode

The pin level is read and an operation is performed on its contents. The result of the operation is written to the output latch, but since the output buffer is off, the pin status does not change.
The data of the output latch is cleared when a reset signal is generated.

4.5 Settings of Port Mode Register and Output Latch When Using Alternate Function

To use the alternate function of a port pin, set the port mode register and output latch as shown in Table 4-5.

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

Pin Name	Alternate Function		PM××	P××
	Function Name	I/O		
P00	TI000	Input	1	×
P01	TI010	Input	1	×
	TO00	Output	0	0
P10	$\overline{SCK10}$	Input	1	×
		Output	0	1
P11	SI10	Input	1	×
P12	SO10	Output	0	0
P13	TxD6	Output	0	1
P14	RxD6	Input	1	×
P16	TOH1	Output	0	0
P17	TI50	Input	1	×
	TO50	Output	0	0
P30 to P32	INTP1 to INTP3	Input	1	×
P33	TI51	Input	1	×
	TO51	Output	0	0
P120	INTP0	Input	1	×
P121	X1Note	–	×	×
P122	X2Note	–	×	×
	EXCLKNote	Input	×	×

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

Remarks 1. ×: Don't care
PM××: Port mode register
P××: Port output latch

2. The X1, X2, P31, and P32 pins of the µPD78F0730 can be used as on-chip debug mode setting pins (OCD0A, OCD0B, OCD1A, OCD1B) when the on-chip debug function is used. For details, see **CHAPTER 20 ON-CHIP DEBUG FUNCTION.**

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 kinds of system clocks and clock oscillators are selectable.

(1) Main system clock

<1> X1 oscillator

This circuit oscillates a clock of f_X = 12 or 16 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} = 16 MHz (TYP.). After a reset release, the CPU always starts operating with this internal high-speed oscillation clock. Oscillation can be stopped by executing the STOP instruction or using the internal oscillation mode register (RCM).

An external main system clock (f_{EXCLK} = 12 or 16 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 MOC.
As the main system clock, a high-speed system clock (X1 clock or external main system clock) or internal high-speed oscillation clock can be selected by using the main clock mode register (MCM).

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

- **Internal low-speed oscillator**

  This circuit oscillates a clock of f_{RL} = 240 kHz (TYP.). After a reset release, the internal low-speed oscillation clock always starts operating.
  Oscillation can be stopped by using the internal oscillation mode register (RCM) when “internal low-speed oscillator can be stopped by software” is set by option byte.
  The internal low-speed oscillation clock cannot be used as the CPU clock. The following hardware operates with the internal low-speed oscillation clock.

  - Watchdog timer
  - 8-bit timer H1 (when f_{RL}, $f_{RL}/2^7$, or $f_{RL}/2^9$ 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

(3) USB clock

- **PLL**

  This circuit multiplies the clock generated by the X1 oscillator (f_X) or external main system clock (f_{EXCLK}) by 8 or 12.

  Multiplication ratio x8 or x12 can be selected using the PLLM bit of the PLL control register (PLLC), and operation of PLL is started or stopped by setting the PLLSTOP bit.

Remarks 1. f_X: X1 clock oscillation frequency

2. f_{EXCLK}: external main system clock frequency

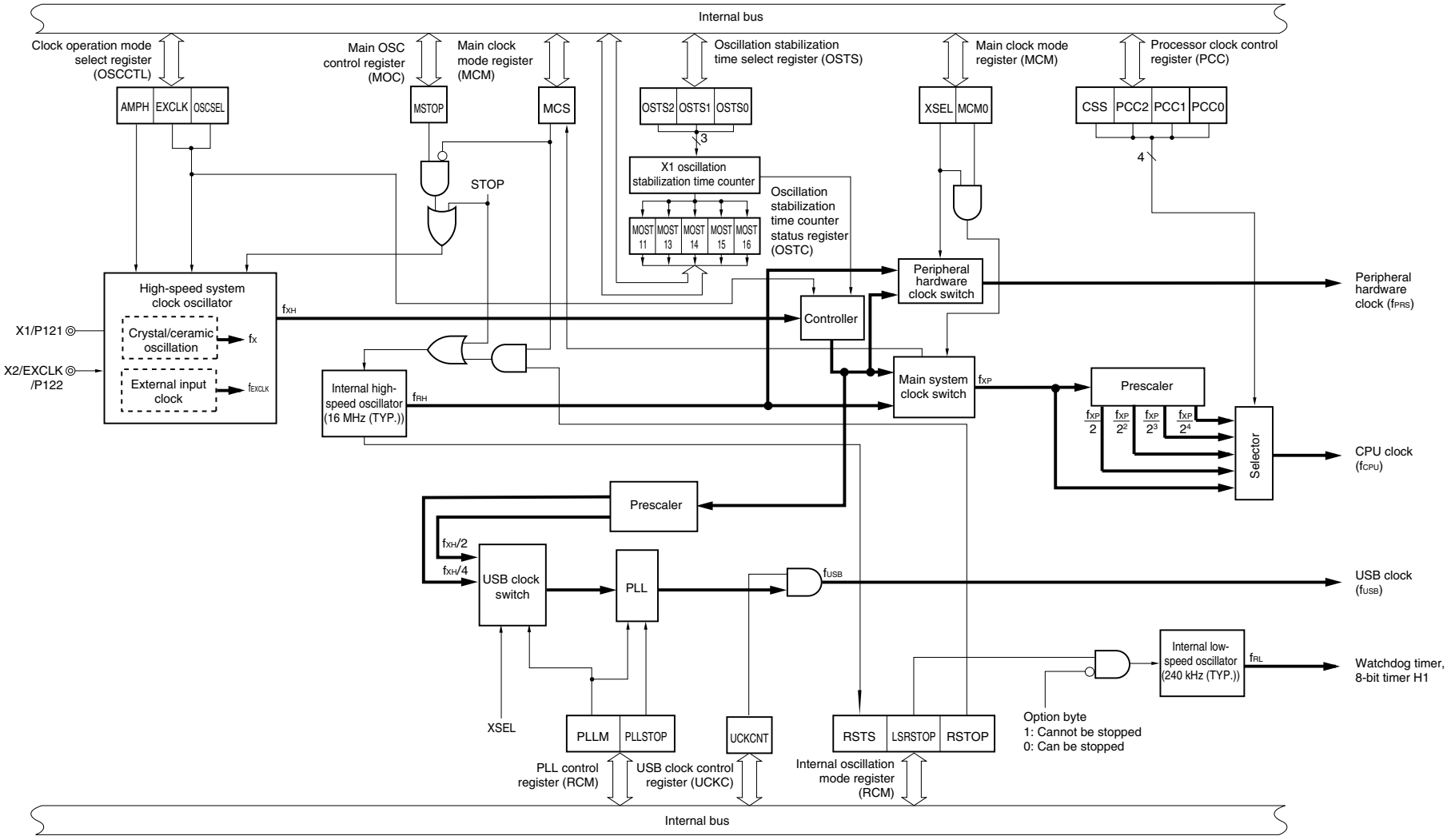
5.2 Configuration of Clock Generator

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) PLL control register (PLLC) USB clock control register (UCKC)
Oscillators	X1 oscillator Internal high-speed oscillator Internal low-speed oscillator

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 oscillation frequency
5. f_{XP}: Main system clock oscillation frequency
6. f_{PRS}: Peripheral hardware clock oscillation frequency
7. f_{CPU}: CPU clock oscillation frequency
8. f_{RL}: Internal low-speed oscillation clock frequency
9. f_{USB}: USB clock oscillation frequency

5.3 Registers Controlling Clock Generator

The following ten 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)
- PLL control register (PLLC)
- USB clock control register (UCKC)

(1) Clock operation mode select register (OSCCTL)

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

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

AMPH	Operating frequency control
0	$f_{XH} \leq 10$ MHz
1	10 MHz < f_{XH}

Cautions 1. Be sure to set AMPH to 1 if the high-speed system clock oscillation frequency exceeds 10 MHz.

2. Set AMPH before setting the peripheral functions after a reset release. The value of AMPH can be changed only once after a reset release. The clock supply to the CPU is stopped for 5 µs (MIN.) after AMPH has been set to 1.

3. If the STOP instruction is executed with AMPH set to 1 when the internal high-speed oscillation clock or external main system clock is used as the CPU clock, then the clock supply to the CPU is stopped for 5 µs (MIN.) after the STOP mode has been released. If the X1 clock is used as the CPU clock, oscillation stabilization time is counted after the STOP mode has been released.

4. 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_{XH}: High-speed system clock oscillation frequency

(2) Processor clock control register (PCC)

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

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

Reset signal generation sets PCC to 01H.

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

Address: FFFBH After reset: 01H R/W

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

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

Caution Be sure to clear bits 3 and 7 to 0.

Remark f_{XP}: Main system clock oscillation frequency

The fastest instruction can be executed in 2 clocks of the CPU clock in the µPD78F0730. Therefore, the relationship between the CPU clock (f_{CPU}) and the minimum instruction execution time is as shown in Table 5-2.

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

CPU Clock (f_{CPU})	Minimum Instruction Execution Time: $2/f_{CPU}$		
	High-Speed System Clock[Note]		Internal High-Speed Oscillation Clock[Note]
	At 12 MHz Operation	At 16 MHz Operation	At 16 MHz (TYP.) Operation
f_{XP}	0.167 µs	0.125 µs	0.125 µs (TYP.)
$f_{XP}/2$	0.333 µs	0.25 µs	0.25 µs (TYP.)
$f_{XP}/2^2$	0.667 µs	0.5 µs	0.5 µs (TYP.)
$f_{XP}/2^3$	1.33 µs	1.0 µs	1.0 µs (TYP.)
$f_{XP}/2^4$	2.67 µs	2.0 µs	2.0 µs (TYP.)

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

(3) Internal oscillation mode register (RCM)

This register sets the operation mode of internal oscillators.

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

Reset signal generation sets this register to 80H[Note 1].

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

Address: FFA0H After reset: 80H[Note 1] R/W[Note 2]

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

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

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

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

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

2. Bit 7 is read-only.

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

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

(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 MSTOP to 1 under the following condition.

- **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 sets this register to 00H.

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

Address: FFA1H After reset: 00H R/W[Note]

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

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

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

Note Bit 1 is read-only.

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

2. Be sure to set XSEL=1, MCM0=1 if using the USB function.

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

- **Watchdog timer (operates with internal low-speed oscillation clock)**
- **When "f_{RL}", "$f_{RL}/2^7$", or "$f_{RL}/2^9$" is selected as the count clock for 8-bit timer H1 (operates with internal low-speed oscillation clock)**
- **Peripheral hardware selects the external clock as the clock source (Except when the external count clock of TM00 is selected (TI000 pin valid edge))**

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

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

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

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

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

Address: FFA3H After reset: 00H R

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

MOST11	MOST13	MOST14	MOST15	MOST16	Oscillation stabilization time status		
						fx = 12 MHz	fx = 16 MHz
1	0	0	0	0	2^{11}/fx min.	170.7 µs min.	128 µs min.
1	1	0	0	0	2^{13}/fx min.	682.7 µs min.	512 µs min.
1	1	1	0	0	2^{14}/fx min.	1.37 ms min.	1.024 ms min.
1	1	1	1	0	2^{15}/fx min.	2.73 ms min.	2.048 ms min.
1	1	1	1	1	2^{16}/fx min.	5.46 ms min.	4.096 ms min.

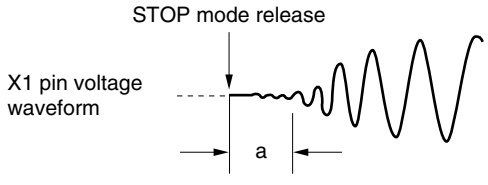
Cautions 1. After the above time has elapsed, the bits are set to 1 in order from MOST11 and remain 1.

2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTS. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.

- **Desired OSTC oscillation stabilization time ≤ Oscillation stabilization time set by OSTS**

Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC after STOP mode is released.

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



Remark fx: 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 OSTS.

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		
				fx = 12 MHz	fx = 16 MHz
0	0	1	2^{11}/fx	170.7 µs	128 µs
0	1	0	2^{13}/fx	682.7 µs	512 µs
0	1	1	2^{14}/fx	1.37 ms	1.024 ms
1	0	0	2^{15}/fx	2.73 ms	2.048 ms
1	0	1	2^{16}/fx	5.46 ms	4.096 ms
Other than above			Setting prohibited		

Cautions 1. To set the STOP mode when the X1 clock is used as the CPU clock, set OSTS before executing the STOP instruction.

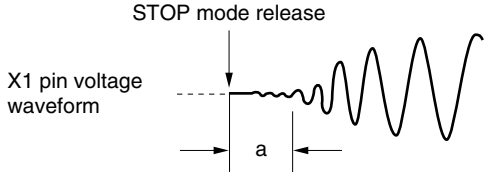
2. Do not change the value of the OSTS register during the X1 clock oscillation stabilization time.

3. The oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTS. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.

- **Desired OSTC oscillation stabilization time ≤ Oscillation stabilization time set by OSTS**

Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC after STOP mode is released.

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



Remark fx: X1 clock oscillation frequency

(8) PLL control register (PLLC)

This register sets the operation mode of PLL.

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

Reset signal generation sets this register to 01H.

Figure 5-9. Format of PLL Control Register (PLLC)

Address: FFA6H After reset: 01H R/W

Symbol	7	6	5	4	3	2	<1>	<0>
PLLC	0	0	0	0	0	0	PLLM	PLLSTOP

XSEL	PLLM	Selection of multiplication ratio for clock supplied to PLL/PLL	
		Supply clock	Multiplication ratio selection
0	0	Setting prohibited	Setting prohibited
	1	Setting prohibited	Setting prohibited
1	0	$f_{XH}/2$	x8[Note 1]
	1	$f_{XH}/4$	x12[Note 2]

PLLSTOP	PLL operation control
0	PLL oscillating
1	PLL stopped

Notes 1. f_{USB} = 48 MHz when f_{XH} = 12 MHz.

2. f_{USB} = 48 MHz when f_{XH} = 16 MHz.

Cautions. When using the USB function, set the clock supplied to PLL as initial setting after reset.

<Setting procedure>

<1> Stop PLL. (PLLSTOP=1)

<2> Select PLLM (0: f_{XH}=12 MHz 1: f_{XH}=16 MHz).

<3 > Set XSEL to 1.

<4> enable PLL driven. (PLLSTOP=0)

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

(9) USB clock control register (UCKC)

This register controls the USB clock (f_{USB}) supplied to the USB function controller.

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

Reset signal generation sets this register to 00H.

Figure 5-10. Format of USB Clock Control Register (UCKC)

Address: FFA7H After reset: 00H R/W

Symbol	<7>	6	5	4	3	2	1	0
UCKC	UCKCNT	0	0	0	0	0	0	0

UCKCNT	Control of USB clock supplied to USB function controller
0	Stops USB clock supply.
1	Supplies USB clock.

Caution Before shifting to the STOP mode, stop the clock supply to the USB function controller. After the STOP mode is released, count the PLL oscillation stabilization time (800 µs) using software, and supply the clock to the USB function controller after the oscillation stabilization wait time has elapsed.

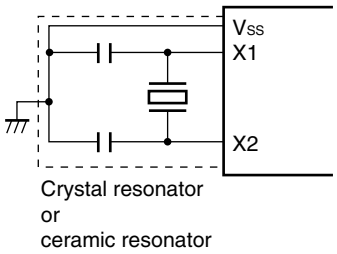
5.4 System Clock Oscillator

5.4.1 X1 oscillator

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

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

Figure 5-11. Example of External Circuit of X1 Oscillator (Crystal or Ceramic Oscillation)



Cautions 1. When using the X1 oscillator, wire as follows in the area enclosed by the broken lines in the Figure 5-11 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 Vss. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

Figure 5-12 shows examples of incorrect resonator connection.

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

(a) Too long wiring **(b) Crossed signal line**

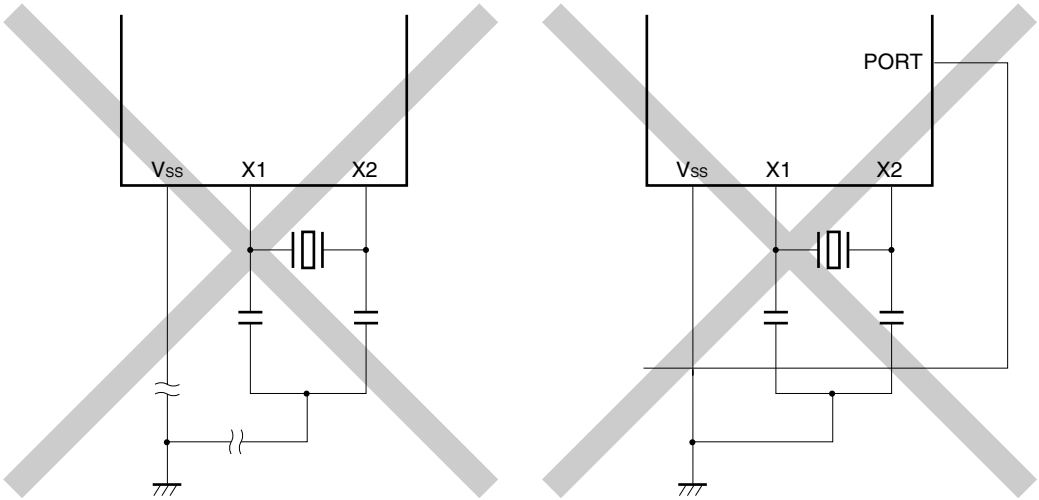
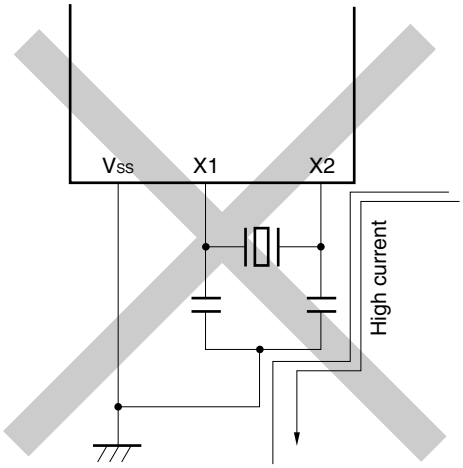
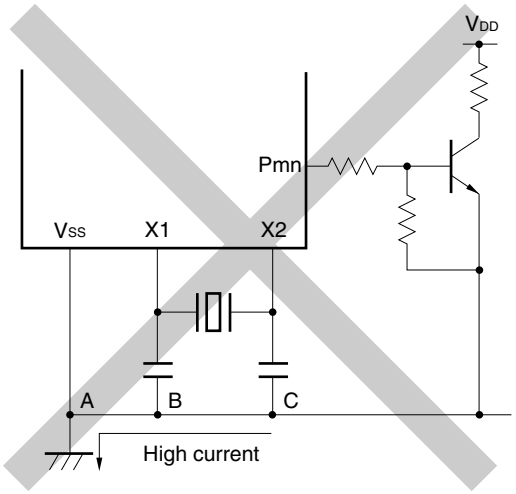


Figure 5-12. Examples of Incorrect Resonator Connection (2/2)

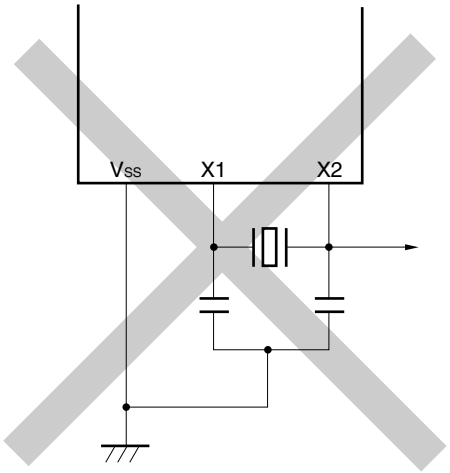
(c) Wiring near high alternating current



(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

An internal high-speed oscillator is incorporated in the µPD78F0730. Oscillation can be controlled by the internal oscillation mode register (RCM).

After a reset release, the internal high-speed oscillator automatically starts oscillation (16 MHz (TYP.)).

5.4.3 Internal low-speed oscillator

An internal low-speed oscillator is incorporated in the µPD78F0730.

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

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

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

5.4.4 Prescaler

The prescaler generates 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}
- USB clock f_{USB}
- Peripheral hardware clock f_{PRS}

The CPU starts operation when the internal high-speed oscillator starts outputting after a reset release in the µPD78F0730, 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.

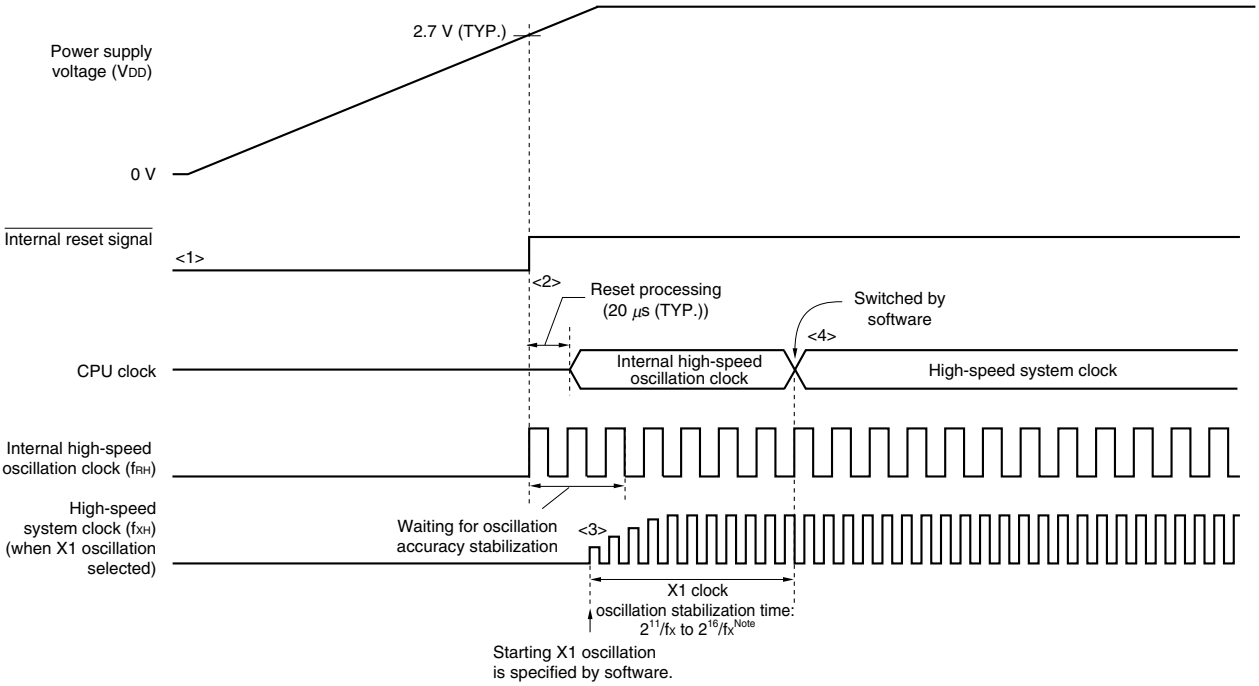
(2) Improvement of performance

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 5-13.

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

When 2.7 V/1.59 V POC Mode Is Set (Option Byte: POCMODE = 1)



<1> When the power is turned on, an internal reset signal is generated by the power-on-clear (POC) circuit.

<2> When the power supply voltage exceeds 2.7 V (TYP.), the reset is released and the internal high-speed oscillator automatically starts oscillation.

<3> After the reset is released and reset processing is performed, the CPU starts operation on the internal high-speed oscillation clock.

<4> Set the start of oscillation of the X1 clock via software (see **(1)** in **5.6.1 Example of controlling high-speed system clock)**.

<5> When switching the CPU clock to the X1 clock, wait for the clock oscillation to stabilize, and then set switching via software (see **(3)** in **5.6.1 Example of controlling high-speed system clock**).

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

Cautions 1. It is not necessary to wait for the oscillation stabilization time when an external clock input from the EXCLK pin is used.

2. For the µPD78F0730, be sure to set the 2.7 V/1.59 V POC mode by using the option byte (POCMODE = 1).

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 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 pin.
- 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 frequency (OSCCTL register)
Using AMPH, set the gain of the on-chip oscillator according to the frequency to be used.

AMPH[Note]	Operating Frequency Control
0	$f_{XH} \leq 10$ MHz
1	10 MHz < f_{XH}

Note Set AMPH before setting the peripheral functions after a reset release. The value of AMPH can be changed only once after a reset release. When AMPH is set to 1, the clock supply to the CPU is stopped for 5 μs (MIN.).

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

<2> Setting P121/X1 and P122/X2/EXCLK pins and selecting X1 clock or external clock (OSCCTL register)
When EXCLK is cleared to 0 and OSCSEL is set to 1, the mode is switched from port mode to X1 oscillation mode.

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

<3> Controlling oscillation of X1 clock (MOC register)
If MSTOP is cleared to 0, the X1 oscillator starts oscillating.

<4> 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 22 ELECTRICAL SPECIFICATIONS).

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

<1> Setting frequency (OSCCTL register)
Using AMPH, set the frequency to be used.

AMPH[Note]	Operating Frequency Control
0	$f_{XH} \leq 10$ MHz
1	10 MHz < f_{XH}

Note Set AMPH before setting the peripheral functions after a reset release. The value of AMPH can be changed only once after a reset release. When AMPH is set to 1, the clock supply to the CPU is stopped for 5 μs (MIN.).

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

<2> Setting P121/X1 and P122/X2/EXCLK pins and selecting operation mode (OSCCTL register)
When EXCLK and OSCSEL are set to 1, the mode is switched from port mode to external clock input mode.

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

<3> 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 22 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[Note]
(See **5.6.1 (1) Example of setting procedure when oscillating the X1 clock** and **(2) Example of setting procedure when using the external main system clock.**)

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

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

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

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

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

<3> 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.

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

(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 to set the STOP mode
- 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 14 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 CLS and MCS that the CPU is operating on a clock other than the high-speed system clock. When CLS = 0 and MCS = 1, the high-speed system clock is supplied to the CPU, so change the CPU clock to the internal high-speed oscillation clock.

CLS	MCS	CPU Clock Status
0	0	Internal high-speed oscillation clock
0	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 or CLS = 1 when setting MSTOP to 1. In addition, stop peripheral hardware that is operating on the high-speed system clock.

5.6.2 Example of controlling internal high-speed oscillation clock

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

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

(1) Example of setting procedure when restarting oscillation of the internal high-speed oscillation clock[Note 1]

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

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

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

Wait until RSTS is set to 1[Note 2].

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

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

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

<1> • Restarting oscillation of the internal high-speed oscillation clock[Note]
(See **5.6.2 (1) Example of setting procedure when restarting internal high-speed oscillation clock**).

• Oscillating the high-speed system clock[Note]
(This setting is required when using the high-speed system clock as the peripheral hardware clock. See **5.6.1 (1) Example of setting procedure when oscillating the X1 clock** and **(2) Example of setting procedure when using the external main system clock.**)

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

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

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

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

<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.

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

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

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

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

(a) To execute a STOP instruction

<1> Setting of peripheral hardware
Stop peripheral hardware that cannot be used in the STOP mode (for peripheral hardware that cannot be used in STOP mode, see **CHAPTER 14 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 CLS and MCS that the CPU is operating on a clock other than the internal high-speed oscillation clock.

When CLS = 0 and MCS = 0, the internal high-speed oscillation clock is supplied to the CPU, so change the CPU clock to the high-speed system clock.

CLS	MCS	CPU Clock Status
0	0	Internal high-speed oscillation clock
0	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 or CLS = 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}, $f_{RL}/2^7$, or $f_{RL}/2^9$ is selected as the count clock)

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

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

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

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

<1> Setting LSRSTOP to 1 (RCM register)

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

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

<1> Clearing LSRSTOP to 0 (RCM register)

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

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

5.6.4 Example of controlling USB clock

The clock (f_{USB} = 48 MHz) of the USB function controller (USBF) uses multiplication of division clock of high-speed system clock (f_{XH}) by PLL.

● Example of setting procedure when supplying the USB clock from the high-speed system clock (f_{XH} = 12/16 MHz)

<1> Setting PLLSTOP to 1 (PLLC register)
When PLLSTOP is set to 1, the PLL stops operation.

<2> Setting PLLM to 0/1 (PLLC register)
In the case of f_{XH} = 12 MHz, PLLM is set to 0 in order to select “8 times”.
In the case of f_{XH} = 16 MHz, PLLM is set to 1 in order to select “12 times”.

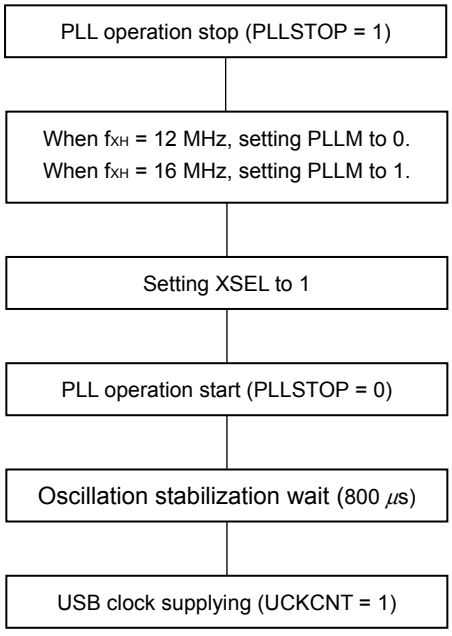
<3> Setting XSEL to 1 (MCM register)
When XSEL is set to 1, the high-speed system clock is supplied to the PLL.

<4> Clearing PLLSTOP to 0 (PLLC register)
When PLLSTOP is cleared to 0, the PLL starts operating.

<5> Waiting for oscillation stabilization of the PLL
Wait for 800 µs by software. Other software processing can be executed while waiting.

<6> Setting UCKCNT to 1 (UCKC register)
When UCKCNT is set to 1, the USB clock is supplied to the USB function controller.

[Control flow]



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

Remarks 1. XSEL: Bit 2 of the main clock mode register (MCM)

2. CSS: Bit 4 of the processor clock control register (PCC)

3. MCM0: Bit 0 of MCM

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

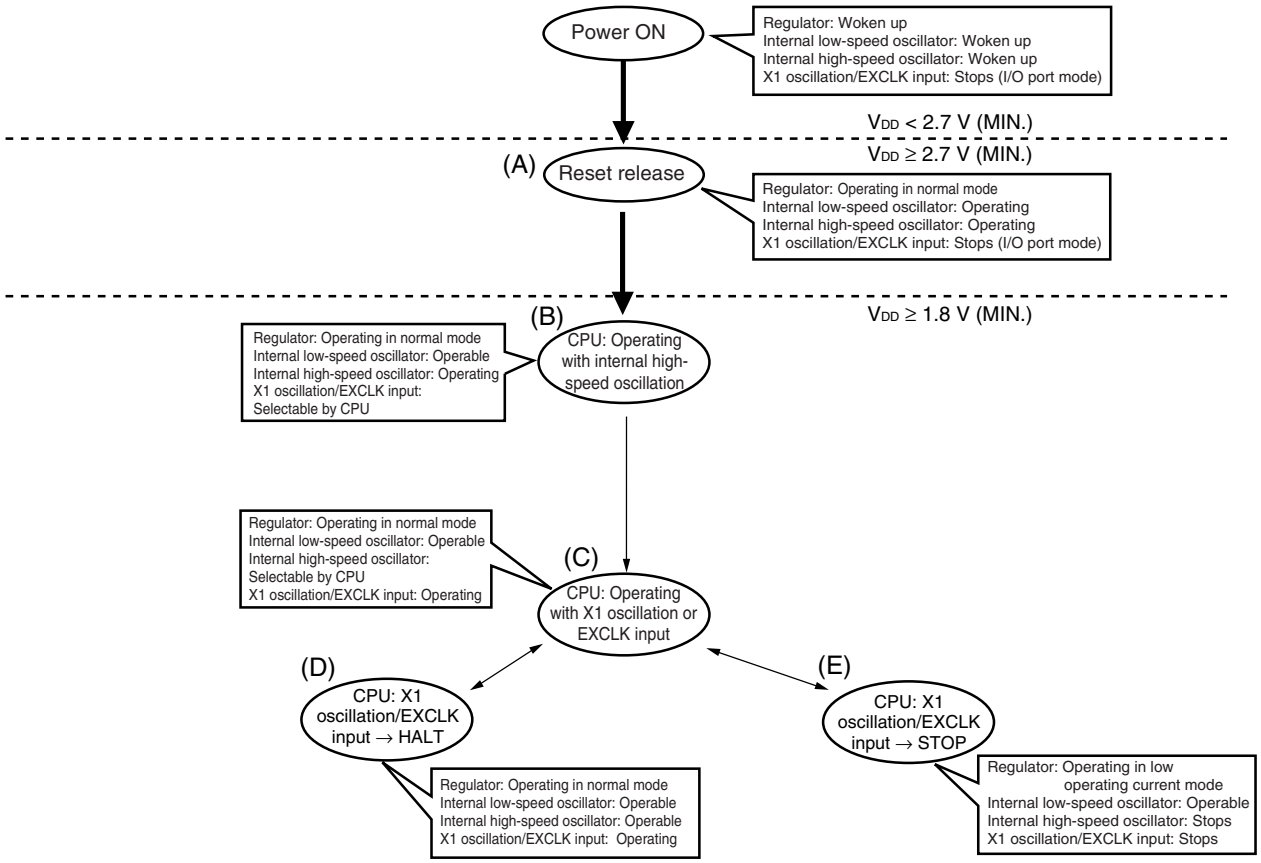
5. ×: don't care

5.6.6 CPU clock status transition diagram

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

Figure 5-14. CPU Clock Status Transition Diagram

When 2.7 V/1.59 V POC Mode Is Set (Option Byte: POCMODE = 1)



Remark In the 2.7 V/1.59 V POC mode (option byte: POCMODE = 1), the CPU clock status changes to (A) in the above figure when the supply voltage exceeds 2.7 V (TYP.), and to (B) after reset processing (20 μs (TYP.)).

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 (1/3)

(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	AMPH	EXCLK	OSCSEL	MSTOP	OSTC Register	XSEL	MCM0
(A) → (B) → (C) (X1 clock: f_{XH} ≤ 10 MHz)	0	0	1	0	Must be checked	1	1
(A) → (B) → (C) (external main clock: f_{XH} ≤ 10 MHz)	0	1	1	0	Must not be checked	1	1
(A) → (B) → (C) (X1 clock: 10 MHz < f_{XH})	1	0	1	0	Must be checked	1	1
(A) → (B) → (C) (external main clock: 10 MHz < f_{XH})	1	1	1	0	Must not be checked	1	1

Caution Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 22 ELECTRICAL SPECIFICATIONS).

Remarks 1. (A) to (E) in Table 5-4 correspond to (A) to (E) in Figure 5-14.

2. EXCLK, OSCSEL, AMPH: Bits 7, 6 and 0 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)

Table 5-4. CPU Clock Transition and SFR Register Setting Examples (2/3)

(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	AMPH[Note]	EXCLK	OSCSEL	MSTOP	OSTC Register	XSEL[Note]	MCM0
(B) → (C) (X1 clock: f_{XH} ≤ 10 MHz)	0	0	1	0	Must be checked	1	1
(B) → (C) (external main clock: f_{XH} ≤ 10 MHz)	0	1	1	0	Must not be checked	1	1
(B) → (C) (X1 clock: 10 MHz < f_{XH})	1	0	1	0	Must be checked	1	1
(B) → (C) (external main clock: 10 MHz < f_{XH})	1	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.

Cautions 1. Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 22 ELECTRICAL SPECIFICATIONS).

2. CPU clock cannot changes from high-speed system clock (C) to internal high-speed oscillation clock (B)

Remarks 1. (A) to (E) in Table 5-4 correspond to (A) to (E) in Figure 5-14.

2. EXCLK, OSCSEL, AMPH: Bits 7, 6 and 0 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)
RSTS, RSTOP: Bits 7 and 0 of the internal oscillation mode register (RCM)

Table 5-4. CPU Clock Transition and SFR Register Setting Examples (3/3)

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

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

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

(Setting sequence) ——————————▶

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

Remark (A) to (E) in Table 5-4 correspond to (A) to (E) in Figure 5-14.

5.6.7 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). • Clock supply to CPU is stopped for 5 µs (MIN.) after AMPH has been set to 1.
	External main system clock	Enabling input of external clock from EXCLK pin • MSTOP = 0, OSCSEL = 1, EXCLK = 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).

5.6.8 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 (the internal high-speed oscillation clock to 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
	1
0	$1 + 2f_{RH}/f_{XH}$ 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} = 16 MHz, f_{XH} = 12 MHz)

$1 + 2f_{RH}/f_{XH} = 1 + 2 \times 16/12 = 1 + 2 \times 1.33 = 1 + 2.66 = 3.66 \rightarrow 3$ clocks

5.6.9 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 External main system clock	MCS = 1 (The CPU is operating on a clock other than the high-speed system clock)	MSTOP = 1

5.6.10 Peripheral hardware and source clocks

The following lists peripheral hardware and source clocks incorporated in the μPD78F0730.

Table 5-9. Peripheral Hardware and Source Clocks

Peripheral Hardware \ Source Clock		Peripheral Hardware Clock (f_{PRS})	Internal Low-Speed Oscillation Clock (f_{RL})	TM50 Output	External Clock from Peripheral Hardware Pins
16-bit timer/event counter 00		Y	N	N	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 H1		Y	Y	N	N
Watchdog timer		N	Y	N	N
Serial interface	UART6	Y	N	Y	N
	CSI10	Y	N	N	Y ($\overline{SCK10}$ pin)

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

CHAPTER 6 16-BIT TIMER/EVENT COUNTER 00

6.1 Functions of 16-Bit Timer/Event Counter 00

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

(1) Interval timer

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

(2) Square-wave output

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

(3) External event counter

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

(4) One-shot pulse output

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

(5) PPG output

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

(6) Pulse width measurement

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

6.2 Configuration of 16-Bit Timer/Event Counter 00

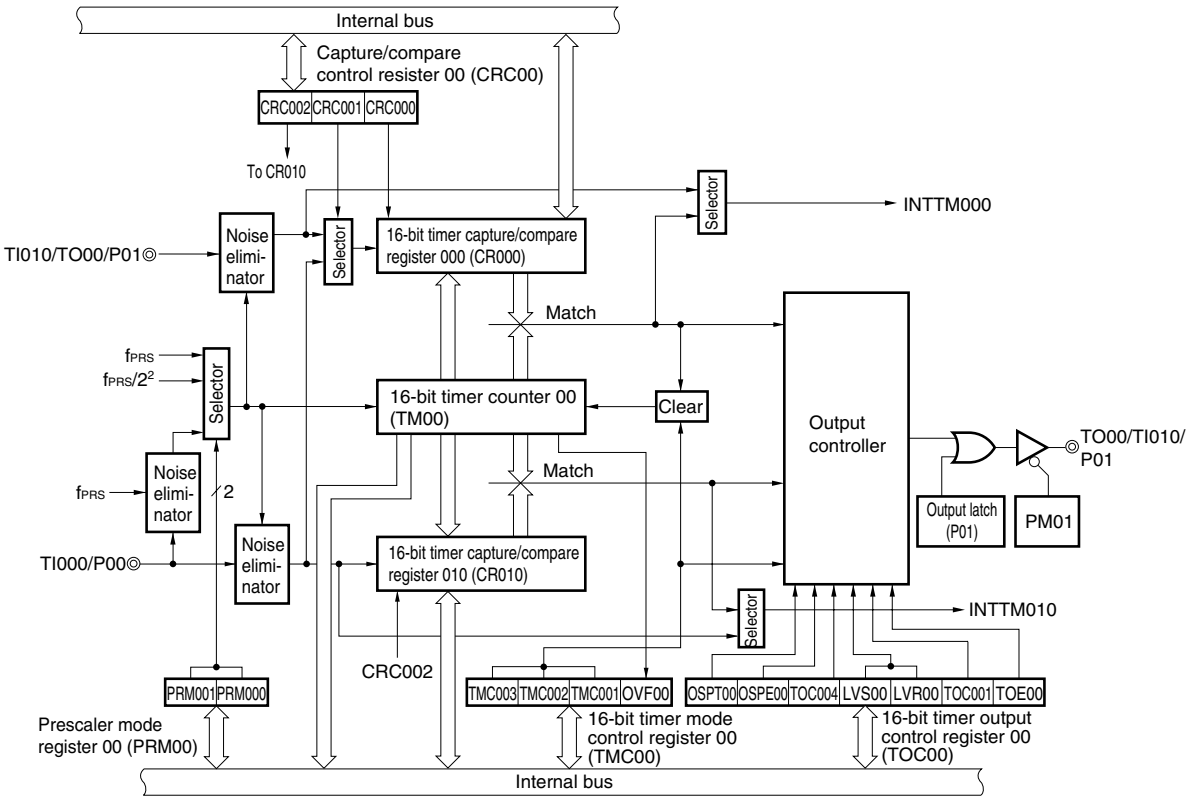
16-bit timer/event counter 00 includes the following hardware.

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

Item	Configuration
Time/counter	16-bit timer counter 00 (TM00)
Register	16-bit timer capture/compare registers 000, 010 (CR000, CR010)
Timer input	TI000, TI010 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)

Figure 6-1 shows the block diagram.

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



Remark f_{PRS}: Peripheral hardware clock frequency

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

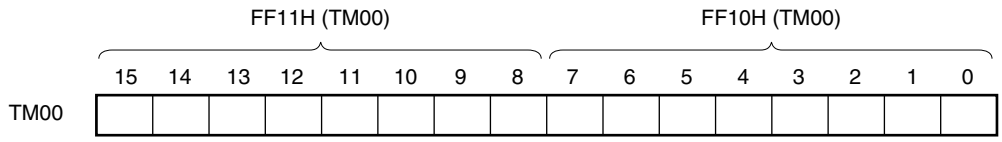
TM00 is a 16-bit read-only register that counts count pulses.

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)

Address: FF10H, FF11H (TM00) After reset: 0000H R



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 or the valid edge is input to the TI000 pin in one-shot pulse output mode

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

2. When TM00 is read, input of the count clock is temporarily stopped and it is resumed after the timer has been read. Therefore, no clock miss occurs.

(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).

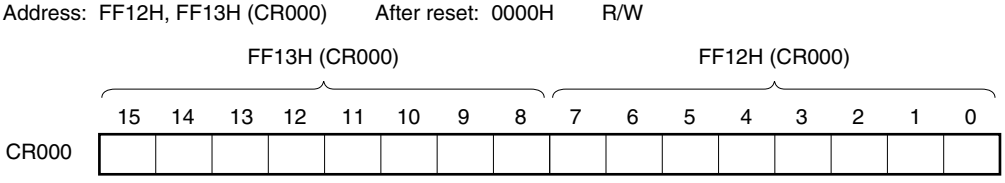
The value of CR010 can be changed during operation if the value has been set in a specific way. For details, see **6.5.1 Rewriting CR010 during TM00 operation**.

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)

Address: FF12H, FF13H (CR000) After reset: 0000H R/W



(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.

(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)

Address: FF14H, FF15H (CR010) After reset: 0000H R/W

	FF15H (CR010)								FF14H (CR010)							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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.

(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.

Cautions 1. To use this register as a compare register, set a value other than 0000H to CR000 and CR010.

2. The valid edge of TI010 and timer output (TO00) cannot be used for the P01 pin at the same time. Select either of the functions.

3. If clearing of its 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.

4. 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.

5. CR000/CR010 does not perform the capture operation when it is set in the comparison mode, even if a capture trigger is input to it.

Table 6-2. Capture Operation of CR000 and CR010

External Input Signal / Capture Operation	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[Note]	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.		

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)

(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 TM00 operation. Stops supplying operating clock. Asynchronously resets the internal circuit.
0	1	Free-running timer mode
1	0	Clear & start mode entered by TI000 pin valid edge input[Note]
1	1	Clear & start mode entered upon a match between TM00 and CR000

TMC001	Condition to reverse timer output (TO00)
0	• Match between TM00 and CR000 or match between TM00 and CR010
1	• Match between TM00 and CR000 or match between TM00 and CR010 • Trigger input of TI000 pin valid edge

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

Note The TI000 pin valid edge is set by bits 5 and 4 (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: FFBCH After reset: 00H R/W

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

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

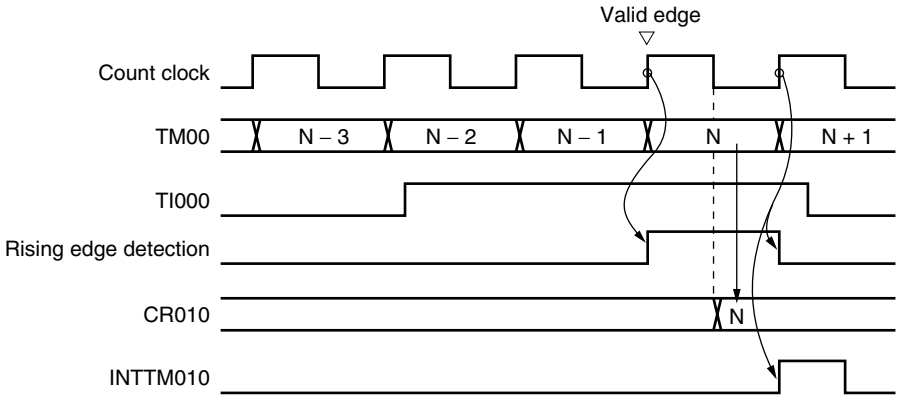
CRC001	CR000 capture trigger selection
0	Captures on valid edge of TI010 pin
1	Captures on valid edge of TI000 pin by reverse phase[Note]
The valid edge of the TI010 and TI000 pin is set by PRM00. If ES001 and ES000 are set to 11 (both edges) when CRC001 is 1, the valid edge of the TI000 pin cannot be 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 the TO00 pin 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 pin 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 pin output status
0	0	No change
0	1	Initial value of TO00 pin output is low level (TO00 pin output is cleared to 0).
1	0	Initial value of TO00 pin output is high level (TO00 pin output is set to 1).
1	1	Setting prohibited

- LVS00 and LVR00 can be used to set the initial value of the output level of the TO00 pin. If the initial value does not have to be set, leave LVS00 and LVR00 as 00.
- Be sure to set LVS00 and LVR00 when TOE00 = 1.
  LVS00, LVR00, and TOE00 being simultaneously set to 1 is prohibited.
- LVS00 and LVR00 are trigger bits. By setting these bits to 1, the initial value of the output level of the TO00 pin can be set. Even if these bits are cleared to 0, output of the TO00 pin is not affected.
- The values of LVS00 and LVR00 are always 0 when they are read.
- For how to set LVS00 and LVR00, see **6.5.2 Setting LVS00 and LVR00**.

TOC001	TO00 pin 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 pin output control
0	Disables output (TO00 pin 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 sets 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 P01 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} = 12 MHz	f_{PRS} = 16 MHz
0	0	f_{PRS}	12 MHz	16 MHz
0	1	$f_{PRS}/2^2$	3 MHz	4 MHz
1	0	Setting prohibited		
1	1	TI000 valid edge[Note]		

Note The external clock requires a pulse two cycles longer than internal 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 P01/TO00/TI010 pin for timer output, set PM01 and the output latches of P01 6 to 0.

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

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

Reset signal generation sets PM0 to FFH.

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

Address: FF20H After reset: FFH R/W

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

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

6.4 Operation of 16-Bit Timer/Event Counter 00

6.4.1 Interval timer operation

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

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

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

2. For how to enable the INTTM000 interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

Figure 6-11. Block Diagram of Interval Timer Operation

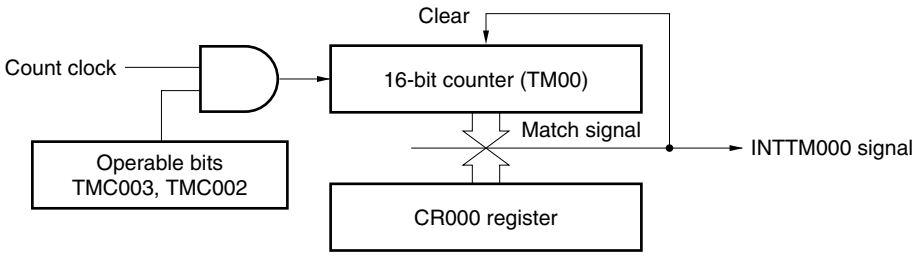


Figure 6-12. Basic Timing Example of Interval Timer Operation

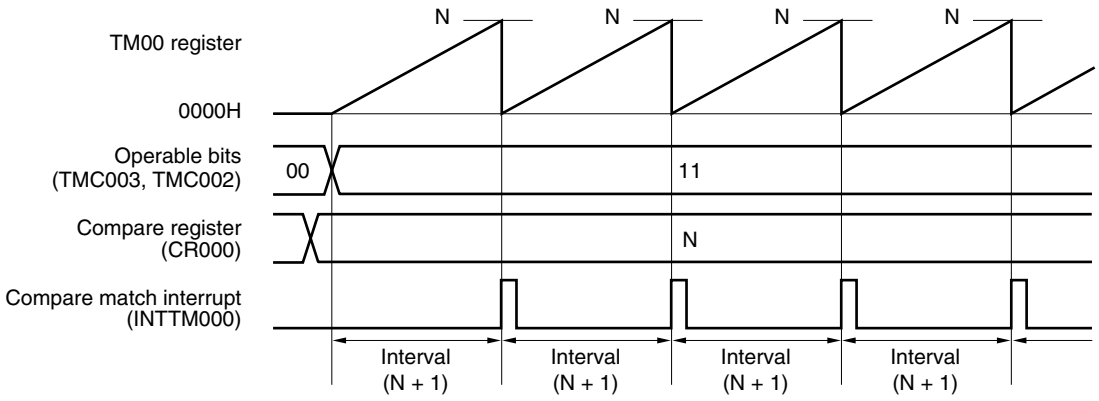


Figure 6-13. Example of Register Settings for Interval Timer Operation

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

				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	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	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) × Count clock cycle

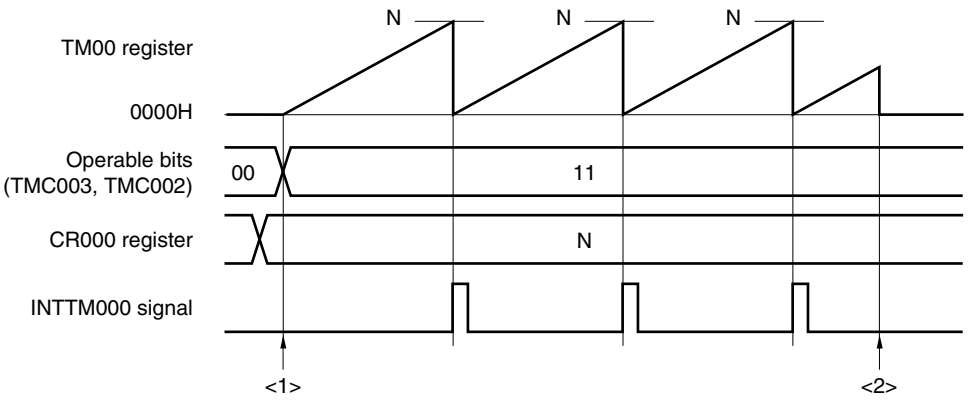
Setting CR000 to 0000H is prohibited.

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

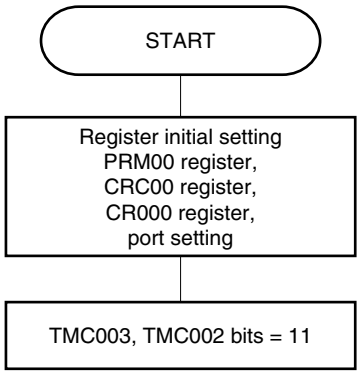
Usually, CR010 is not used for the interval timer function. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

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

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



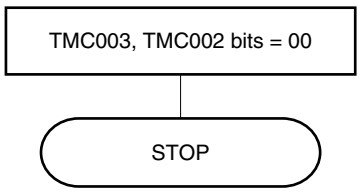
<1> Count operation start flow



Initial setting of these registers is performed before setting the TMC003 and TMC002 bits to 11.

Starts count operation

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TMC003 and TMC002 bits to 00.

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 output of the TO00 pin is inverted. This TO00 pin 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)**.

2. For how to enable the INTTM000 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

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

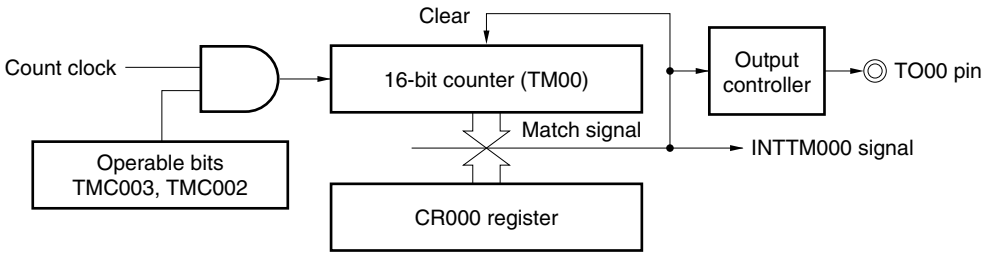


Figure 6-16. Basic Timing Example of Square Wave Output Operation

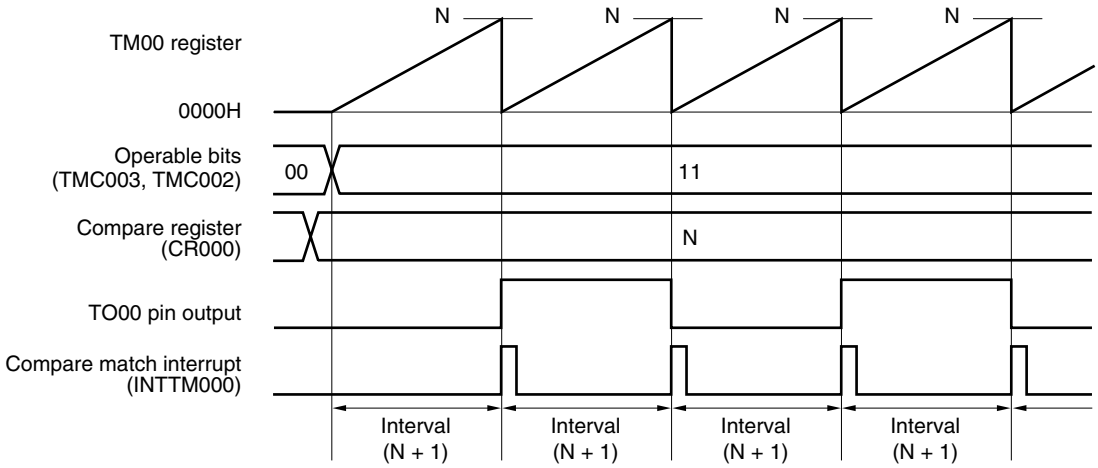
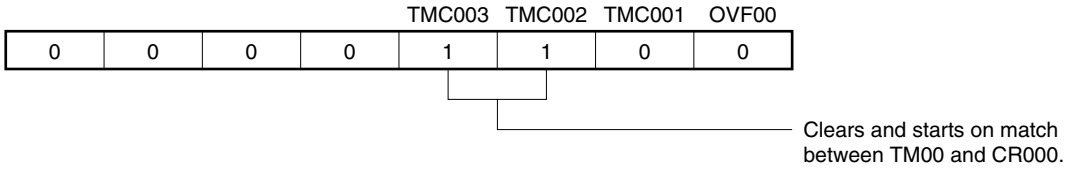
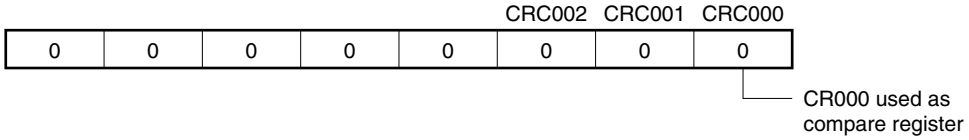


Figure 6-17. Example of Register Settings for Square Wave Output Operation

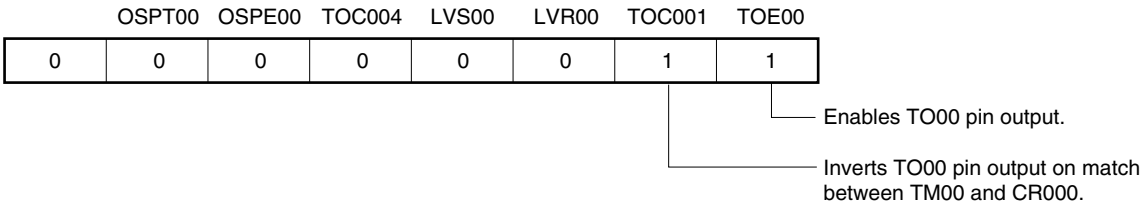
(a) 16-bit timer mode control register 00 (TMC00)



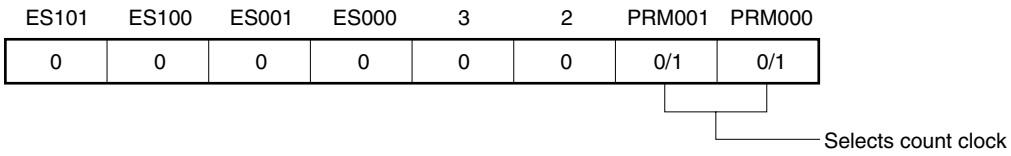
(b) Capture/compare control register 00 (CRC00)



(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 × (M + 1) × Count clock cycle]

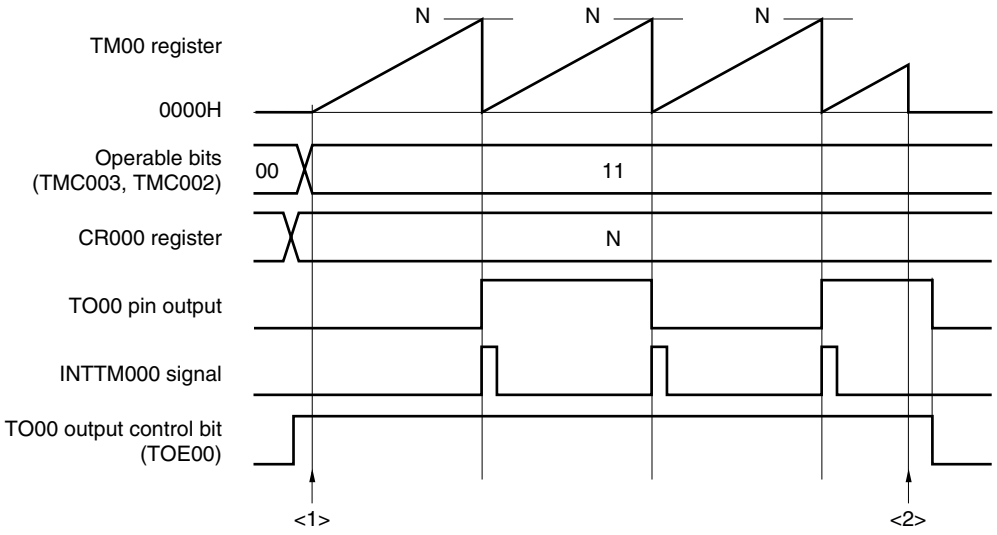
Setting CR000 to 0000H is prohibited.

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

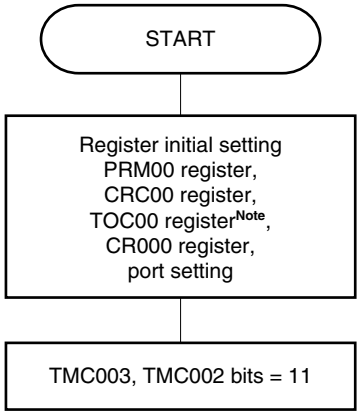
Usually, CR010 is not used for the square wave output function. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

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

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



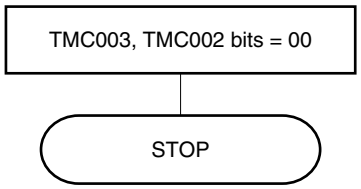
<1> Count operation start flow



Initial setting of these registers is performed before setting the TMC003 and TMC002 bits to 11.

Starts count operation

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TMC003 and TMC002 bits to 00.

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 × (Set value of CR000 + 1)

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

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

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

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

2. For how to enable the INTTM000 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

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

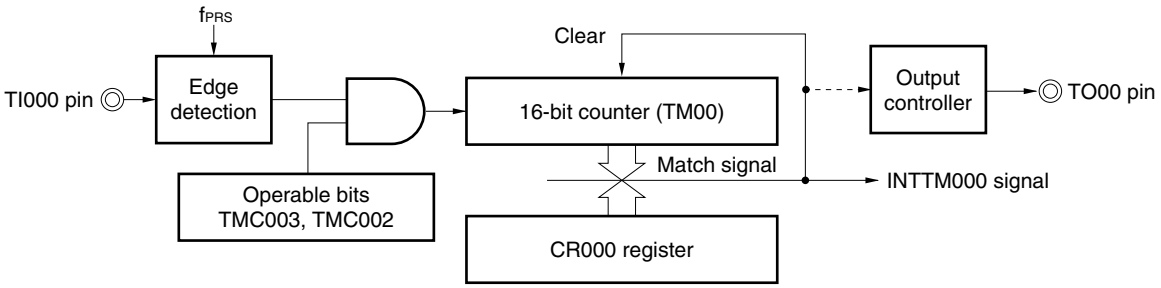
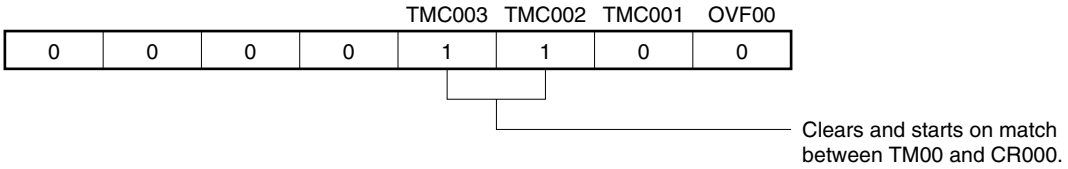
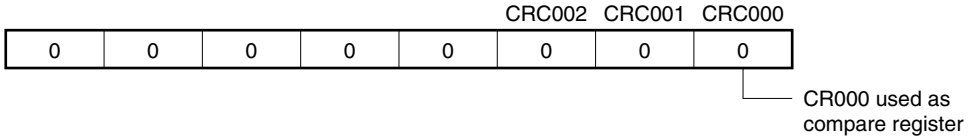


Figure 6-20. Example of Register Settings in External Event Counter Mode

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



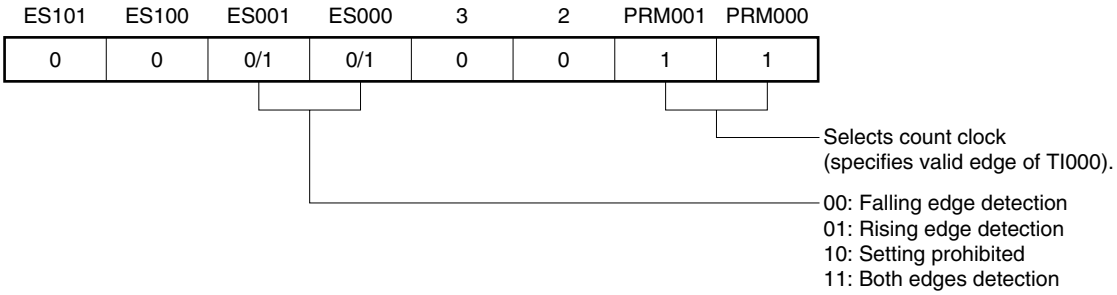
(b) Capture/compare control register 00 (CRC00)



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

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

(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).

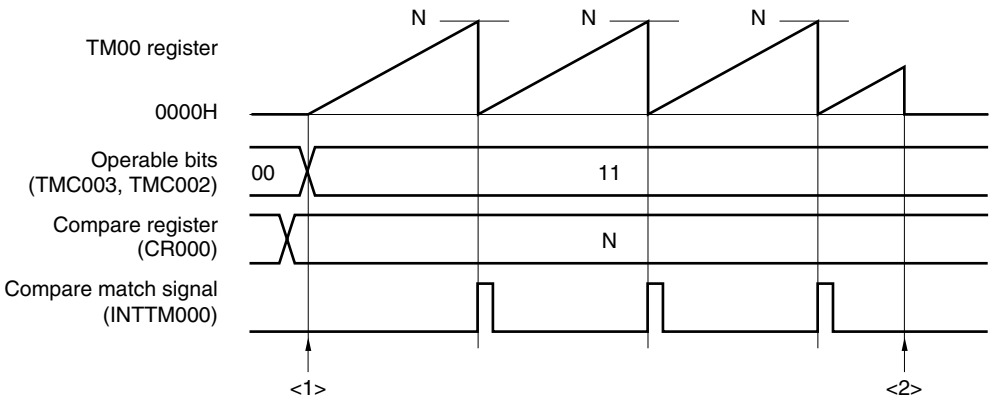
Setting CR000 to 0000H is prohibited.

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

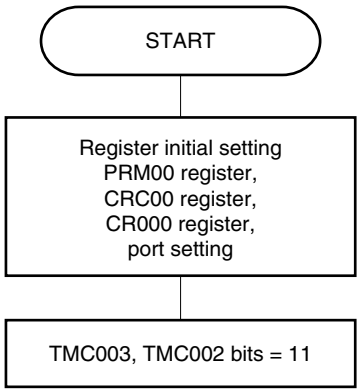
Usually, CR010 is not used in the external event counter mode. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

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

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



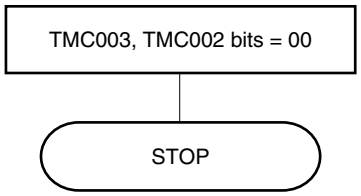
<1> Count operation start flow



Initial setting of these registers is performed before setting the TMC003 and TMC002 bits to 11.

Starts count operation

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TMC003 and TMC002 bits to 00.

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 13 INTERRUPT FUNCTIONS**.

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

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

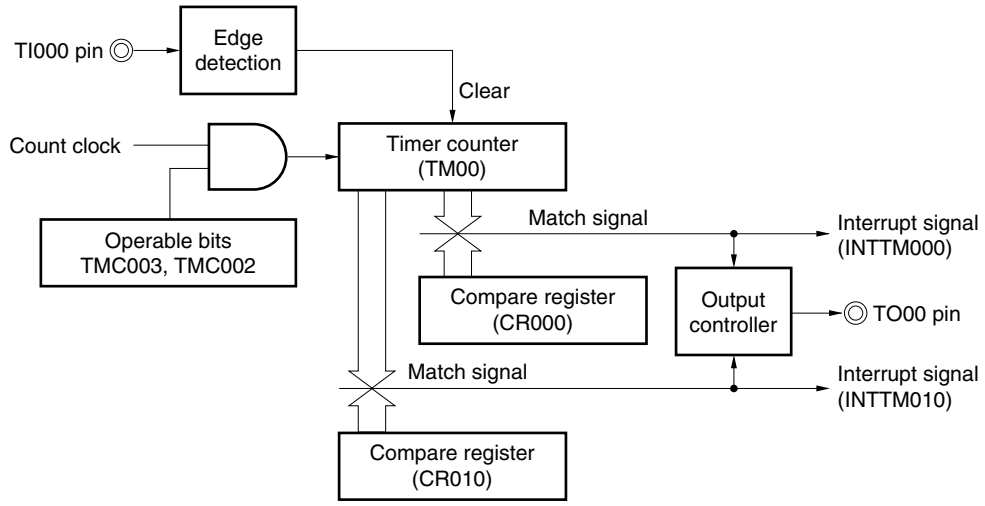
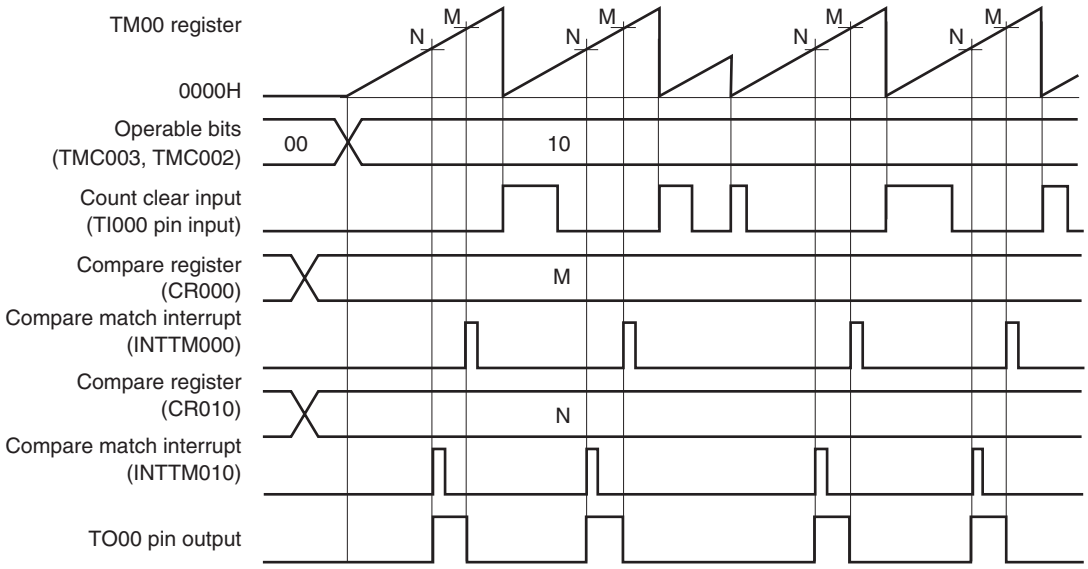
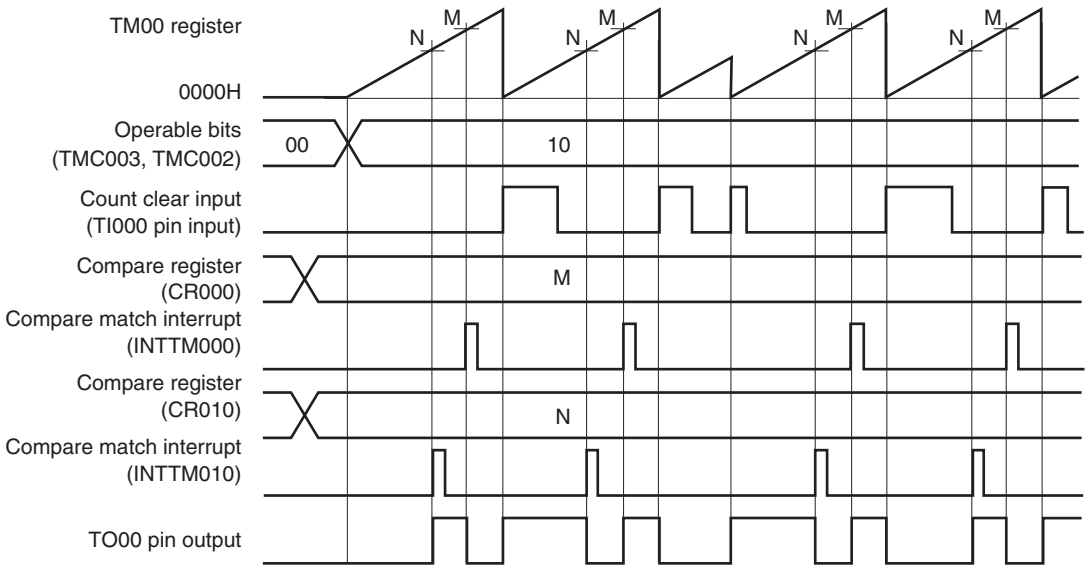


Figure 6-23. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input (CR000: Compare Register, CR010: Compare Register)

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

(a) The output level of the TO00 pin is inverted when TM00 matches a compare register.

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

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

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

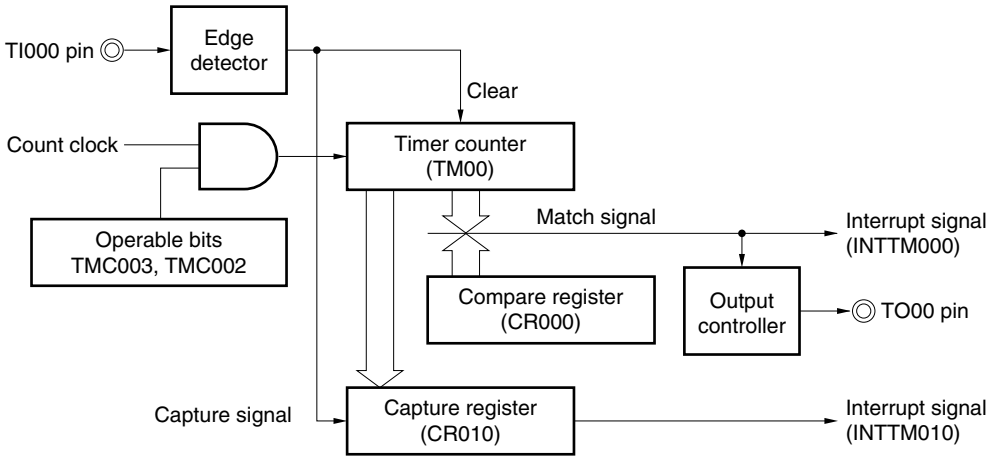
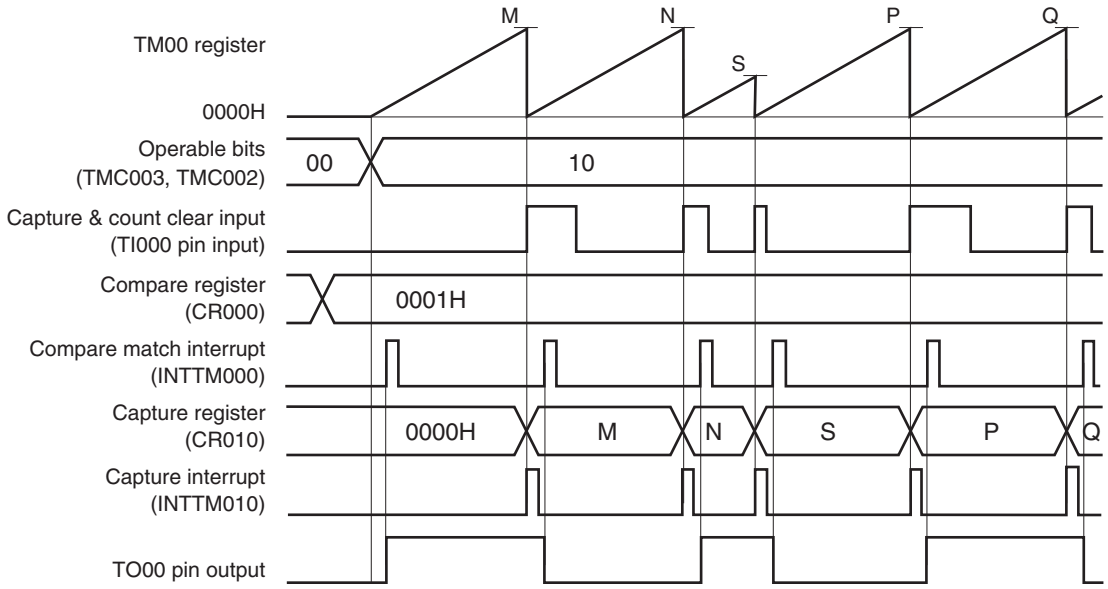


Figure 6-25. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input (CR000: Compare Register, CR010: Capture Register) (1/2)

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

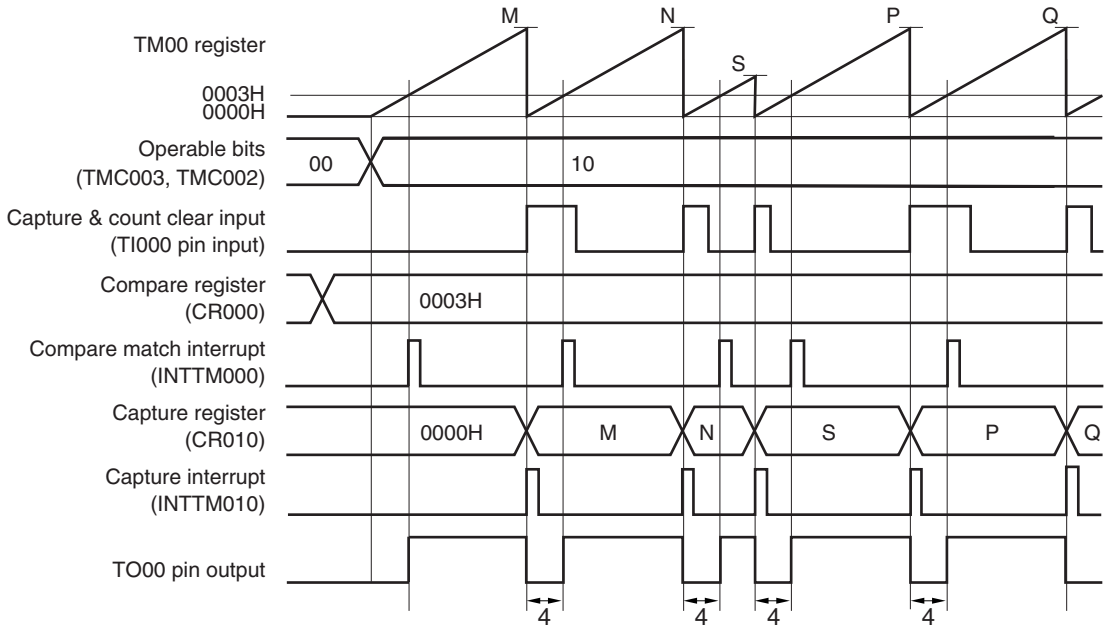


This is an application example where the output level of the TO00 pin is inverted when the count value has been captured & cleared.

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 output level of the TO00 pin is inverted.

Figure 6-25. 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 output level of the TO00 pin 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 output level of the TO00 pin is inverted.

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

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

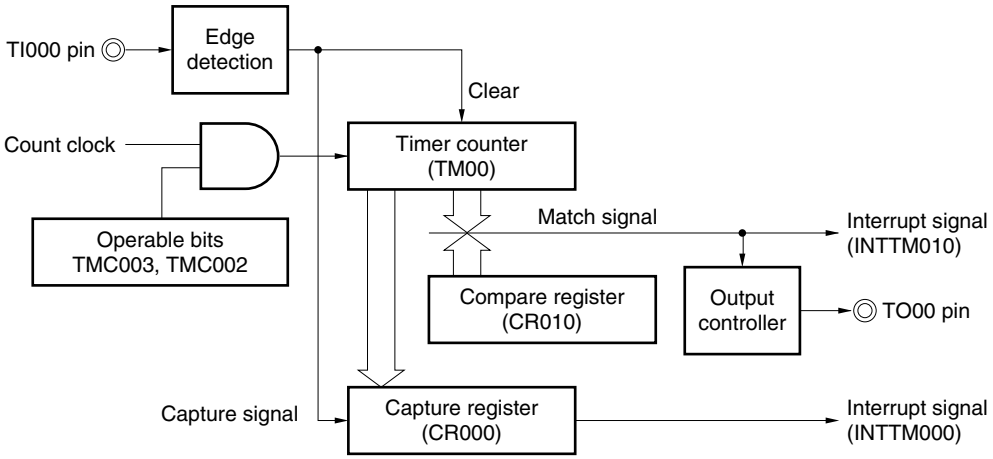
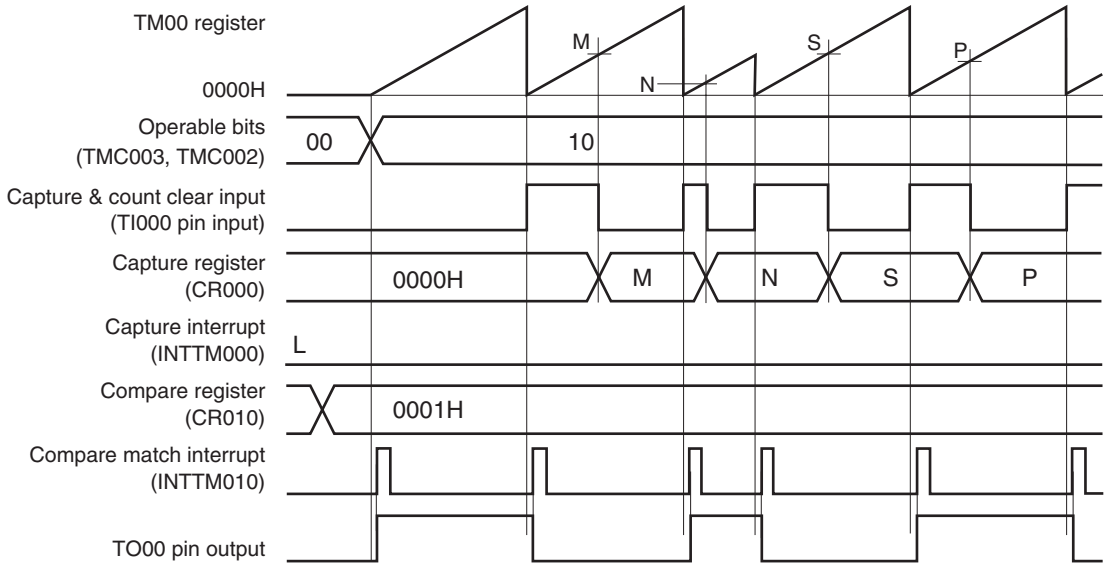


Figure 6-27. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input (CR000: Capture Register, CR010: Compare Register) (1/2)

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



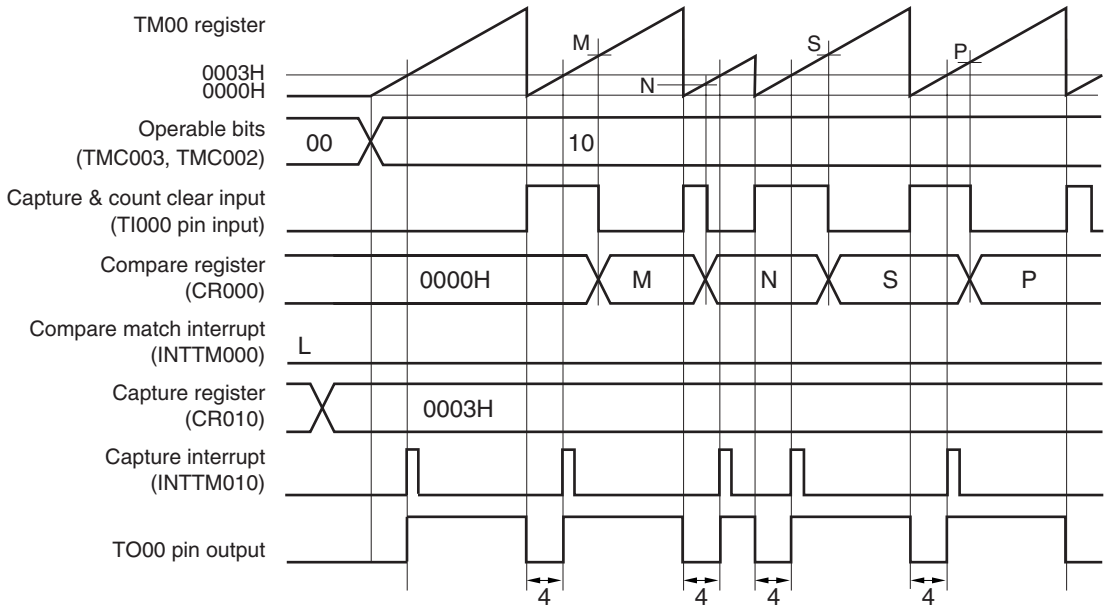
This is an application example where the output level of the TO00 pin is to be inverted when the count value has been captured & cleared.

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

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

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

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



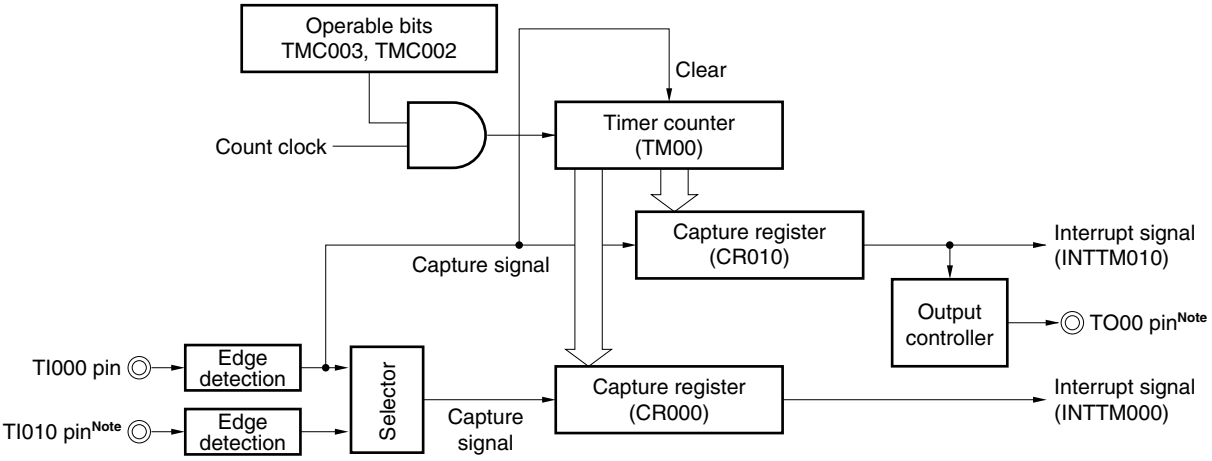
This is an application example where the width set to CR010 (4 clocks in this example) is to be output from the TO00 pin when the count value has been captured & cleared.

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

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

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

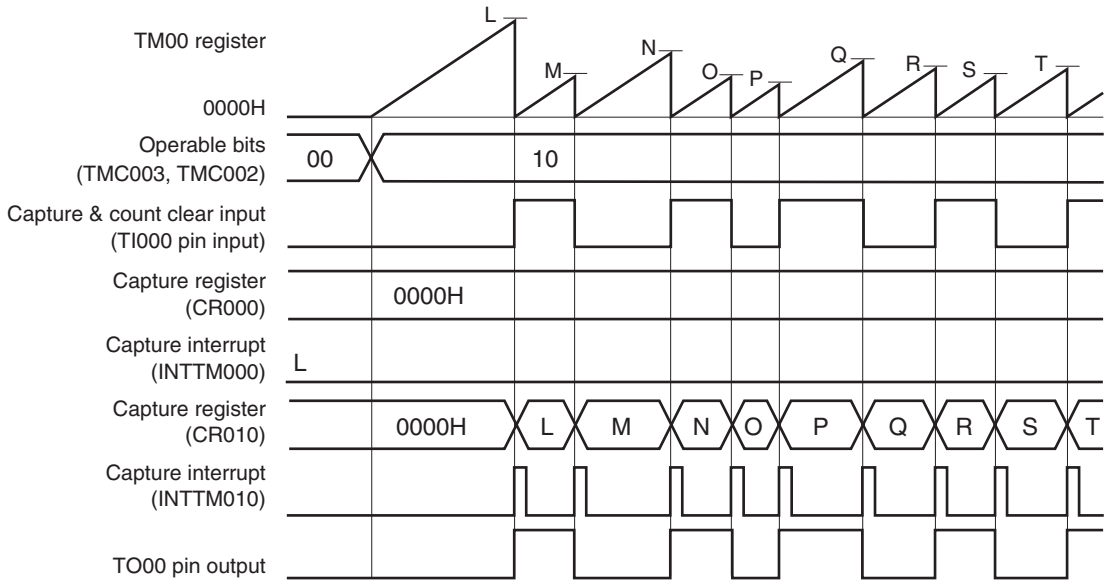
Figure 6-28. Block Diagram of Clear & Start Mode Entered by TI000 Pin Valid Edge Input (CR000: Capture Register, CR010: Capture Register)



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

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

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

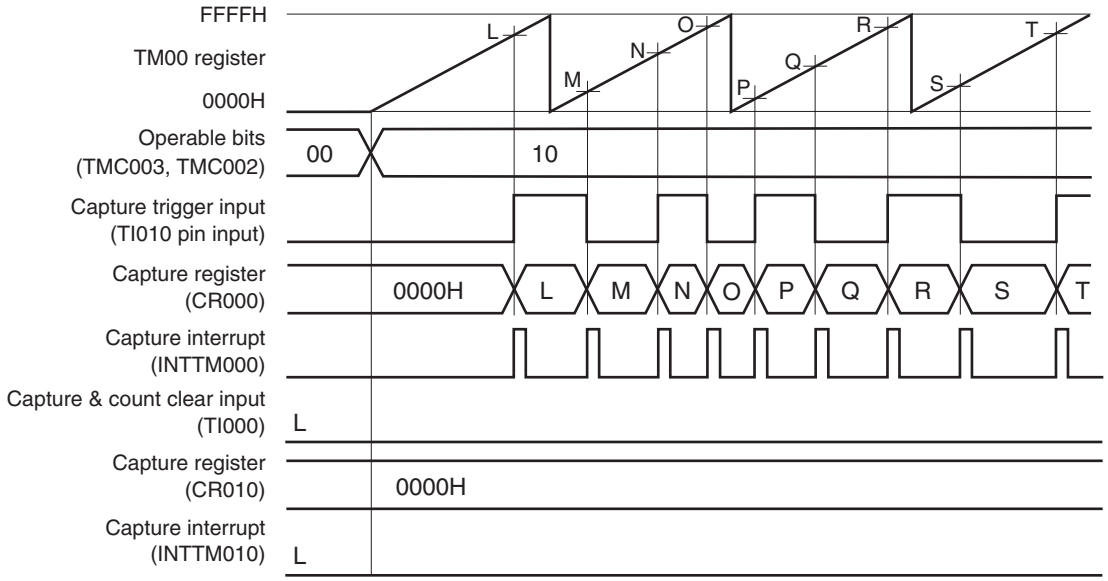


This is an application example where the count value is captured to CR010, TM00 is cleared, and the TO00 pin output is inverted when the rising or falling edge of the TI000 pin is detected.

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

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

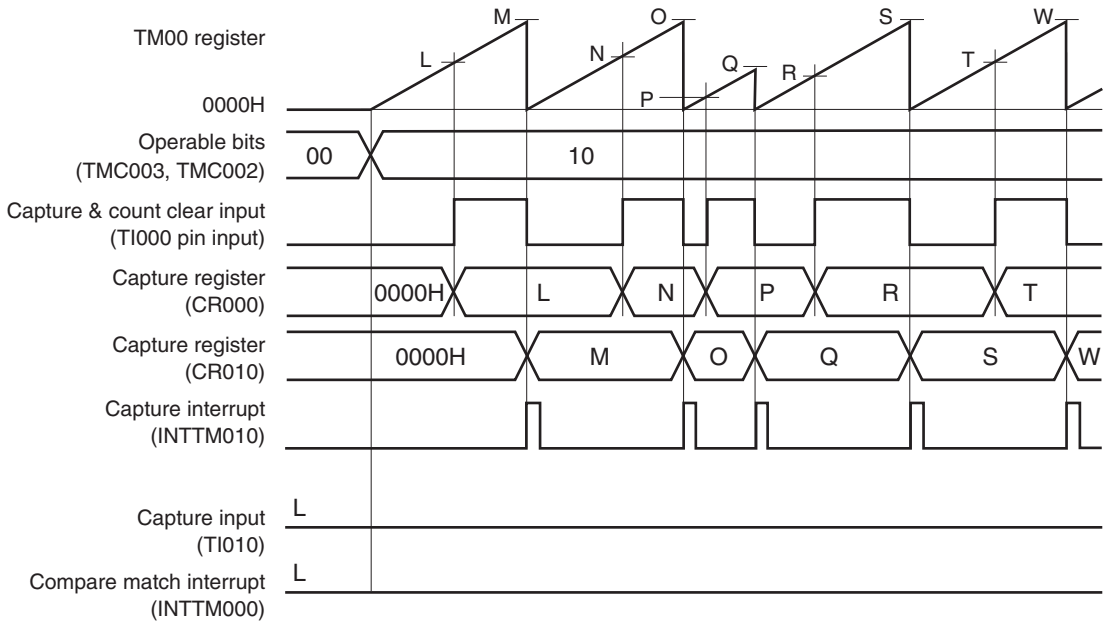
(b) TOC00 = 13H, PRM00 = C0H, CRC00 = 05H, TMC00 = 0AH



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

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

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



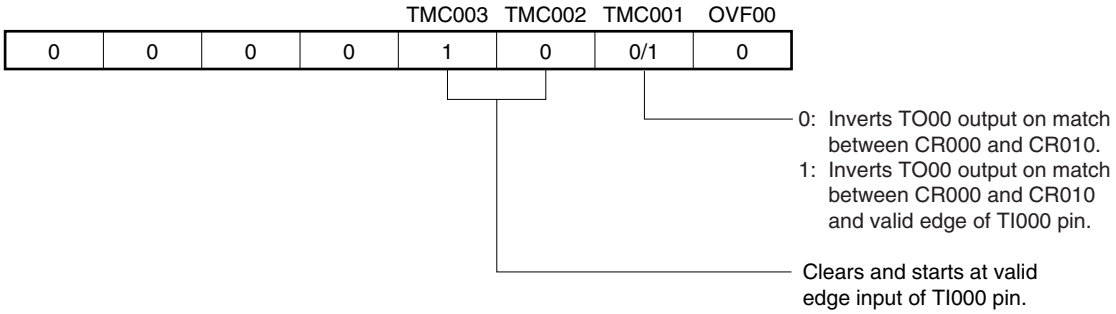
This is an application example where the pulse width of the signal input to the TI000 pin is measured.
By setting CRC00, the count value can be captured to CR000 in the phase reverse to the falling edge of the TI000 pin (i.e., rising edge) and to CR010 at the falling edge of the TI000 pin.
The high- and low-level widths of the input pulse can be calculated by the following expressions.

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

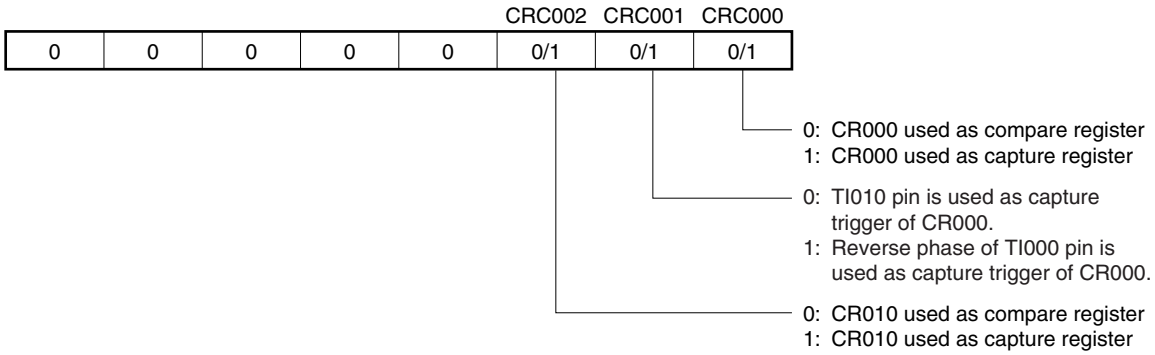
If the reverse phase of the TI000 pin is selected as a trigger to capture the count value to CR000, the INTTM000 signal is not generated. Read the values of CR000 and CR010 to measure the pulse width immediately after the INTTM010 signal is generated.
However, if the valid edge specified by bits 6 and 5 (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 6-30. Example of Register Settings in Clear & Start Mode Entered by TI000 Pin Valid Edge Input (1/2)

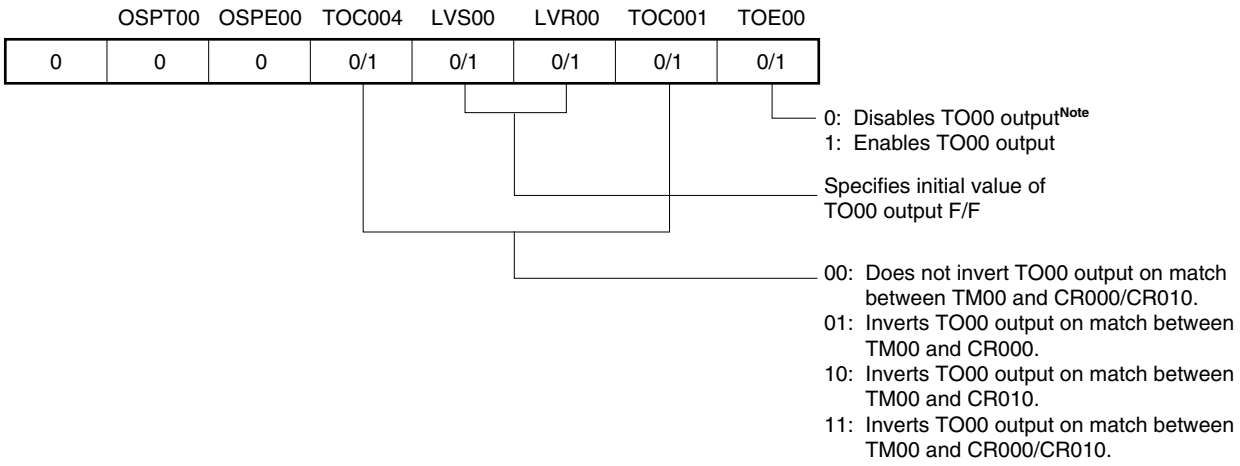
(a) 16-bit timer mode control register 00 (TMC00)



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



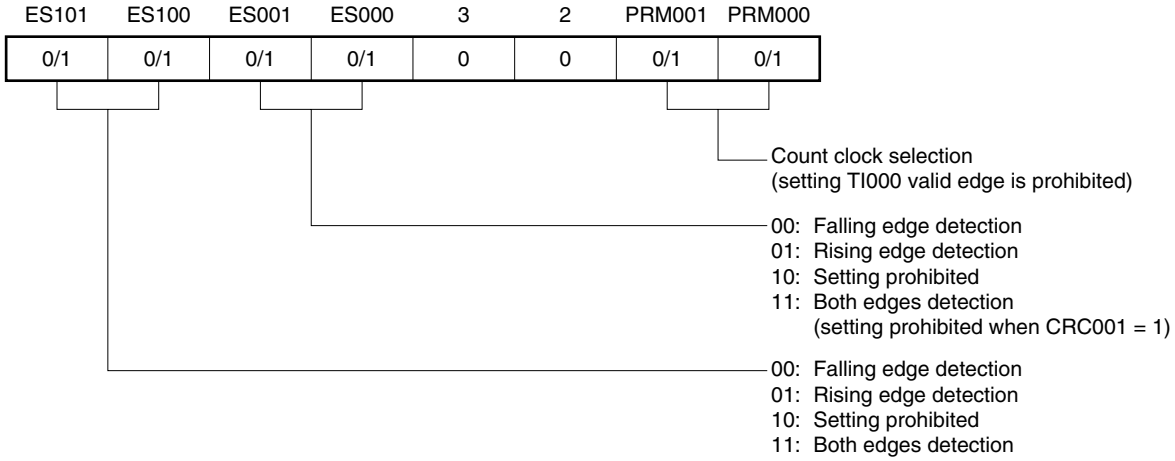
(c) 16-bit timer output control register 00 (TOC00)



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

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

(d) Prescaler mode register 00 (PRM00)



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

By reading TM00, the count value can be read.

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

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

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

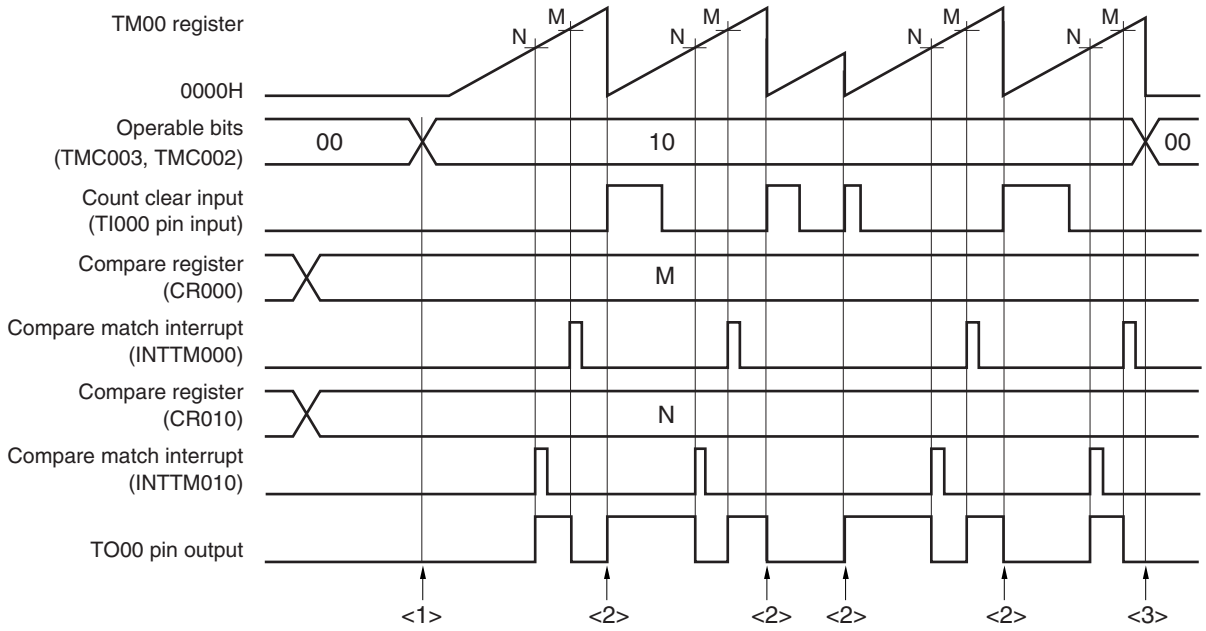
Note The timer output (TO00) cannot be used when detection of the valid edge of the TI010 pin is used.

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

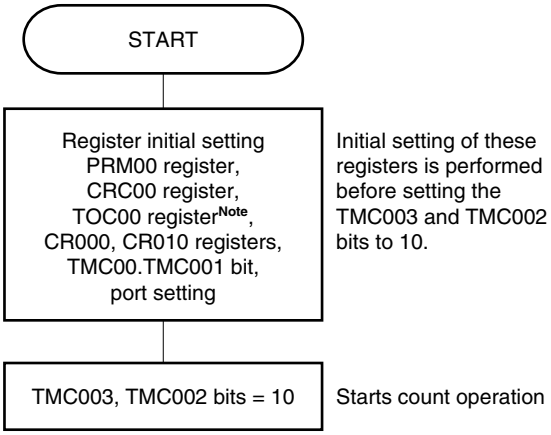
When this register is used as a compare register and when its value matches the count value of TM00, an interrupt signal (INTTM010) is generated. The count value of TM00 is not cleared.

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

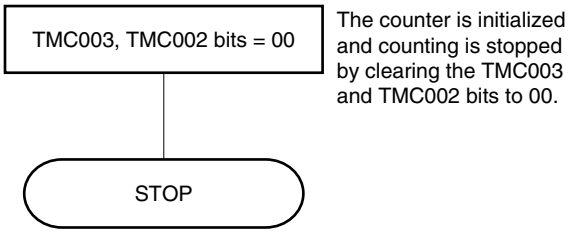
Figure 6-31. Example of Software Processing in Clear & Start Mode Entered by TI000 Pin Valid Edge Input



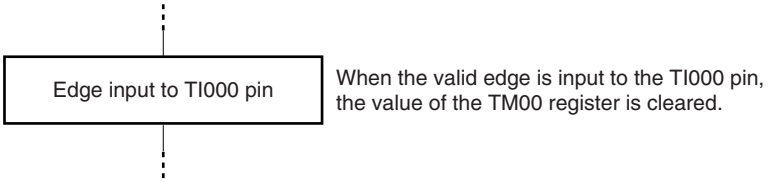
<1> Count operation start flow



<3> Count operation stop flow



<2> TM00 register clear & start flow



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

2. For how to enable the INTTM000 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

(1) Free-running timer mode operation

(CR000: compare register, CR010: compare register)

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

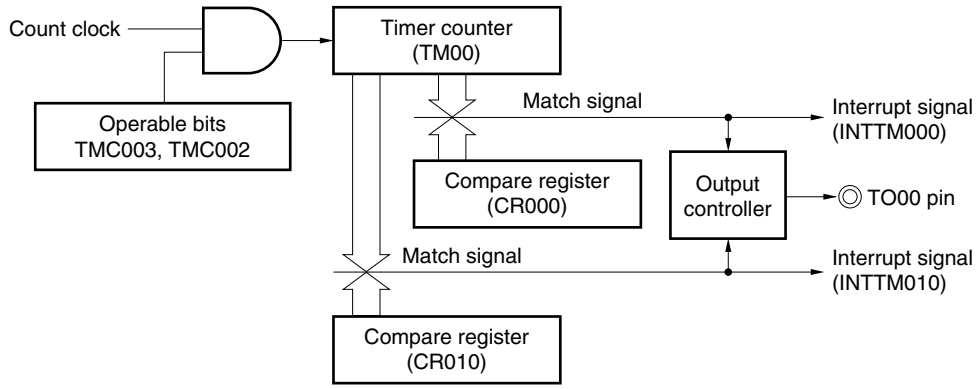
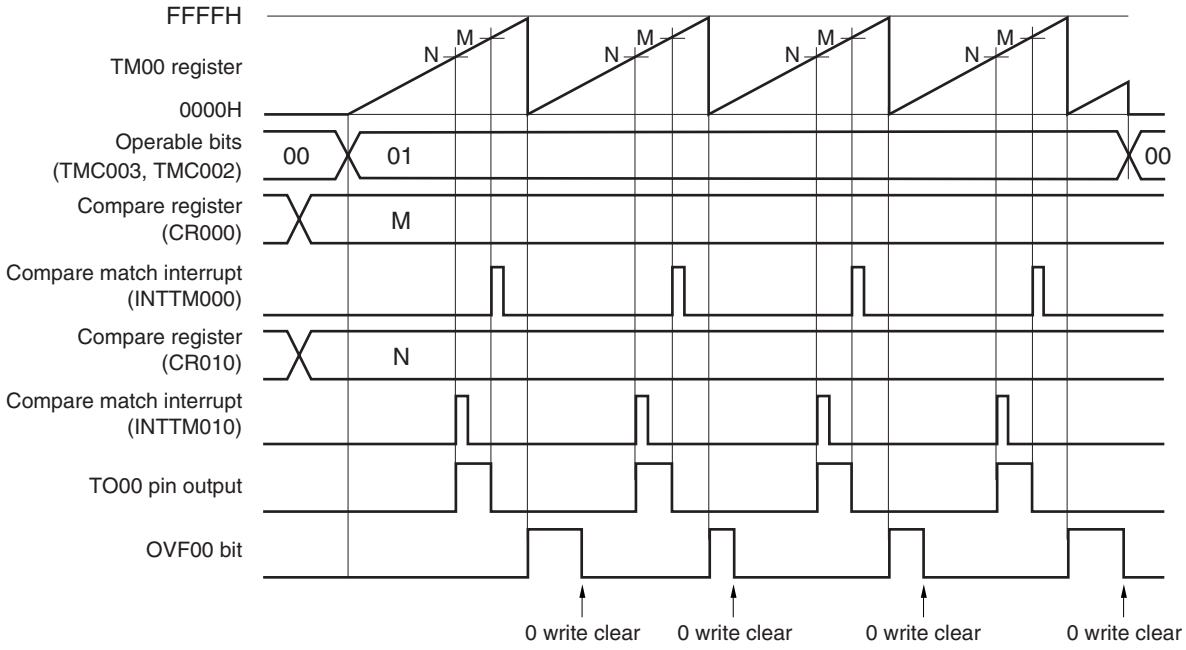


Figure 6-33. Timing Example of Free-Running Timer Mode
(CR000: Compare Register, CR010: Compare Register)

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



This is an application example where two compare registers are used in the free-running timer mode.
The output level of the TO00 pin is reversed each time the count value of TM00 matches the set value of CR000 or CR010. When the count value matches the register value, the INTTM000 or INTTM010 signal is generated.

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

Figure 6-34. Block Diagram of Free-Running Timer Mode
(CR000: Compare Register, CR010: Capture Register)

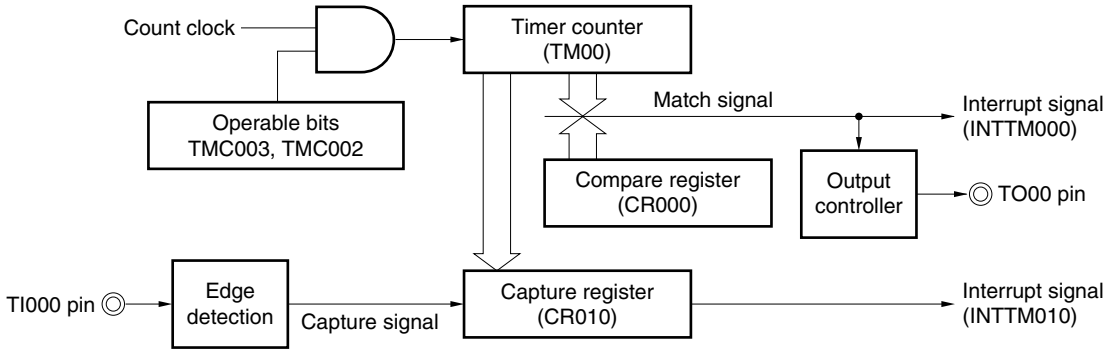
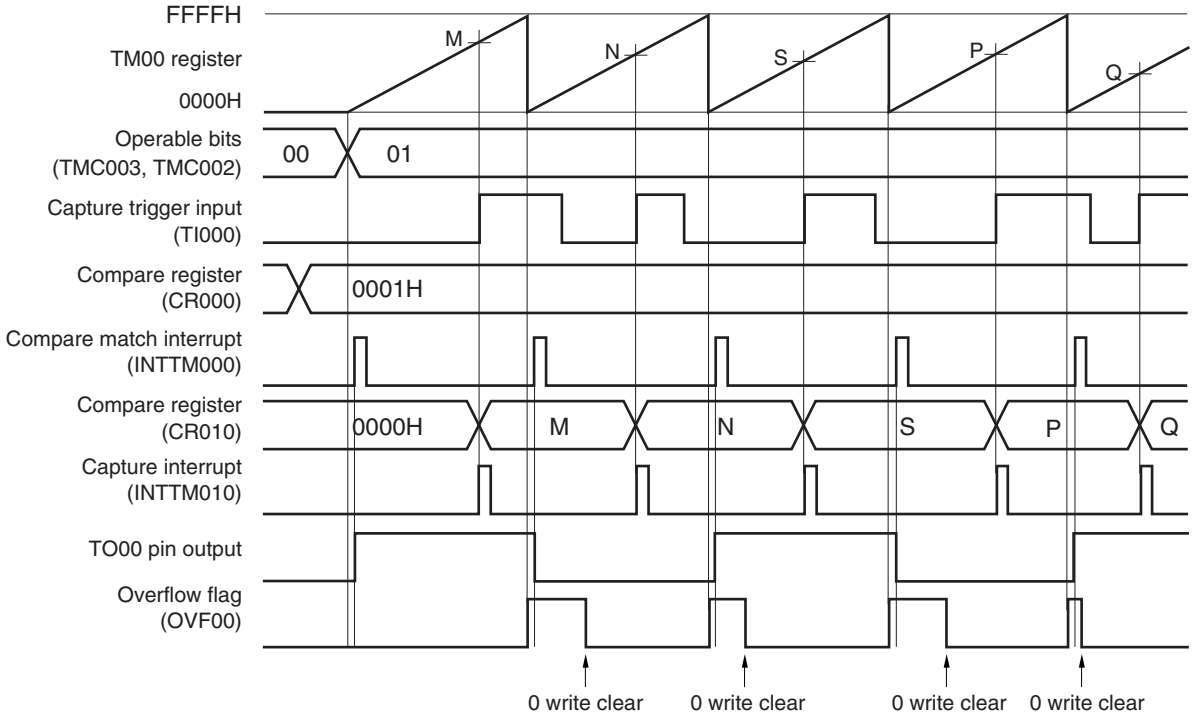


Figure 6-35. Timing Example of Free-Running Timer Mode
(CR000: Compare Register, CR010: Capture Register)

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

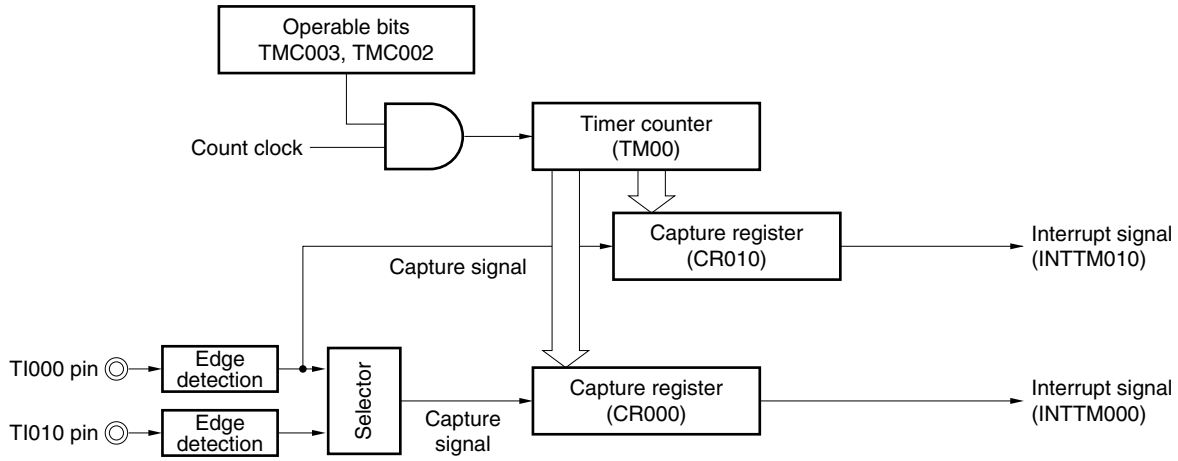


This is an application example where a compare register and a capture register are used at the same time in the free-running timer mode.

In this example, the INTTM000 signal is generated and the output level of the TO00 pin is reversed each time the count value of TM00 matches the set value of CR000 (compare register). In addition, the INTTM010 signal is generated and the count value of TM00 is captured to CR010 each time the valid edge of the TI000 pin is detected.

(3) Free-running timer mode operation
(CR000: capture register, CR010: capture register)

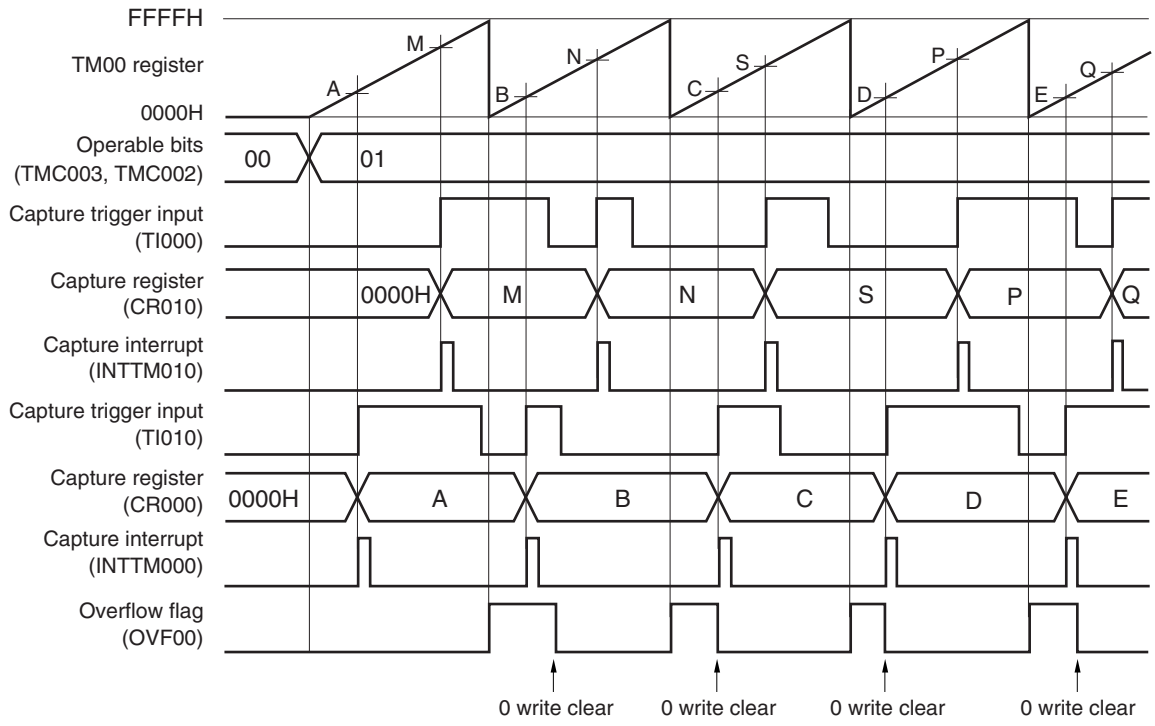
Figure 6-36. Block Diagram of Free-Running Timer Mode
(CR000: Capture Register, CR010: Capture Register)



Remark If both CR000 and CR010 are used as capture registers in the free-running timer mode, the output level of the TO00 pin is not inverted.
However, it can be inverted each time the valid edge of the TI000 pin is detected if bit 1 (TMC001) of 16-bit timer mode control register 00 (TMC00) is set to 1.

Figure 6-37. Timing Example of Free-Running Timer Mode (CR000: Capture Register, CR010: Capture Register) (1/2)

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

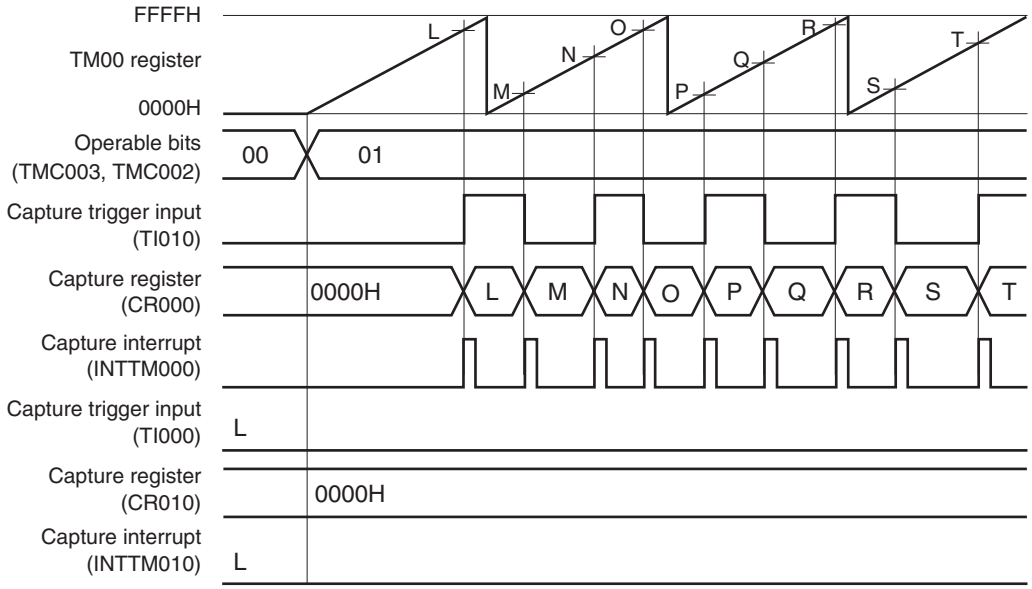


This is an application example where the count values that have been captured at the valid edges of separate capture trigger signals are stored in separate capture registers in the free-running timer mode.

The count value is captured to CR010 when the valid edge of the TI000 pin input is detected and to CR000 when the valid edge of the TI010 pin input is detected.

Figure 6-37. Timing Example of Free-Running Timer Mode (CR000: Capture Register, CR010: Capture Register) (2/2)

(b) TOC00 = 13H, PRM00 = C0H, CRC00 = 05H, TMC00 = 04H

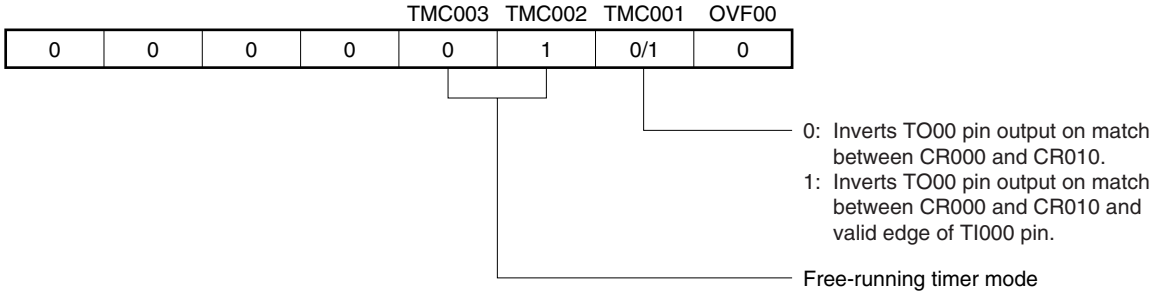


This is an application example where both the edges of the TI010 pin are detected and the count value is captured to CR000 in the free-running timer mode.

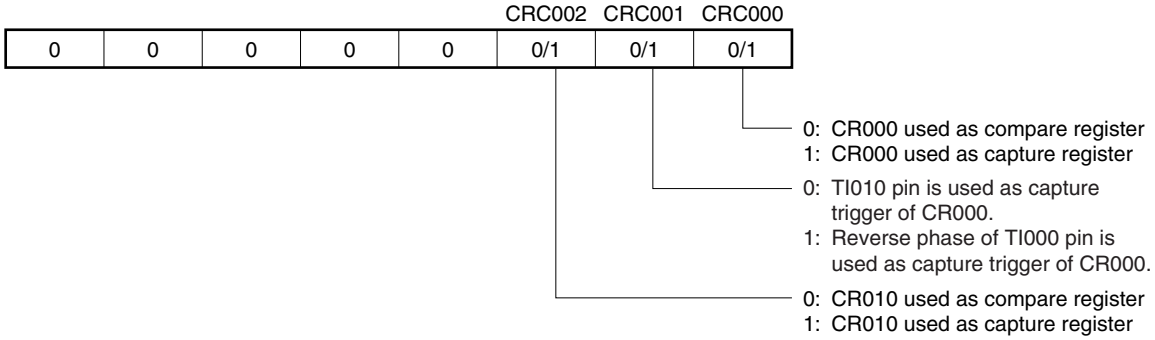
When both CR000 and CR010 are used as capture registers and when the valid edge of only the TI010 pin is to be detected, the count value cannot be captured to CR010.

Figure 6-38. Example of Register Settings in Free-Running Timer Mode (1/2)

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



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



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

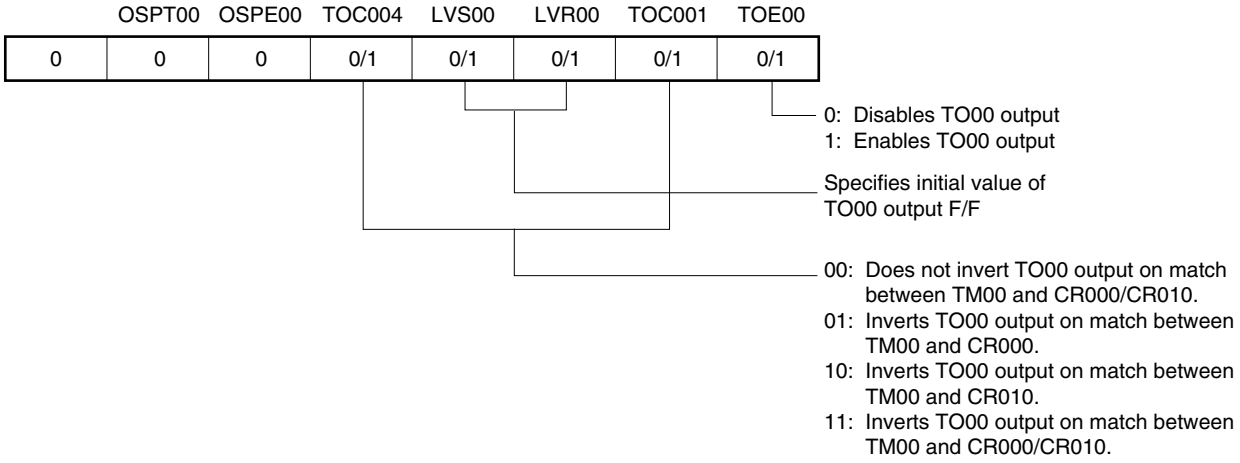
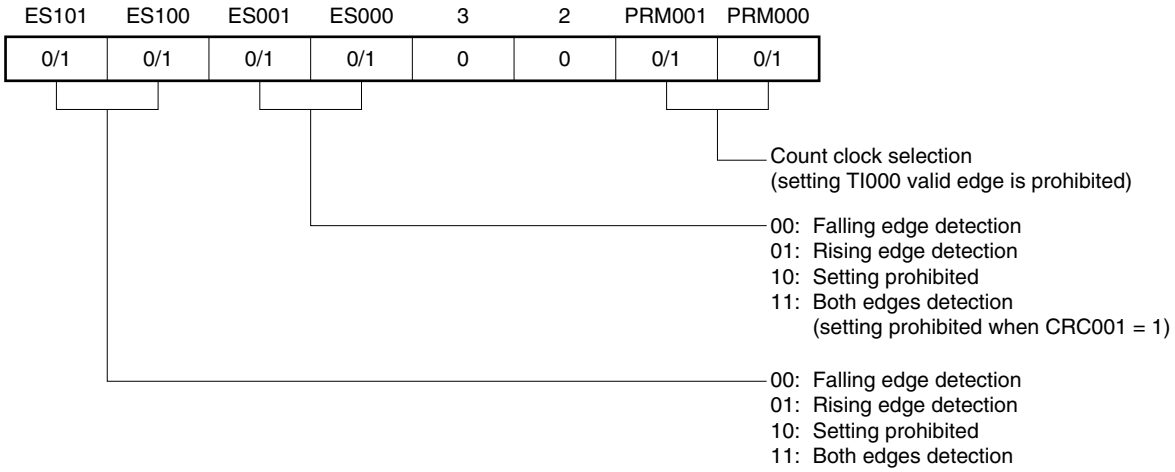


Figure 6-38. Example of Register Settings in Free-Running Timer Mode (2/2)

(d) Prescaler mode register 00 (PRM00)



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

By reading TM00, the count value can be read.

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

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

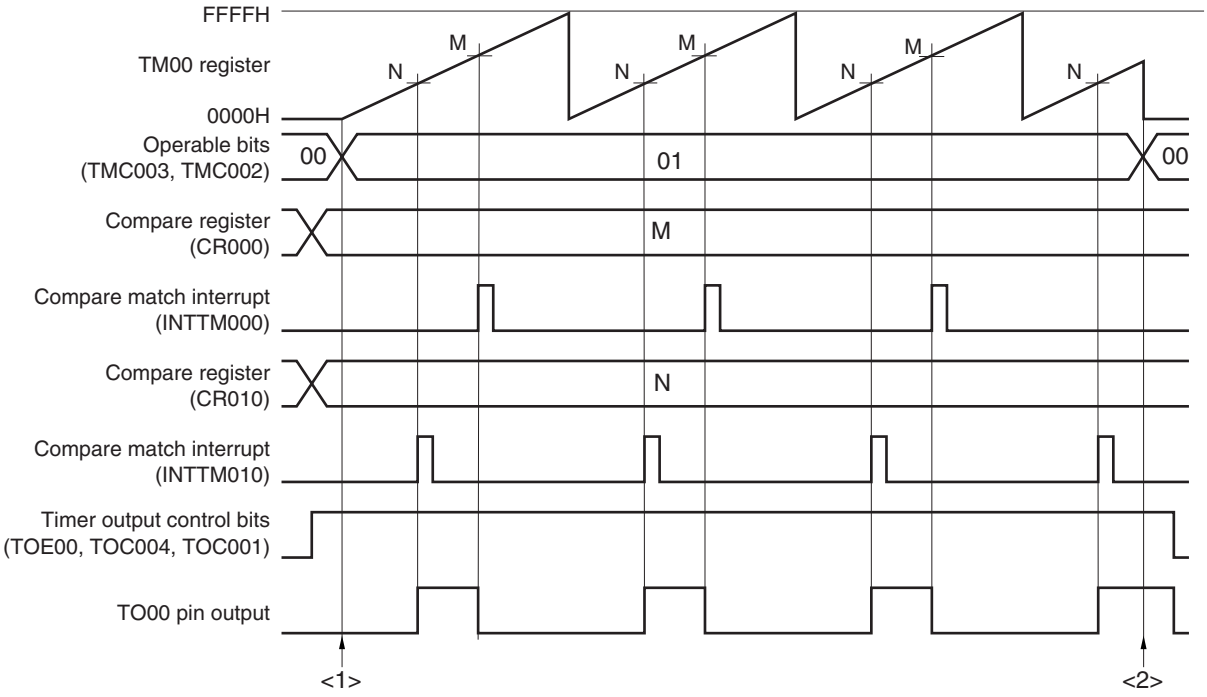
To use this register as a capture register, select either the TI000 or TI010 pin input as a capture trigger. When the valid edge of the capture trigger is detected, the count value of TM00 is stored in CR000.

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

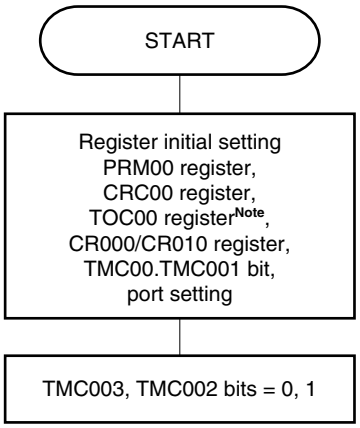
When this register is used as a compare register and when its value matches the count value of TM00, an interrupt signal (INTTM010) is generated. The count value of TM00 is not cleared.

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

Figure 6-39. Example of Software Processing in Free-Running Timer Mode



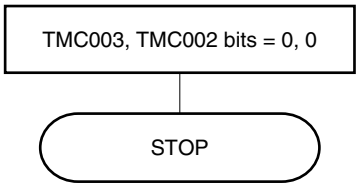
<1> Count operation start flow



Initial setting of these registers is performed before setting the TMC003 and TMC002 bits to 01.

Starts count operation

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TMC003 and TMC002 bits to 00.

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)**.

2. For how to enable the INTTM000 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

Figure 6-40. Block Diagram of PPG Output Operation

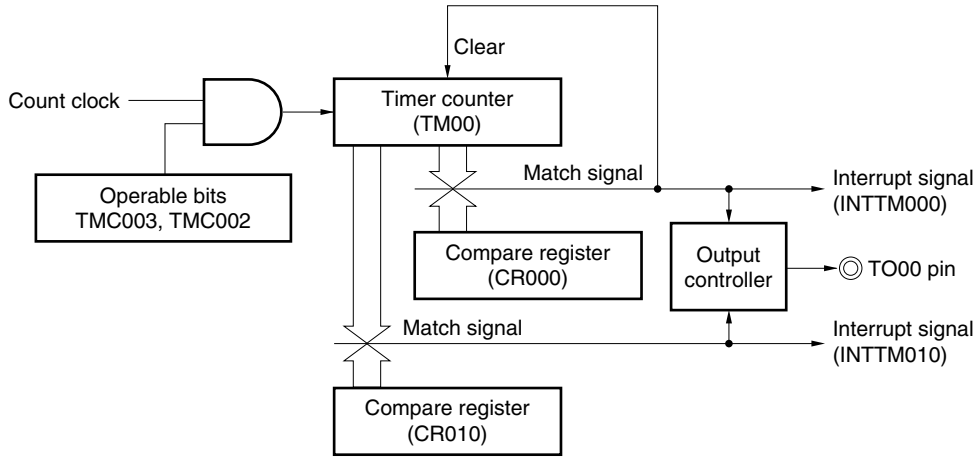
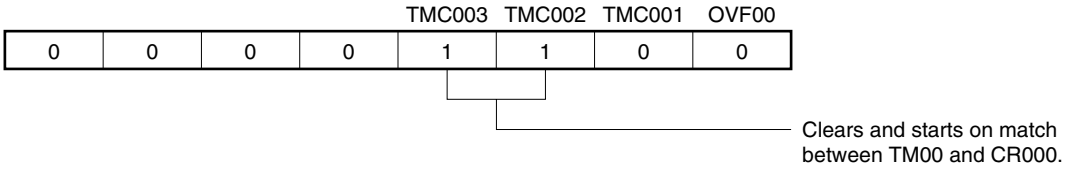
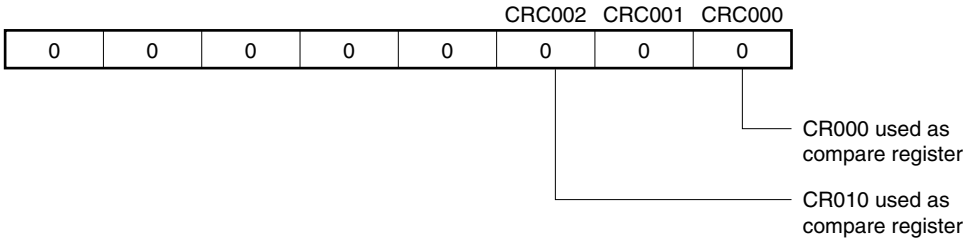


Figure 6-41. Example of Register Settings for PPG Output Operation

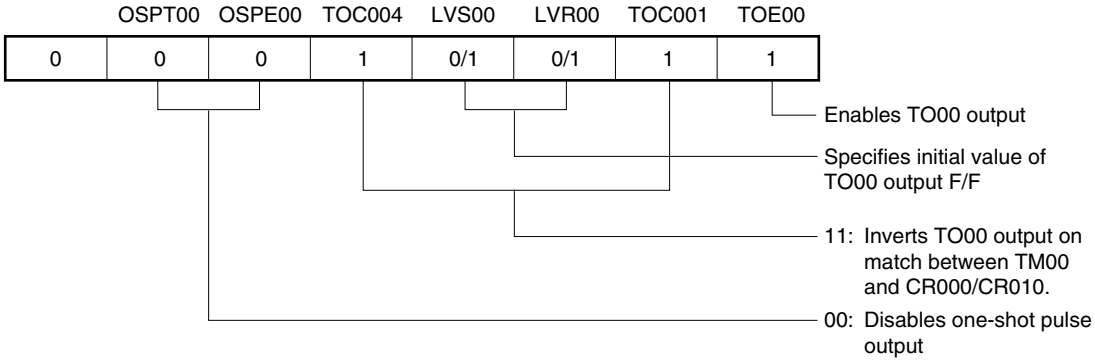
(a) 16-bit timer mode control register 00 (TMC00)



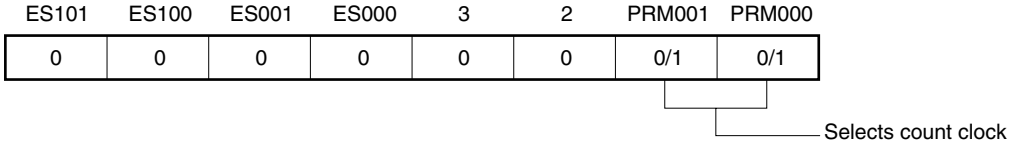
(b) Capture/compare control register 00 (CRC00)



(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)

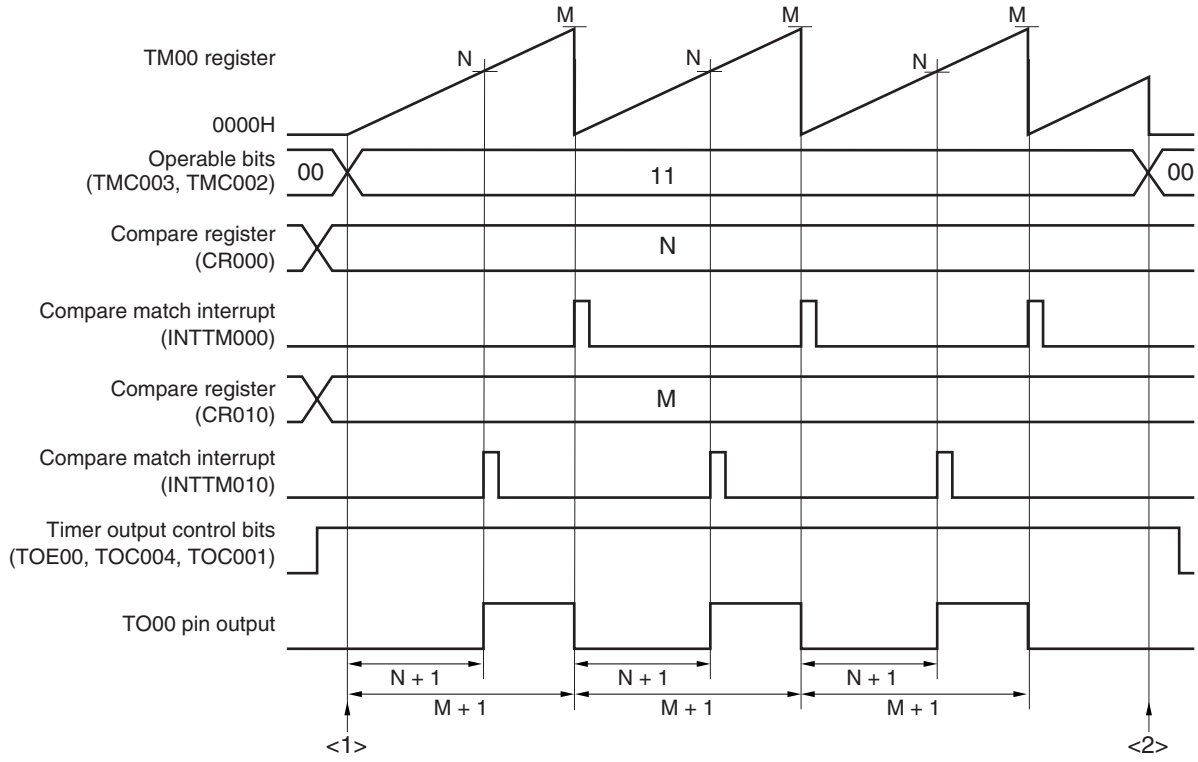
An interrupt signal (INTTM000) is generated when the value of this register matches the count value of TM00. The count value of TM00 is not cleared.

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

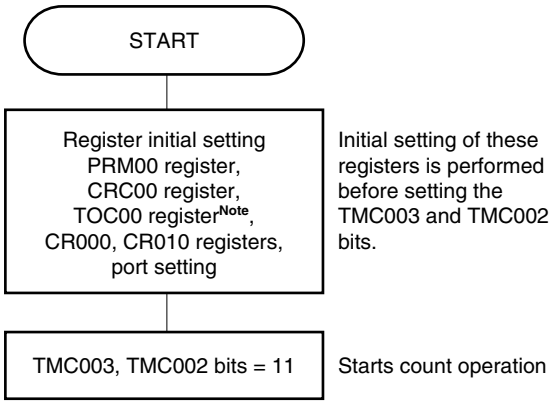
An interrupt signal (INTTM010) is generated when the value of this register matches the count value of TM00. The count value of TM00 is not cleared.

Caution Set values to CR000 and CR010 such that the condition 0000H < CR010 < CR000 ≤ FFFFH is satisfied.

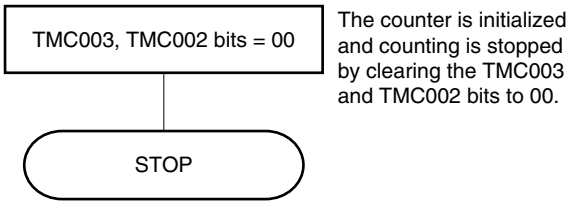
Figure 6-42. Example of Software Processing for PPG Output Operation



<1> Count operation start flow



<2> Count operation stop flow



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

Remark PPG pulse cycle = (M + 1) × Count clock cycle
PPG duty = (N + 1)/(M + 1)

6.4.7 One-shot pulse output operation

A one-shot pulse can be output by setting bits 3 and 2 (TMC003 and TMC002) of the 16-bit timer mode control register 00 (TMC00) to 01 (free-running timer mode) or to 10 (clear & start mode entered by the TI000 pin valid edge) and setting bit 5 (OSPE00) of 16-bit timer output control register 00 (TOC00) to 1.

When bit 6 (OSPT00) of TOC00 is set to 1 or when the valid edge is input to the TI000 pin during timer operation, clearing & starting of TM00 is triggered, and a pulse of the difference between the values of CR000 and CR010 is output only once from the TO00 pin.

Cautions 1. Do not input the trigger again (setting OSPT00 to 1 or detecting the valid edge of the TI000 pin) while the one-shot pulse is output. To output the one-shot pulse again, generate the trigger after the current one-shot pulse output has completed.

2. To use only the setting of OSPT00 to 1 as the trigger of one-shot pulse output, do not change the level of the TI000 pin or its alternate function port pin. Otherwise, the pulse will be unexpectedly output.

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

2. For how to enable the INTTM000 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

Figure 6-43. Block Diagram of One-Shot Pulse Output Operation

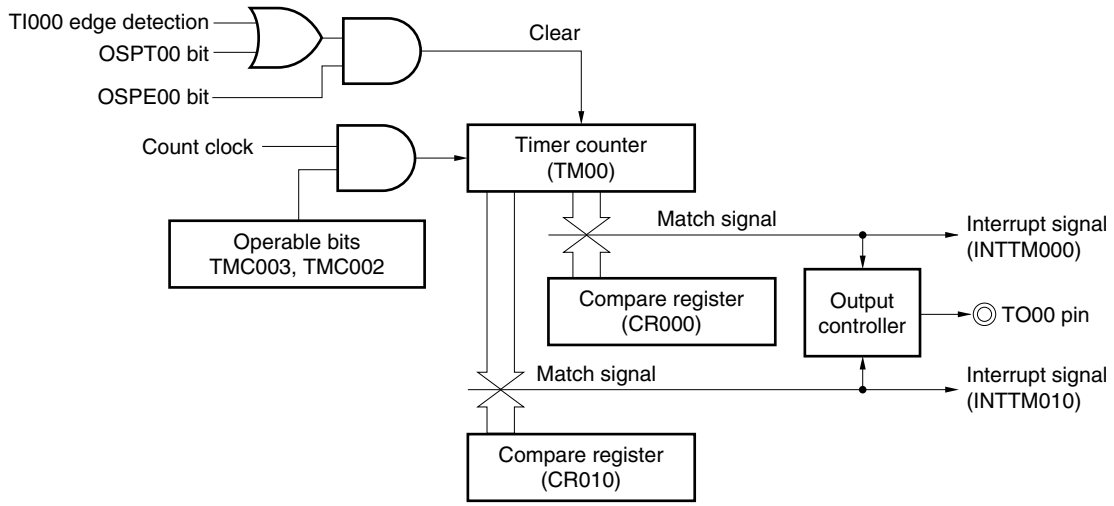
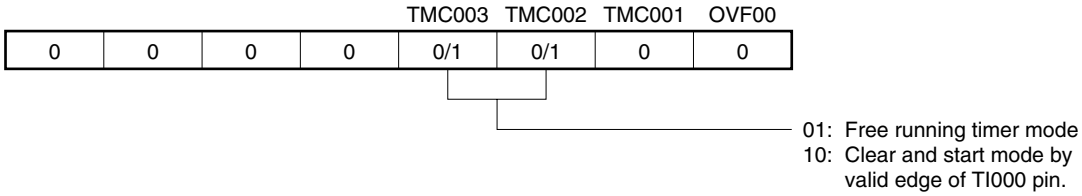
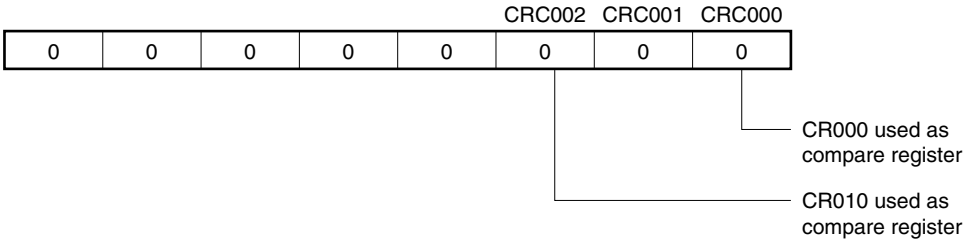


Figure 6-44. Example of Register Settings for One-Shot Pulse Output Operation (1/2)

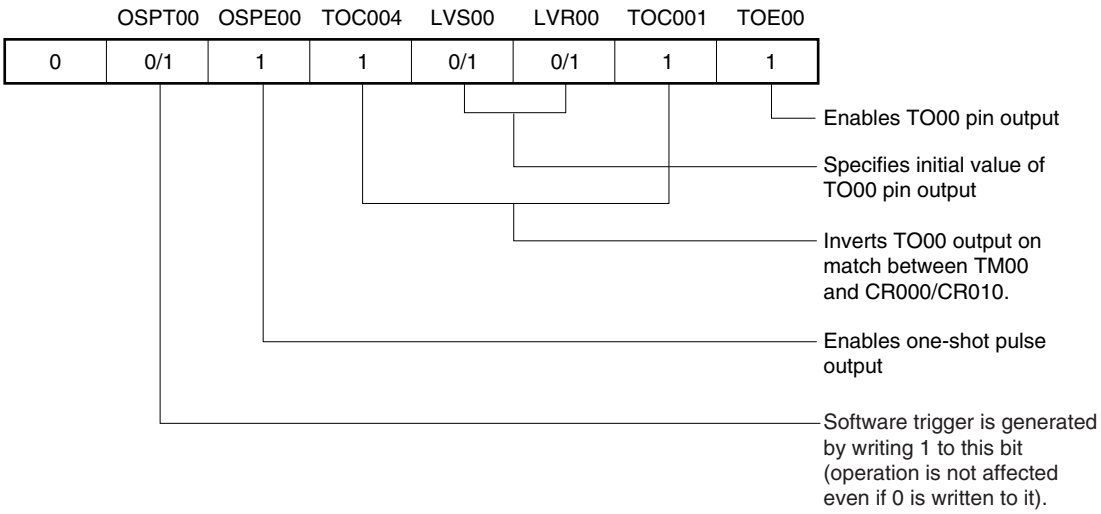
(a) 16-bit timer mode control register 00 (TMC00)



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



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



(d) Prescaler mode register 00 (PRM00)

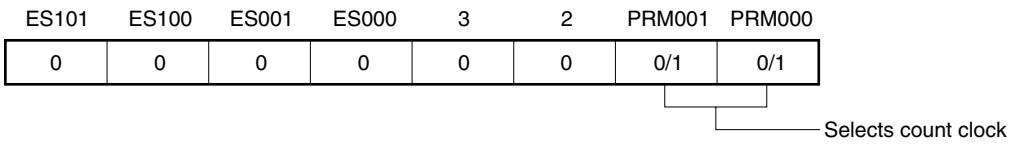


Figure 6-44. Example of Register Settings for One-Shot Pulse Output Operation (2/2)

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

By reading TM00, the count value can be read.

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

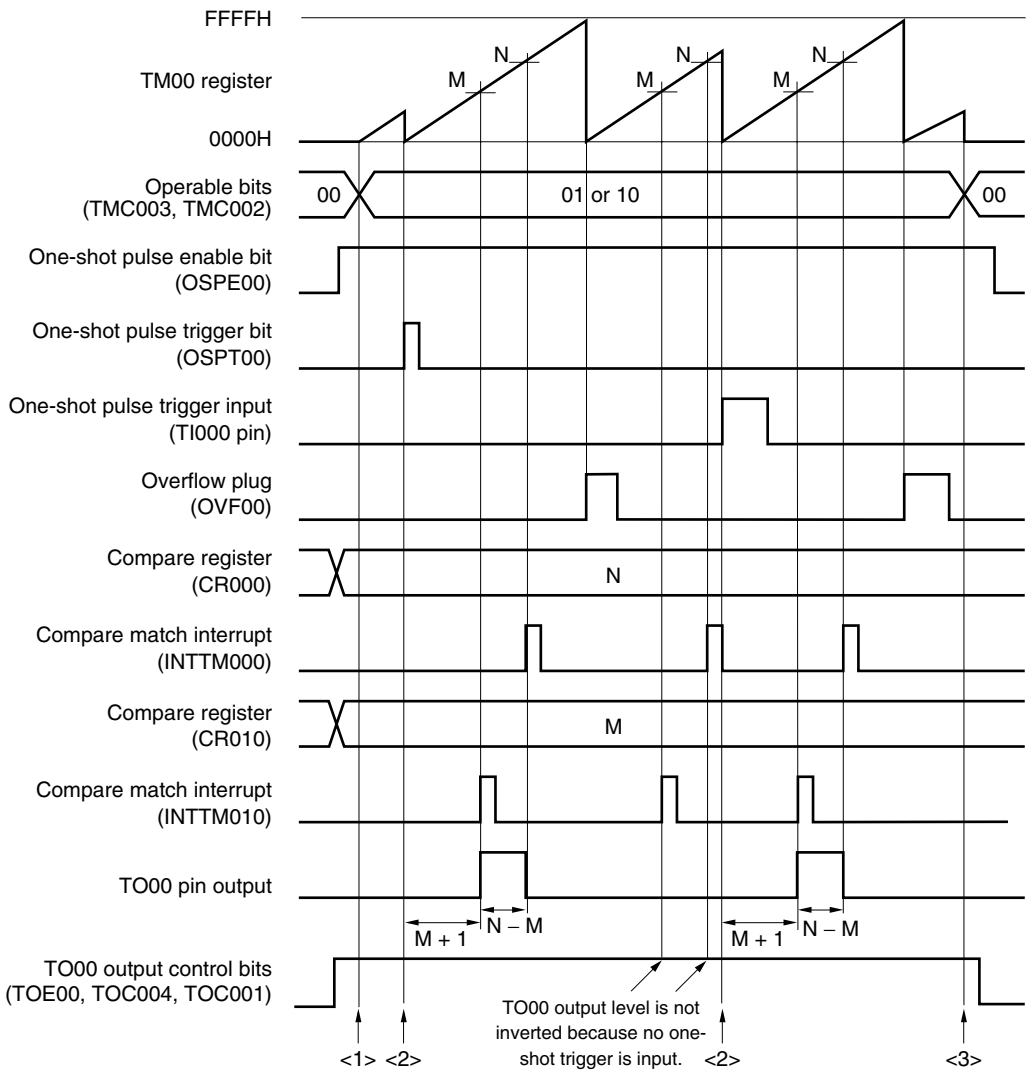
This register is used as a compare register when a one-shot pulse is output. When the value of TM00 matches that of CR000, an interrupt signal (INTTM000) is generated and the output level of the TO00 pin is inverted.

(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 output level of the TO00 pin is inverted.

Caution Do not set identical values or 0000H for CR000 and CR001.

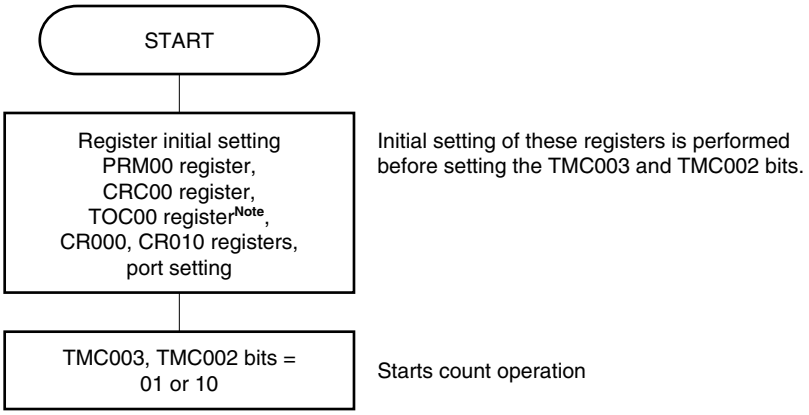
Figure 6-45. Example of Software Processing for One-Shot Pulse Output Operation (1/2)



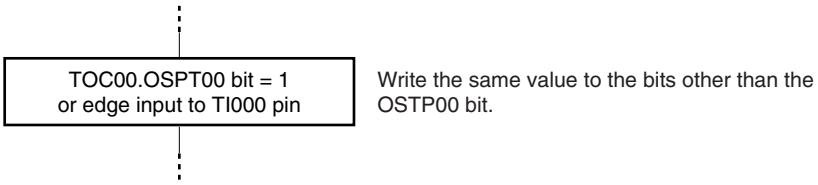
- Time from when the one-shot pulse trigger is input until the one-shot pulse is output
  = (M + 1) × Count clock cycle
- One-shot pulse output active level width
  = (N – M) × Count clock cycle

Figure 6-45. Example of Software Processing for One-Shot Pulse Output Operation (2/2)

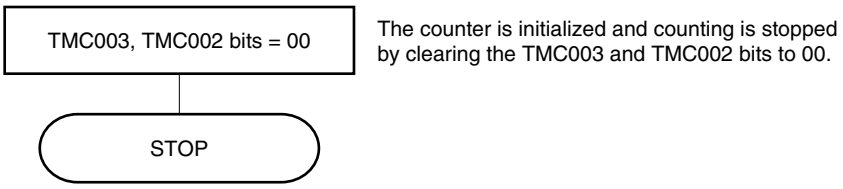
<1> Count operation start flow



<2> One-shot trigger input flow



<3> Count operation stop flow



Note Care must be exercised when setting TOC00. For details, 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 6-46. Block Diagram of Pulse Width Measurement (Free-Running Timer Mode)

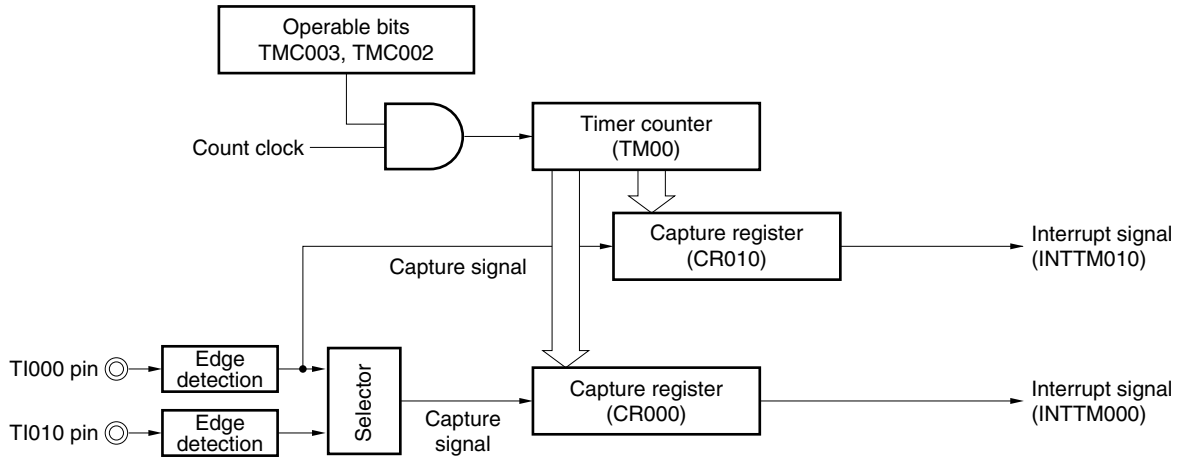
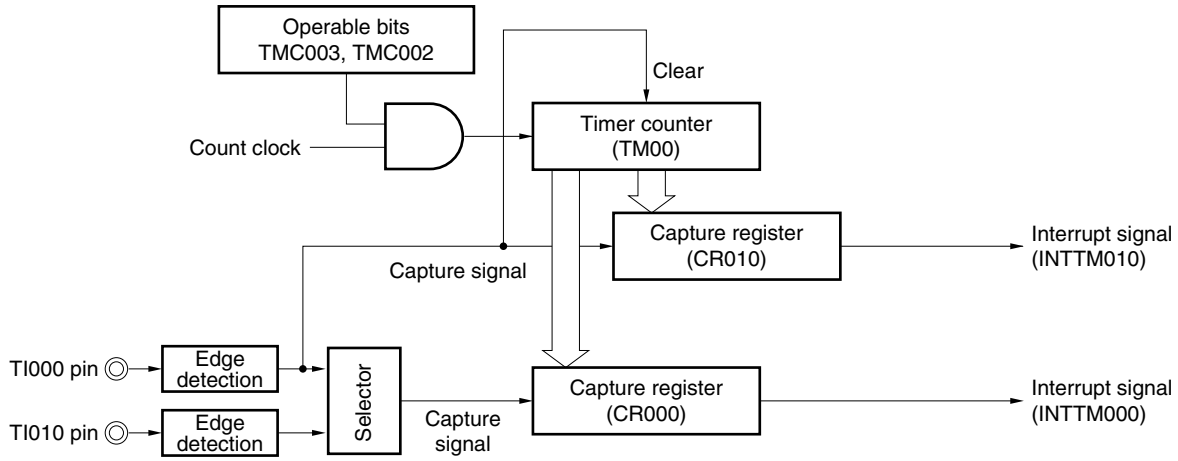


Figure 6-47. Block Diagram of Pulse Width Measurement
(Clear & Start Mode Entered by TI000 Pin Valid Edge Input)



A pulse width can be measured in the following three ways.

- Measuring the pulse width by using two input signals of the TI000 and TI010 pins (free-running timer mode)
- Measuring the pulse width by using one input signal of the TI000 pin (free-running timer mode)
- Measuring the pulse width by using one input signal of the TI000 pin (clear & start mode entered by the TI000 pin valid edge input)

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

2. For how to enable the INTTM000 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

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

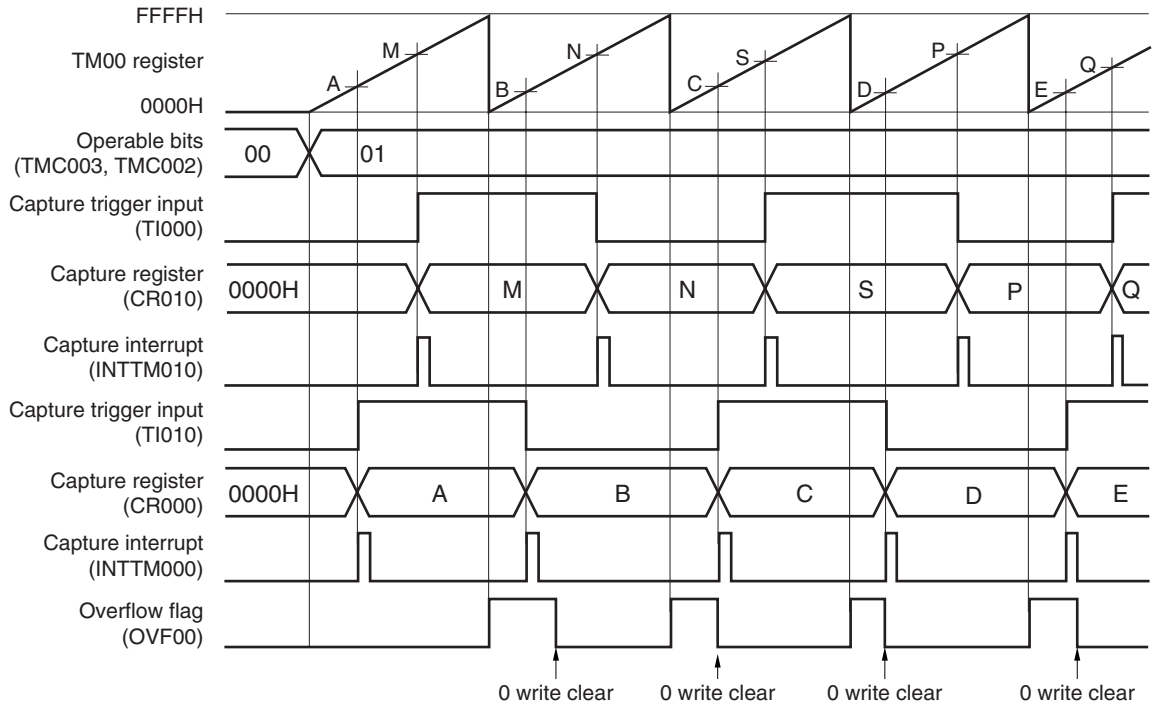
Set the free-running timer mode (TMC003 and TMC002 = 01). When the valid edge of the TI000 pin is detected, the count value of TM00 is captured to CR010. When the valid edge of the TI010 pin is detected, the count value of TM00 is captured to CR000. Specify detection of both the edges of the TI000 and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

By this measurement method, the previous count value is subtracted from the count value captured by the edge of each input signal. Therefore, save the previously captured value to a separate register in advance.

If an overflow occurs, the value becomes negative if the previously captured value is simply subtracted from the current captured value and, therefore, a borrow occurs (bit 0 (CY) of the program status word (PSW) is set to 1). If this happens, ignore CY and take the calculated value as the pulse width. In addition, clear bit 0 (OVF00) of 16-bit timer mode control register 00 (TMC00) to 0.

Figure 6-48. Timing Example of Pulse Width Measurement (1)

• TMC00 = 04H, PRM00 = F0H, CRC00 = 05H



(2) Measuring the pulse width by using one input signal of the TI000 pin (free-running mode)

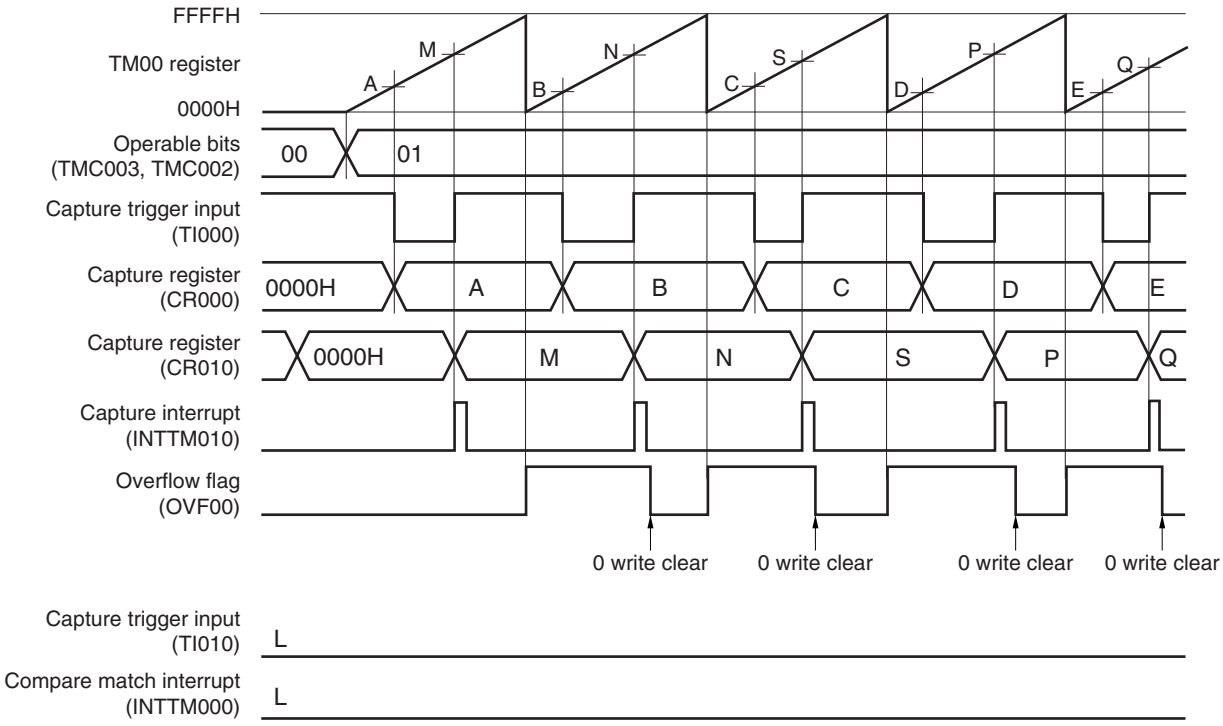
Set the free-running timer mode (TMC003 and TMC002 = 01). The count value of TM00 is captured to CR000 in the phase reverse to the valid edge detected on the TI000 pin. When the valid edge of the TI000 pin is detected, the count value of TM00 is captured to CR010.

By this measurement method, values are stored in separate capture registers when a width from one edge to another is measured. Therefore, the capture values do not have to be saved. By subtracting the value of one capture register from that of another, a high-level width, low-level width, and cycle are calculated.

If an overflow occurs, the value becomes negative if one captured value is simply subtracted from another and, therefore, a borrow occurs (bit 0 (CY) of the program status word (PSW) is set to 1). If this happens, ignore CY and take the calculated value as the pulse width. In addition, clear bit 0 (OVF00) of 16-bit timer mode control register 00 (TMC00) to 0.

Figure 6-49. Timing Example of Pulse Width Measurement (2)

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



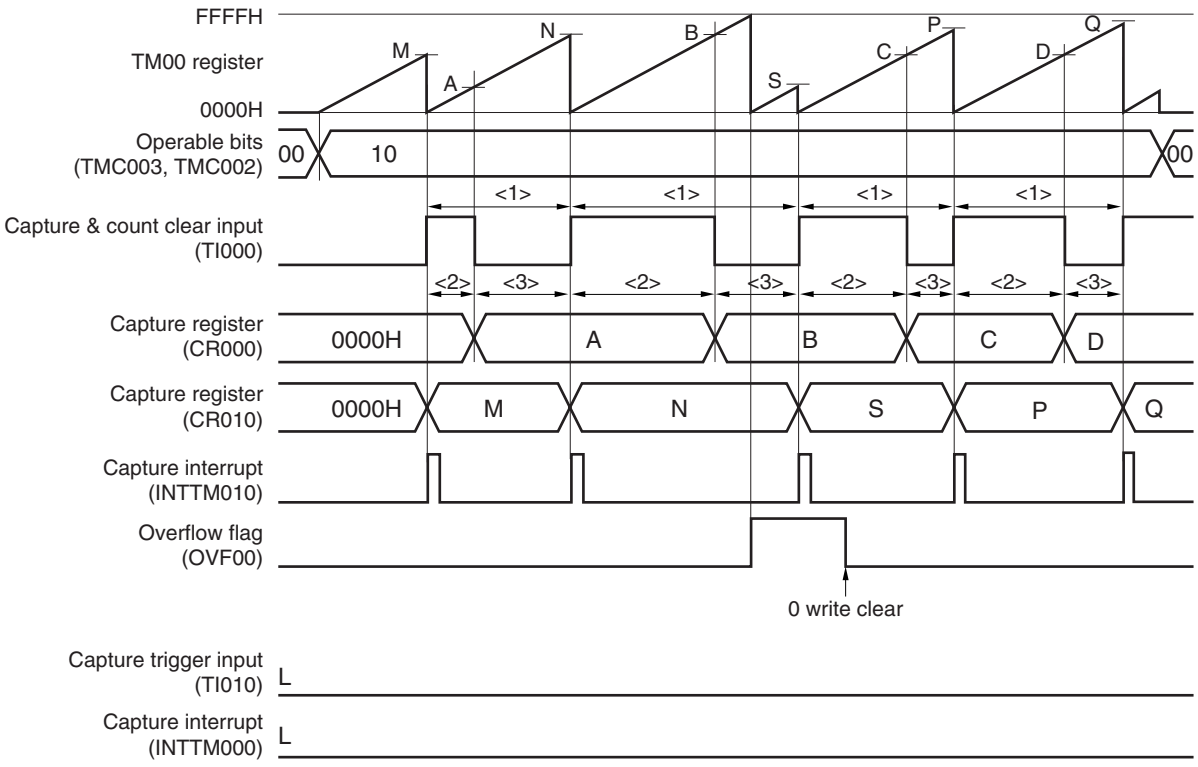
(3) Measuring the pulse width by using one input signal of the TI000 pin (clear & start mode entered by the TI000 pin valid edge input)

Set the clear & start mode entered by the TI000 pin valid edge (TMC003 and TMC002 = 10). The count value of TM00 is captured to CR000 in the phase reverse to the valid edge of the TI000 pin, and the count value of TM00 is captured to CR010 and TM00 is cleared (0000H) when the valid edge of the TI000 pin is detected. Therefore, a cycle is stored in CR010 if TM00 does not overflow.

If an overflow occurs, take the value that results from adding 10000H to the value stored in CR010 as a cycle. Clear bit 0 (OVF00) of 16-bit timer mode control register 00 (TMC00) to 0.

Figure 6-50. Timing Example of Pulse Width Measurement (3)

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



<1> Pulse cycle = (10000H × Number of times OVF00 bit is set to 1 + Captured value of CR010) × Count clock cycle

<2> High-level pulse width = (10000H × Number of times OVF00 bit is set to 1 + Captured value of CR000) × Count clock cycle

<3> Low-level pulse width = (Pulse cycle − High-level pulse width)

Figure 6-51. Example of Register Settings for Pulse Width Measurement (1/2)

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

				TMC003	TMC002	TMC001	OVF00
0	0	0	0	0/1	0/1	0	0

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

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

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

1: CR000 used as capture register

0: 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)

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

(d) Prescaler mode register 00 (PRM00)

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

Selects count clock
(setting valid edge of 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 6-51. Example of Register Settings for Pulse Width Measurement (2/2)

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

By reading TM00, the count value can be read.

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

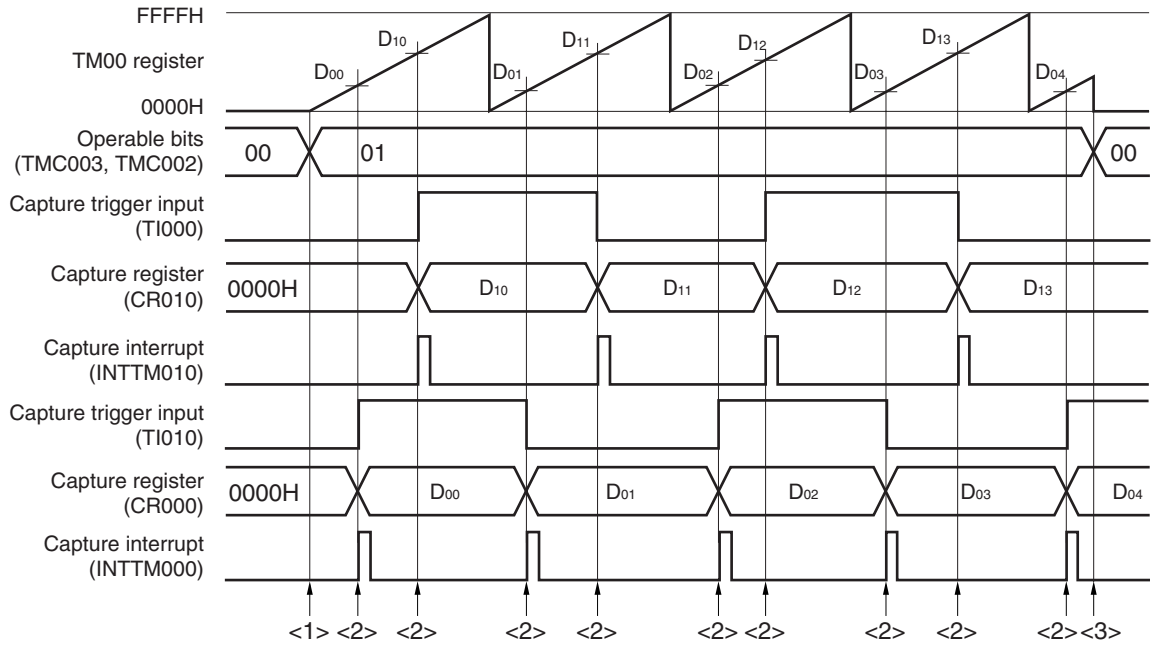
This register is used as a capture register. Either the TI000 or TI010 pin is selected as a capture trigger. When a specified edge of the capture trigger is detected, the count value of TM00 is stored in CR000.

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

This register is used as a capture register. The signal input to the TI000 pin is used as a capture trigger. When the capture trigger is detected, the count value of TM00 is stored in CR010.

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

(a) Example of free-running timer mode



(b) Example of clear & start mode entered by TI000 pin valid edge

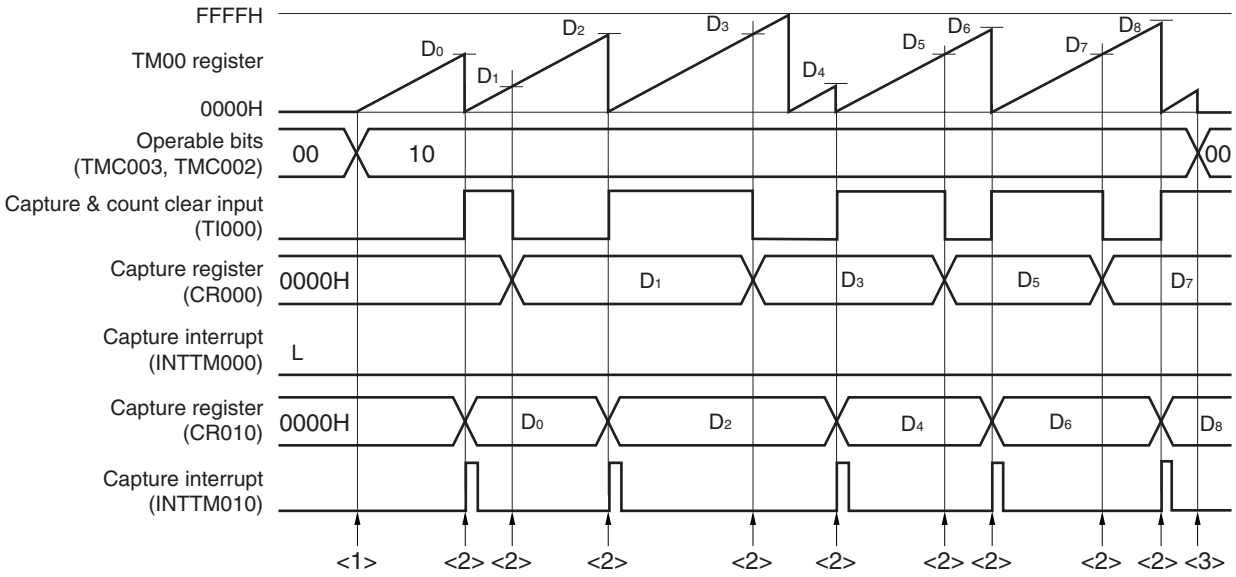
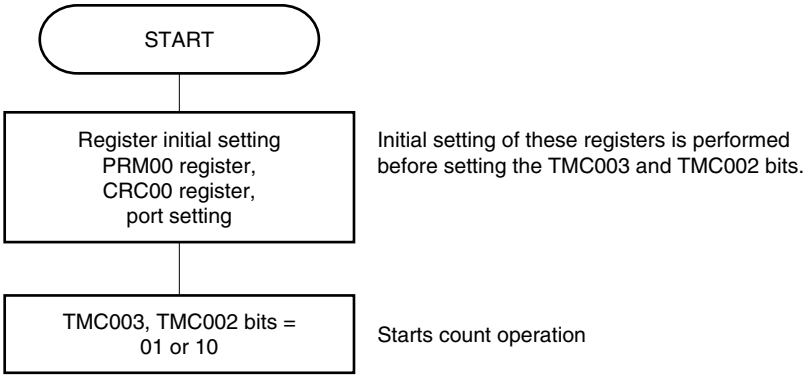
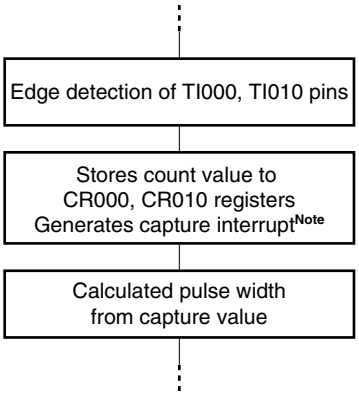


Figure 6-52. Example of Software Processing for Pulse Width Measurement (2/2)

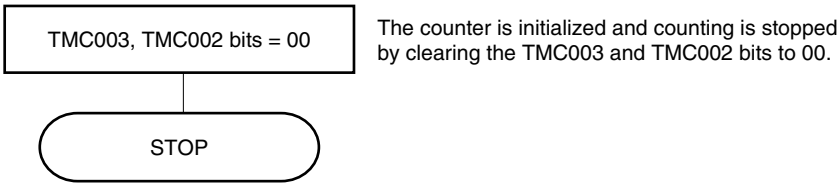
<1> Count operation start flow



<2> Capture trigger input flow



<3> Count operation stop flow



Note The capture interrupt signal (INTTM000) is not generated when the reverse-phase edge of the TI000 pin input is selected to the valid edge of CR000.

6.5 Special Use of TM00

6.5.1 Rewriting CR010 during TM00 operation

In principle, rewriting CR000 and CR010 of the μPD78F0730 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 (change the value of CR010 immediately after its value matches the value of TM00. If the value of CR010 is changed immediately before its value matches 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 13 INTERRUPT FUNCTIONS**.

6.5.2 Setting LVS00 and LVR00

(1) Usage of LVS00 and LVR00

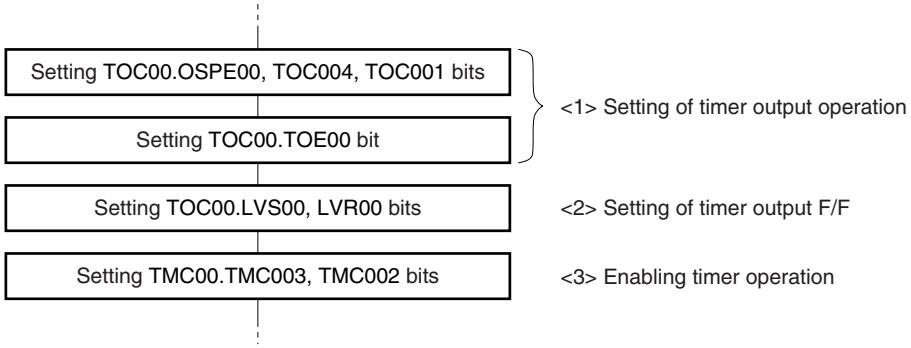
LVS00 and LVR00 are used to set the default value of the TO00 pin output and to invert the timer output without enabling the timer operation (TMC003 and TMC002 = 00). Clear LVS00 and LVR00 to 00 (default value: low-level output) when software control is unnecessary.

LVS00	LVR00	Timer Output Status
0	0	Not changed (low-level output)
0	1	Cleared (low-level output)
1	0	Set (high-level output)
1	1	Setting prohibited

(2) Setting LVS00 and LVR00

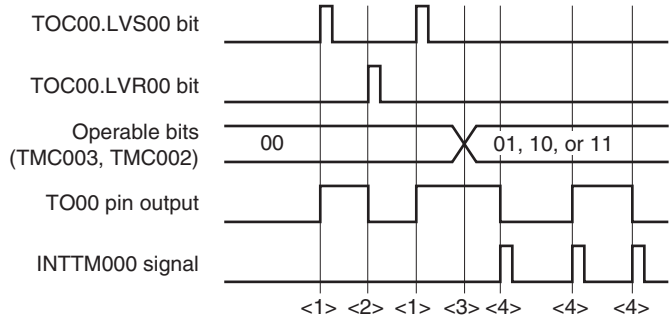
Set LVS00 and LVR00 using the following procedure.

Figure 6-53. Example of Flow for Setting LVS00 and LVR00 Bits



Caution Be sure to set LVS00 and LVR00 following steps <1>, <2>, and <3> above.
Step <2> can be performed after <1> and before <3>.

Figure 6-54. Timing Example of LVR00 and LVS00



<1> The TO00 pin output goes high when LVS00 and LVR00 = 10.

<2> The TO00 pin output goes low when LVS00 and LVR00 = 01 (the pin output remains unchanged from the high level even if LVS00 and LVR00 are cleared to 00).

<3> The timer starts operating when TMC003 and TMC002 are set to 01, 10, or 11. Because LVS00 and LVR00 were set to 10 before the operation was started, the TO00 pin 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 output level of the TO00 pin 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.

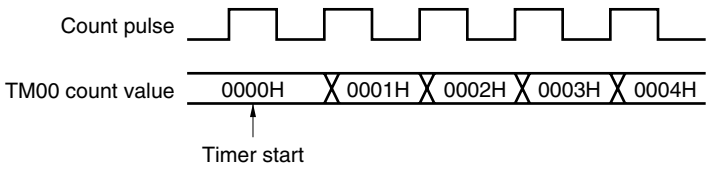
Table 6-3. Restrictions for Each Channel of 16-Bit Timer/Event Counter 00

Operation	Restriction
As interval timer	–
As square wave output	
As external event counter	TOC00 = 00H
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	Setting identical values or 0000H to CR000 and CP010 is prohibited.
As one-shot pulse output	–
As pulse width measurement	TOC00 = 00H

(2) Timer start errors

An error of up to one clock may occur in the time required for a match signal to be generated after timer start. This is because counting TM00 is started asynchronously to the count pulse.

Figure 6-55. Start Timing of TM00 Count



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

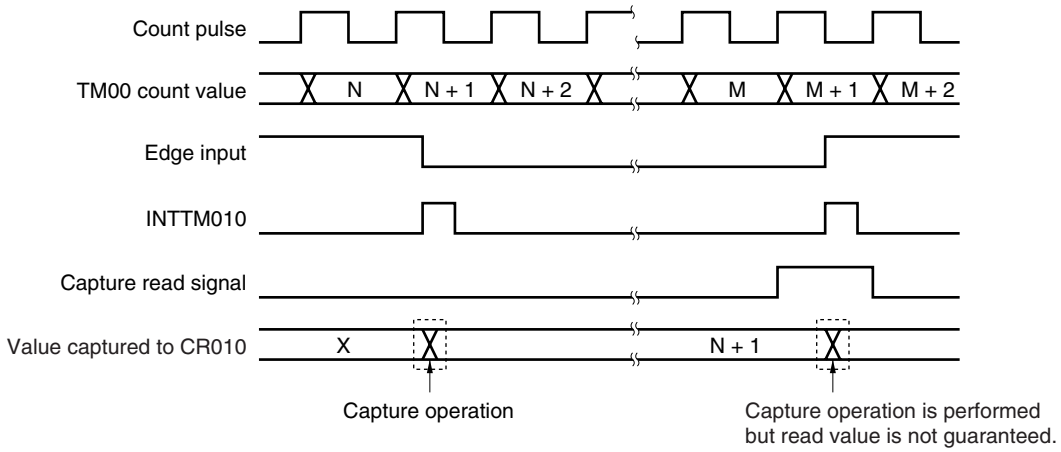
Set a value other than 0000H to CR000 and CR010 (TM00 cannot count one pulse when it is used as an external event counter).

(4) Timing of holding data by capture register

(a) When the valid edge is input to the TI000/TI010 pin and the reverse phase of the TI000 pin is detected while CR000/CR010 is read, CR010 performs a capture operation but the read value of CR000/CR010 is not guaranteed. At this time, an interrupt signal (INTTM000/INTTM010) is generated when the valid edge of the TI000/TI010 pin is detected (the interrupt signal is not generated when the reverse-phase edge of the TI000 pin is detected).

When the count value is captured because the valid edge of the TI000/TI010 pin was detected, read the value of CR000/CR010 after INTTM000/INTTM010 is generated.

Figure 6-56. Timing of Holding Data by Capture Register



(b) The values of CR000 and CR010 are not guaranteed after 16-bit timer/event counter 00 stops.

(5) Setting valid edge

Set the valid edge of the TI000 pin while the timer operation is stopped (TMC003 and TMC002 = 00). Set the valid edge by using ES000 and 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.

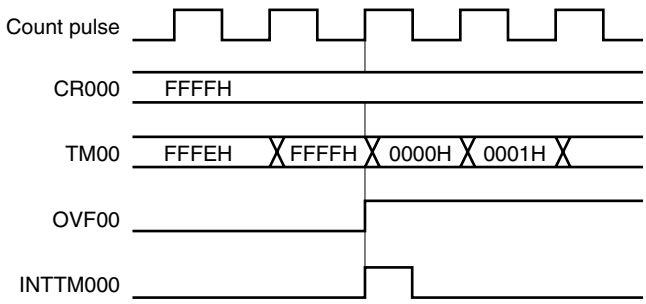
↓

Set CR000 to FFFFH.

↓

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

Figure 6-57. Operation Timing of OVF00 Flag



(b) Clearing OVF00 flag

Even if the OVF00 flag is cleared to 0 after TM00 overflows and before the next count clock is counted (before the value of TM00 becomes 0001H), it is set to 1 again and clearing is invalid.

(8) One-shot pulse output

One-shot pulse output operates correctly in the free-running timer mode or the clear & start mode entered by the TI000 pin valid edge. The one-shot pulse cannot be output in the clear & start mode entered upon a match between TM00 and CR000.

(9) Capture operation

(a) When valid edge of TI000 is specified as count clock

When the valid edge of TI000 is specified as the count clock, the capture register for which TI000 is specified as a trigger does not operate correctly.

(b) Pulse width to accurately capture value by signals input to TI010 and TI000 pins

To accurately capture the count value, the pulse input to the TI000 and TI010 pins as a capture trigger must be wider than two count clocks selected by PRM00 (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 1 (PM1) or port mode register 3 (PM3) Port register 1 (P1) or port register 3 (P3)

Figures 7-1 and 7-2 show the block diagrams of 8-bit timer/event counters 50 and 51.

Figure 7-1. Block Diagram of 8-Bit Timer/Event Counter 50

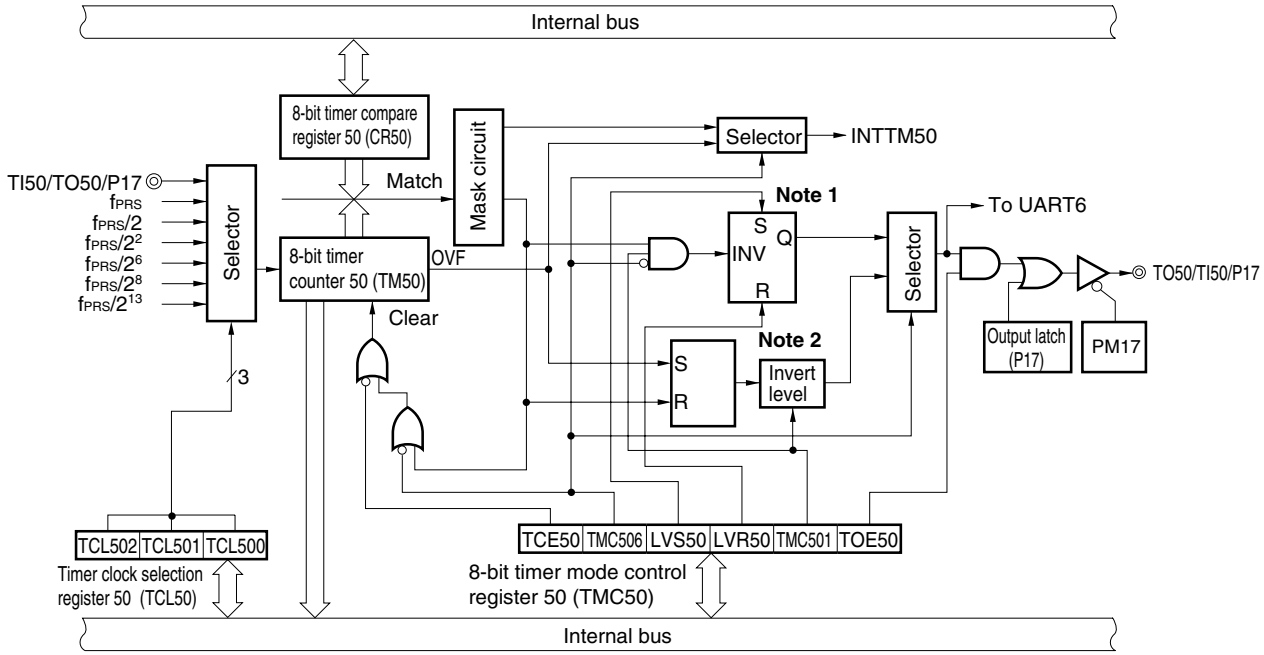
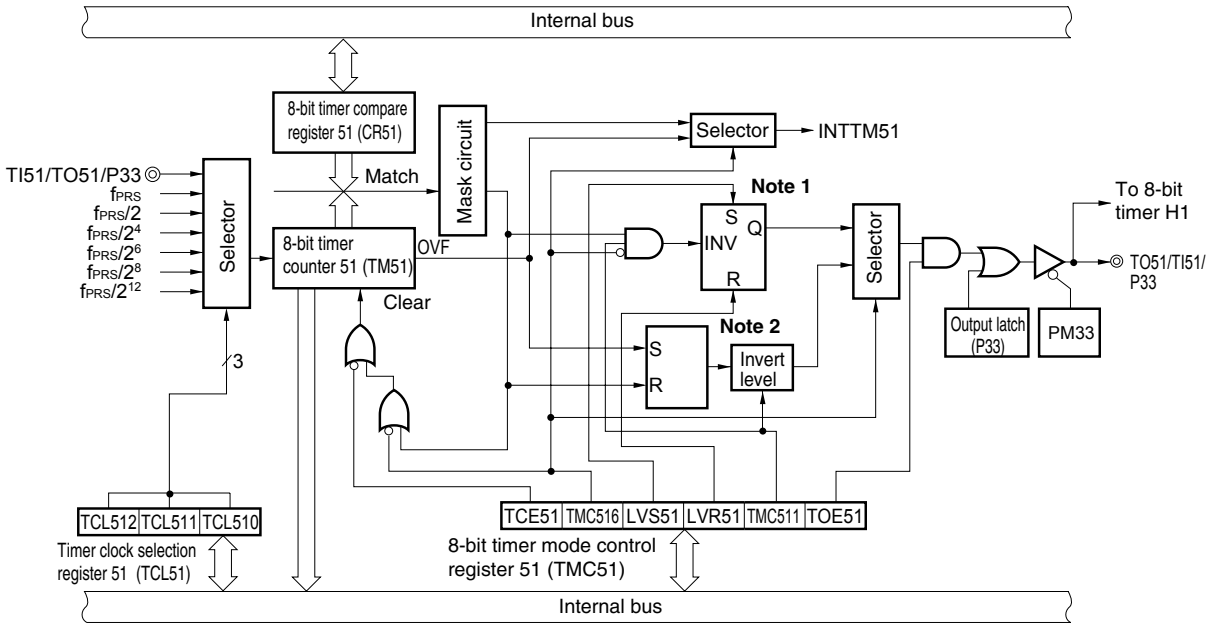


Figure 7-2. Block Diagram of 8-Bit Timer/Event Counter 51



Notes 1. Timer output F/F

2. PWM output F/F

(1) 8-bit timer counter 5n (TM5n)

TM5n is an 8-bit register that counts the count pulses and is read-only.

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

Figure 7-3. Format of 8-Bit Timer Counter 5n (TM5n)

Address: FF16H (TM50), FF1FH (TM51) After reset: 00H R

Symbol	7	6	5	4	3	2	1	0
TM5n (n = 0, 1)								

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, the TO5n pin 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 sets CR5n to 00H.

Figure 7-4. Format of 8-Bit Timer Compare Register 5n (CR5n)

Address: FF17H (CR50), FF41H (CR51) After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CR5n (n = 0, 1)								

Cautions 1. In the mode in which clear & start occurs on a match of TM5n and CR5n (TMC5n6 = 0), do not write other values to CR5n during operation.

2. In PWM mode, make the CR5n rewrite period 3 count clocks of the count clock (clock selected by TCL5n) or more.

Remark n = 0, 1

7.3 Registers Controlling 8-Bit Timer/Event Counters 50 and 51

The following four registers are used to control 8-bit timer/event counters 50 and 51.

- Timer clock selection register 5n (TCL5n)
- 8-bit timer mode control register 5n (TMC5n)
- Port mode register 1 (PM1) or port mode register 3 (PM3)
- Port register 1 (P1) or port register 3 (P3)

(1) Timer clock selection register 5n (TCL5n)

This register sets the count clock of 8-bit timer/event counter 5n and the valid edge of the TI5n pin input.

TCL5n can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets TCL5n to 00H.

Remark n = 0, 1

Figure 7-5. Format of Timer Clock Selection Register 50 (TCL50)

Address: FF6AH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
TCL50	0	0	0	0	0	TCL502	TCL501	TCL500

TCL502	TCL501	TCL500	Count clock selection		
				f_{PRS} = 12 MHz	f_{PRS} = 16 MHz
0	0	0	TI50 pin falling edge		
0	0	1	TI50 pin rising edge		
0	1	0	f_{PRS}	12 MHz	16 MHz
0	1	1	$f_{PRS}/2$	6 MHz	8 MHz
1	0	0	$f_{PRS}/2^2$	3 MHz	4 MHz
1	0	1	$f_{PRS}/2^6$	187.5 kHz	250 kHz
1	1	0	$f_{PRS}/2^8$	46.88 kHz	62.5 kHz
1	1	1	$f_{PRS}/2^{13}$	1.46 kHz	1.95 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_{PRS}: Peripheral hardware clock frequency

Figure 7-6. Format of Timer Clock Selection Register 51 (TCL51)

Address: FF8CH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
TCL51	0	0	0	0	0	TCL512	TCL511	TCL510

TCL512	TCL511	TCL510	Count clock selection		
				f_{PRS} = 12 MHz	f_{PRS} = 16 MHz
0	0	0	TI51 pin falling edge		
0	0	1	TI51 pin rising edge		
0	1	0	f_{PRS}	12 MHz	16 MHz
0	1	1	$f_{PRS}/2$	6 MHz	8 MHz
1	0	0	$f_{PRS}/2^4$	750 kHz	1 MHz
1	0	1	$f_{PRS}/2^6$	187.5 kHz	250 kHz
1	1	0	$f_{PRS}/2^8$	46.88 kHz	62.5 kHz
1	1	1	$f_{PRS}/2^{12}$	2.93 kHz	3.91 kHz

Cautions 1. When rewriting TCL51 to other data, stop the timer operation beforehand.

2. Be sure to clear bits 3 to 7 to 0.

Remark f_{PRS}: Peripheral hardware clock frequency

(2) 8-bit timer mode control register 5n (TMC5n)

TMC5n is a register that performs the following five types of settings.

<1> 8-bit timer counter 5n (TM5n) count operation control
<2> 8-bit timer counter 5n (TM5n) operating mode selection
<3> Timer output F/F (flip flop) status setting
<4> Active level selection in timer F/F control or PWM (free-running) mode.
<5> Timer output control

TMC5n can be set by a 1-bit or 8-bit memory manipulation instruction.
Reset signal generation sets this register to 00H.

Remark n = 0, 1

Figure 7-7. Format of 8-Bit Timer Mode Control Register 50 (TMC50)

Address: FF6BH After reset: 00H R/W[Note]

Symbol	<7>	6	5	4	<3>	<2>	1	<0>
TMC50	TCE50	TMC506	0	0	LVS50	LVR50	TMC501	TOE50

TCE50	TM50 count operation control
0	After clearing to 0, count operation disabled (counter stopped)
1	Count operation start

TMC506	TM50 operating mode selection
0	Mode in which clear & start occurs on a match between TM50 and CR50
1	PWM (free-running) mode

LVS50	LVR50	Timer output F/F status setting
0	0	No change
0	1	Timer output F/F clear (0) (default output value of TO50 pin: low level)
1	0	Timer output F/F set (1) (default output value of TO50 pin: 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 (TM50 output is low level)
1	Output enabled

Note Bits 2 and 3 are write-only.

(**Cautions** and **Remarks** are listed on the next page.)

Figure 7-8. Format of 8-Bit Timer Mode Control Register 51 (TMC51)

Address: FF43H After reset: 00H R/W[Note]

Symbol	<7>	6	5	4	<3>	<2>	1	<0>
TMC51	TCE51	TMC516	0	0	LVS51	LVR51	TMC511	TOE51

TCE51	TM51 count operation control
0	After clearing to 0, count operation disabled (counter stopped)
1	Count operation start

TMC516	TM51 operating mode selection
0	Mode in which clear & start occurs on a match between TM51 and CR51
1	PWM (free-running) mode

LVS51	LVR51	Timer output F/F status setting
0	0	No change
0	1	Timer output F/F clear (0) (default output value of TO51 pin: low)
1	0	Timer output F/F set (1) (default output value of TO51 pin: high)
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 (TM51 output is low level)
1	Output enabled

Note Bits 2 and 3 are write-only.

Cautions 1. The settings of LVS5n and LVR5n are valid in other than PWM mode.

2. 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

3. Stop operation before rewriting TMC5n6.

4. When 8-bit timer H1 is used in the carrier generator mode, set TMC516 to 0.

Remarks 1. In PWM mode, PWM output is made inactive by clearing TCE5n to 0.

2. If LVS5n and LVR5n are read, the value is 0.

3. The values of the TMC5n6, LVS5n, LVR5n, TMC5n1, and TOE5n bits are reflected at the TO5n pin regardless of the value of TCE5n.

4. n = 0, 1

(3) Port mode registers 1 and 3 (PM1, PM3)

These registers set port 1 and 3 input/output in 1-bit units.

When using the P17/TO50/TI50 and P33/TO51/TI51 pins for timer output, clear PM17 and PM33 and the output latches of P17 and P33 to 0.

When using the P17/TO50/TI50 and P33/TO51/TI51 pins for timer input, set PM17 and PM33 to 1. The output latches of P17 and P33 at this time may be 0 or 1.

PM1 and PM3 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

Figure 7-9. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10

PM1n	P1n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Figure 7-10. Format of Port Mode Register 3 (PM3)

Address: FF23H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM3	1	1	1	1	PM33	PM32	PM31	PM30

PM3n	P3n pin I/O mode selection (n = 0 to 3)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

7.4 Operations of 8-Bit Timer/Event Counters 50 and 51

7.4.1 Operation as interval timer

8-bit timer/event counter 5n operates as an interval timer that generates interrupt requests repeatedly at intervals of the count value preset to 8-bit timer compare register 5n (CR5n).

When the count value of 8-bit timer counter 5n (TM5n) matches the value set to CR5n, counting continues with the TM5n value cleared to 0 and an interrupt request signal (INTTM5n) is generated.

The count clock of TM5n can be selected with bits 0 to 2 (TCL5n0 to TCL5n2) of timer clock selection register 5n (TCL5n).

Setting

<1> Set the registers.
- TCL5n: Select the count clock.
- CR5n: Compare value
- TMC5n: Stop the count operation, select the mode in which clear & start occurs on a match of TM5n and CR5n.
  (TMC5n = 0000×××0B × = 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.

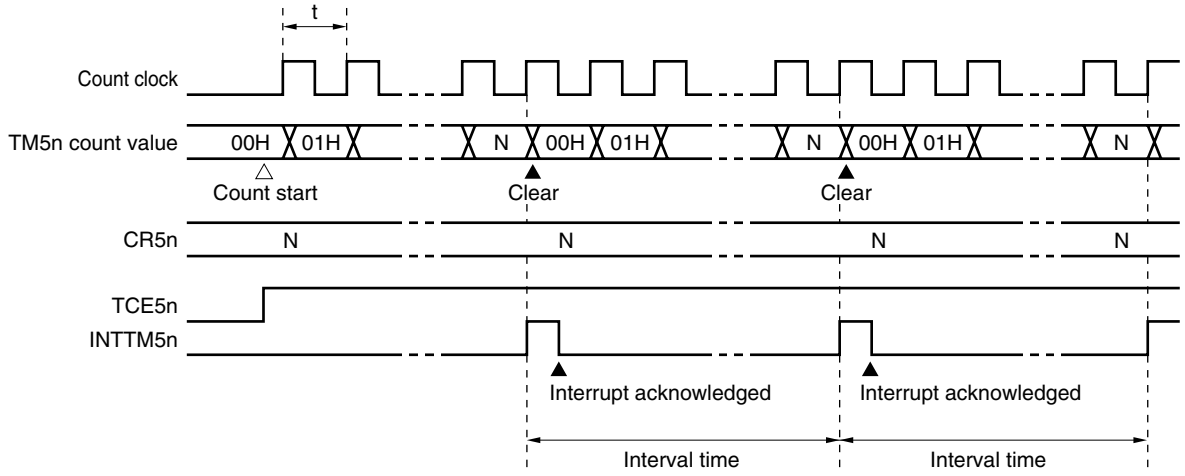
Caution Do not write other values to CR5n during operation.

Remarks 1. For how to enable the INTTM5n signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

2. n = 0, 1

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

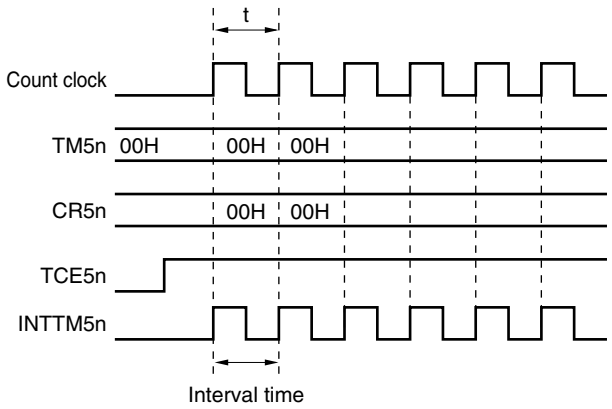
(a) Basic operation



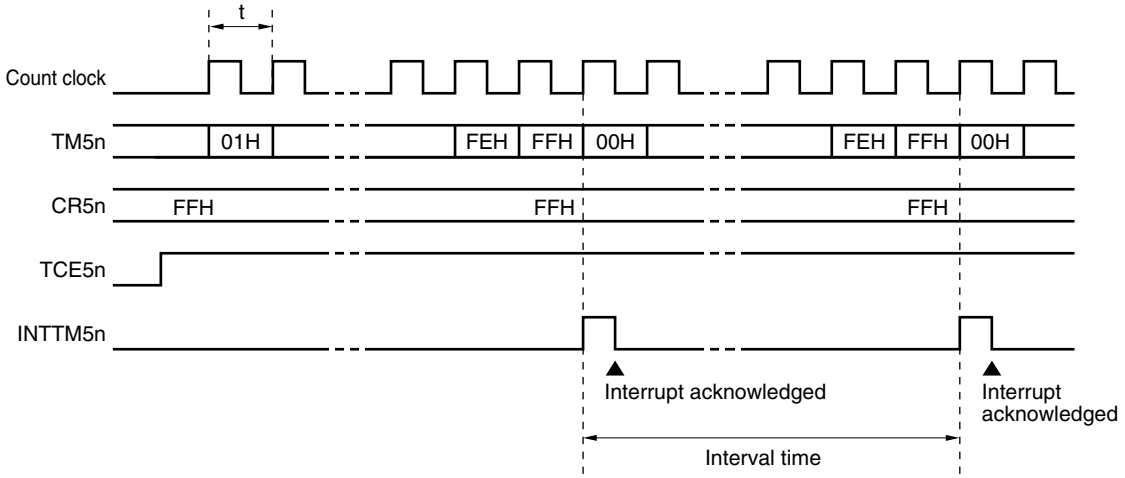
Remark Interval time = (N + 1) × t
N = 01H to FFH
n = 0, 1

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

(b) When CR5n = 00H



(c) When CR5n = FFH



Remark n = 0, 1

7.4.2 Operation as external event counter

The external event counter counts the number of external clock pulses to be input to the TI5n pin by 8-bit timer counter 5n (TM5n).

TM5n is incremented each time the valid edge specified by timer clock selection register 5n (TCL5n) is input. Either the rising or falling edge can be selected.

When the TM5n count value matches the value of 8-bit timer compare register 5n (CR5n), TM5n is cleared to 0 and an interrupt request signal (INTTM5n) is generated.

Whenever the TM5n value matches the value of CR5n, INTTM5n is generated.

Setting

<1> Set each register.

- Set the port mode register (PM17 or PM33)[Note] to 1.
- TCL5n: Select TI5n pin input edge.
  TI5n pin falling edge → TCL5n = 00H
  TI5n pin rising edge → TCL5n = 01H
- CR5n: Compare value
- TMC5n: Stop the count operation, select the mode in which clear & start occurs on match of TM5n and CR5n, disable the timer F/F inversion operation, disable timer output.
  (TMC5n = 0000××00B × = Don't care)

<2> When TCE5n = 1 is set, the number of pulses input from the TI5n pin is counted.

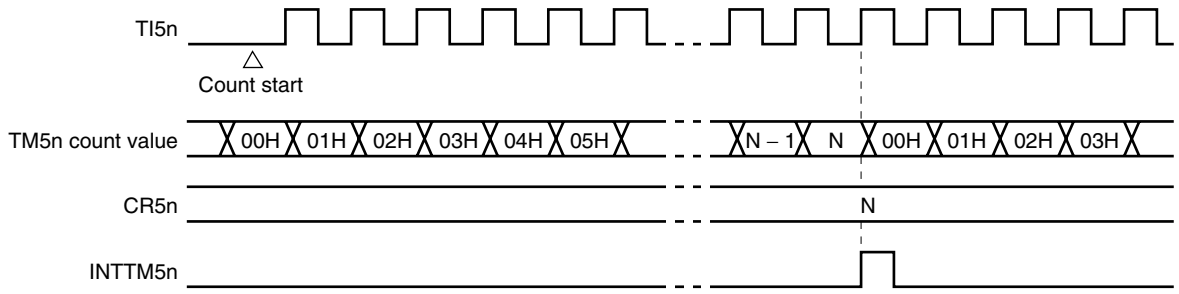
<3> When the values of TM5n and CR5n match, INTTM5n is generated (TM5n is cleared to 00H).

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

Note 8-bit timer/event counter 50: PM17
8-bit timer/event counter 51: PM33

Remark For how to enable the INTTM5n signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

Figure 7-12. External Event Counter Operation Timing (with Rising Edge Specified)



Remark N = 00H to FFH
n = 0, 1

7.4.3 Square-wave output operation

A square wave with any selected frequency is output at intervals determined by the value preset to 8-bit timer compare register 5n (CR5n).

The TO5n pin output status is inverted at intervals determined by the count value preset to CR5n by setting bit 0 (TOE5n) of 8-bit timer mode control register 5n (TMC5n) to 1. This enables a square wave with any selected frequency to be output (duty = 50%).

Setting

<1> Set each register.

- Clear the port output latch (P17 or P33)[Note] and port mode register (PM17 or PM33)[Note] to 0.
- TCL5n: Select the count clock.
- CR5n: Compare value
- TMC5n: Stop the count operation, select the mode in which clear & start occurs on a match of TM5n and CR5n.

LVS5n	LVR5n	Timer Output F/F Status Setting
1	0	Timer output F/F clear (0) (default output value of TO5n pin: low level)
0	1	Timer output F/F set (1) (default output value of TO5n pin: high level)

Timer output enabled
(TMC5n = 00001011B or 00000111B)

<2> After TCE5n = 1 is set, the count operation starts.

<3> The timer output F/F is inverted by a match of TM5n and CR5n. After INTTM5n is generated, TM5n is cleared to 00H.

<4> After these settings, the timer output F/F is inverted at the same interval and a square wave is output from TO5n. The frequency is as follows.

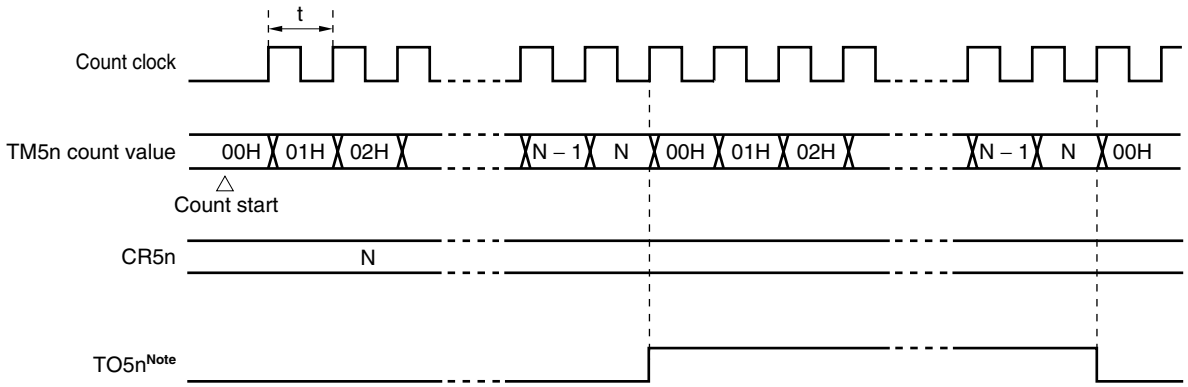
- Frequency = 1/2t (N + 1)
  (N: 00H to FFH)

Note 8-bit timer/event counter 50: P17, PM17
8-bit timer/event counter 51: P33, PM33

Caution Do not write other values to CR5n during operation.

Remarks 1. For how to enable the INTTM5n signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.
2. n = 0, 1

Figure 7-13. Square-Wave Output Operation Timing



Note The initial value of TO5n output can be set by bits 2 and 3 (LVR5n, LVS5n) of 8-bit timer mode control register 5n (TMC5n).

7.4.4 PWM output operation

8-bit timer/event counter 5n operates as a PWM output when bit 6 (TMC5n6) of 8-bit timer mode control register 5n (TMC5n) is set to 1.

The duty pulse determined by the value set to 8-bit timer compare register 5n (CR5n) is output from TO5n.

Set the active level width of the PWM pulse to CR5n; the active level can be selected with bit 1 (TMC5n1) of TMC5n.

The count clock can be selected with bits 0 to 2 (TCL5n0 to TCL5n2) of timer clock selection register 5n (TCL5n).

PWM output can be enabled/disabled with bit 0 (TOE5n) of TMC5n.

Caution In PWM mode, make the CR5n rewrite period 3 count clocks of the count clock (clock selected by TCL5n) or more.

Remark n = 0, 1

(1) PWM output basic operation

Setting

<1> Set each register.

- Clear the port output latch (P17 or P33)[Note] and port mode register (PM17 or PM33)[Note] to 0.
- TCL5n: Select the count clock.
- CR5n: Compare value
- TMC5n: Stop the count operation, select PWM mode.
  The timer output F/F is not changed.

TMC5n1	Active Level Selection
0	Active-high
1	Active-low

Timer output enabled
(TMC5n = 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: P17, PM17
8-bit timer/event counter 51: P33, PM33

PWM output operation

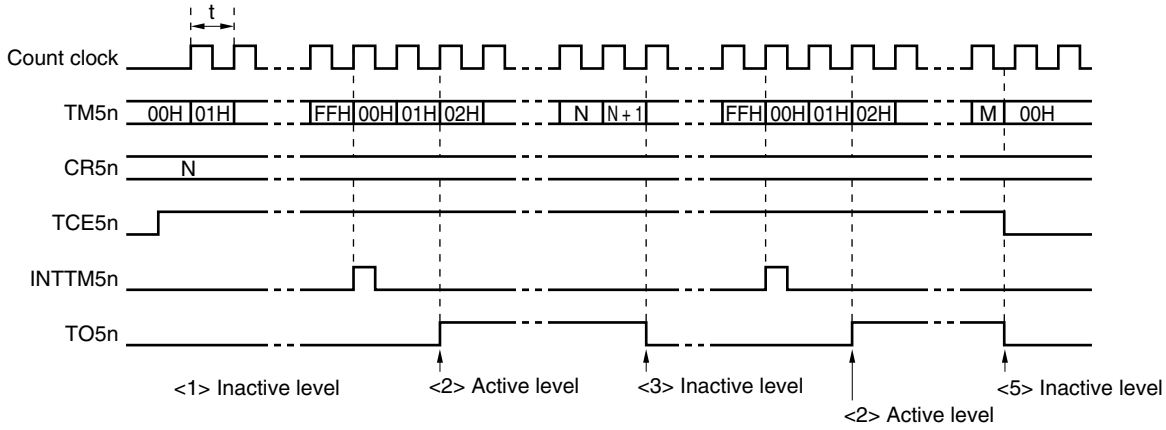
<1> PWM output (output from TO5n) outputs an inactive level until an overflow occurs.
<2> When an overflow occurs, the active level is output. The active level is output until CR5n matches the count value of 8-bit timer counter 5n (TM5n).
<3> After the CR5n matches the count value, the inactive level 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 = 2^8t
- Active-level width = Nt
- Duty = N/2^8
  (N = 00H to FFH)

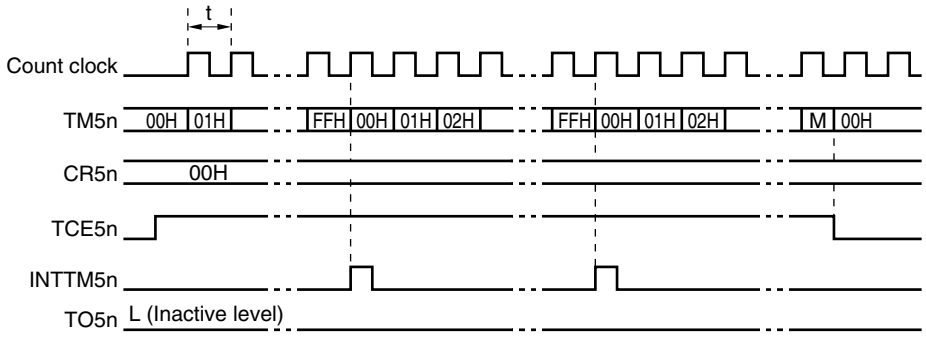
Remark n = 0, 1

Figure 7-14. PWM Output Operation Timing

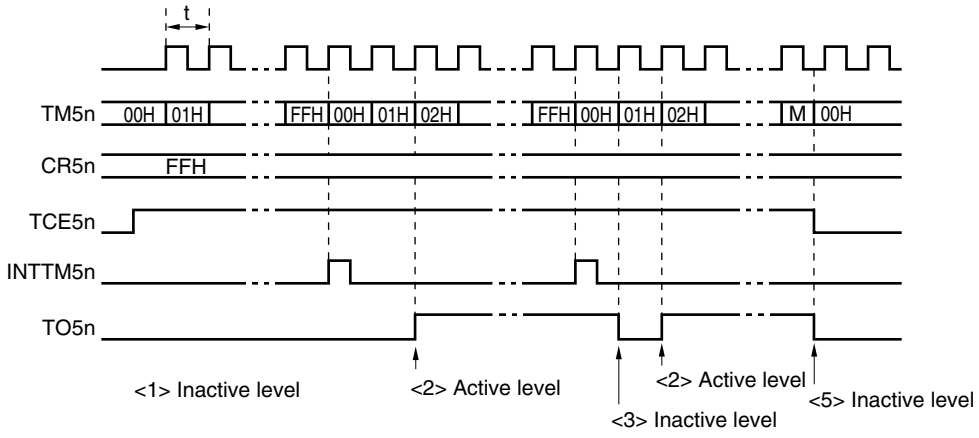
(a) Basic operation (active level = H)



(b) CR5n = 00H



(c) CR5n = FFH



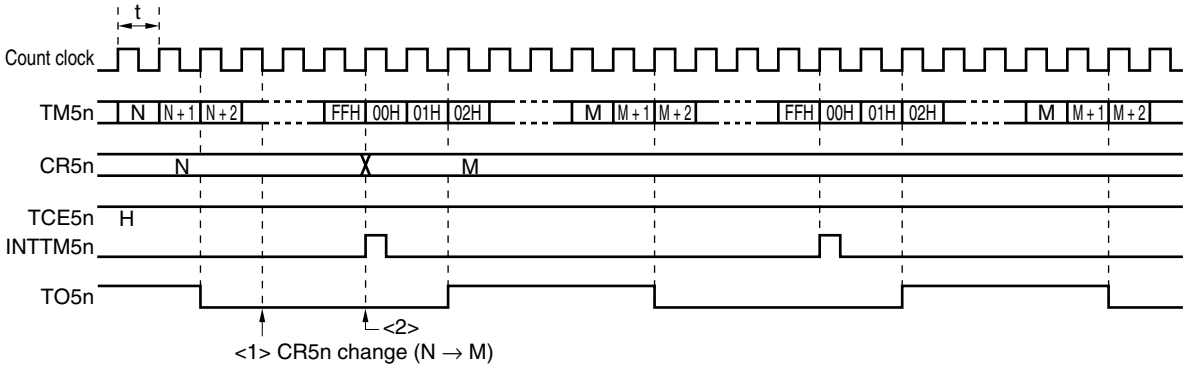
Remarks 1. <1> to <3> and <5> in Figure 7-14 (a) correspond to <1> to <3> and <5> in PWM output operation in **7.4.4 (1) PWM output basic operation**.

2. n = 0, 1

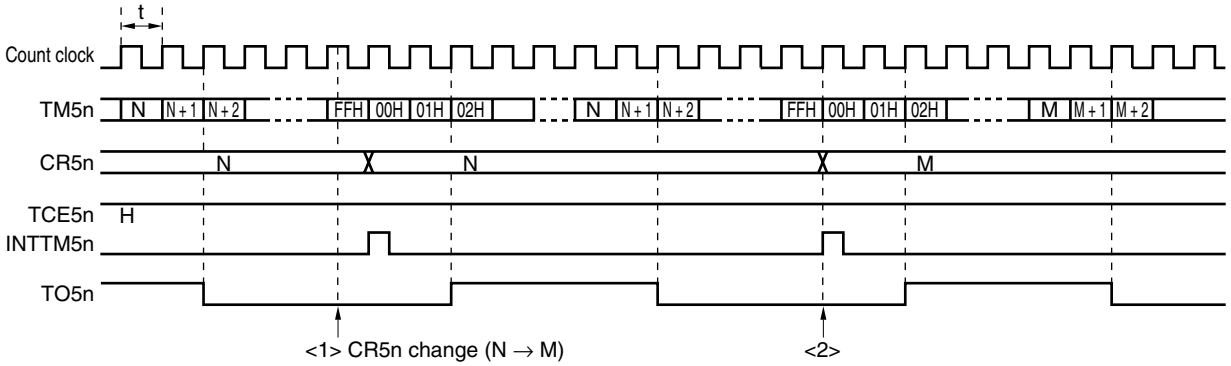
(2) Operation with CR5n changed

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.



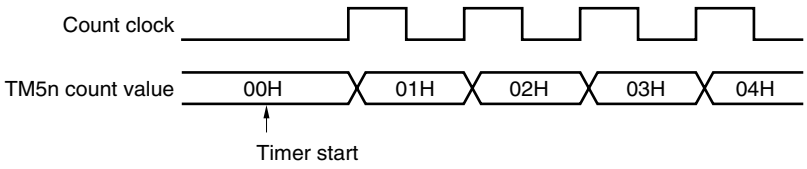
Caution When reading from CR5n between <1> and <2> in Figure 7-15, the value read differs from the actual value (read value: M, actual value of CR5n: N).

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 TIMER H1

8.1 Functions of 8-Bit Timer H1

8-bit timer H1 has the following functions.

- Interval timer
- Square-wave output
- PWM output
- Carrier generator

8.2 Configuration of 8-Bit Timer H1

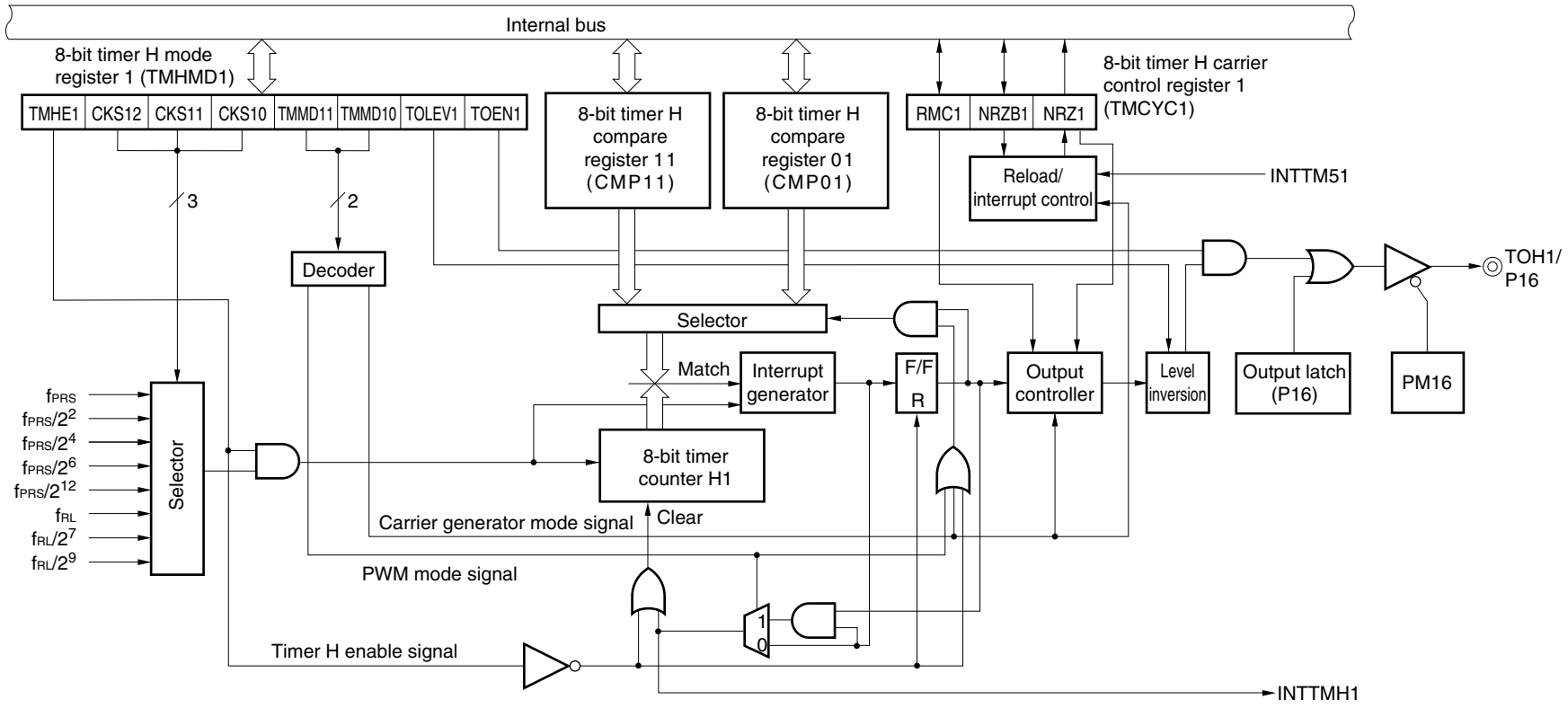
8-bit timer H1 includes the following hardware.

Table 8-1. Configuration of 8-Bit Timer H1

Item	Configuration
Timer register	8-bit timer counter H1
Registers	8-bit timer H compare register 01 (CMP01) 8-bit timer H compare register 11 (CMP11)
Timer output	TOH1, output controller
Control registers	8-bit timer H mode register 1 (TMHMD1) 8-bit timer H carrier control register 1 (TMCYC1) Port mode register 1 (PM1) Port register 1 (P1)

Figure 8-1 shows the block diagram.

Figure 8-1. Block Diagram of 8-Bit Timer H1



(1) 8-bit timer H compare register 01 (CMP01)

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 CMP01 with the count value of 8-bit timer counter H1 and, when the two values match, generates an interrupt request signal (INTTMH1) and inverts the output level of TOH1.

Rewrite the value of CMP01 while the timer is stopped (TMHE1 = 0).

A reset signal generation sets this register to 00H.

Figure 8-2. Format of 8-Bit Timer H Compare Register 01 (CMP01)

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

Symbol	7	6	5	4	3	2	1	0
CMP01								

Caution CMP01 cannot be rewritten during timer count operation.

(2) 8-bit timer H compare register 11 (CMP11)

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 CMP11 with the count value of 8-bit timer counter H1 and, when the two values match, inverts the output level of TOH1. No interrupt request signal is generated.

In the carrier generator mode, the CMP11 register always compares the value set to CMP11 with the count value of 8-bit timer counter H1 and, when the two values match, generates an interrupt request signal (INTTMH1). At the same time, the count value is cleared.

CMP11 can be rewritten during timer count operation.

If the value of CMP11 is rewritten while the timer is operating, the new value is latched and transferred to CMP11 when the count value of the timer matches the old value of CMP11, and then the value of CMP11 is changed to the new value. If matching of the count value and the CMP11 value and writing a value to CMP11 conflict, the value of CMP11 is not changed.

A reset signal generation sets this register to 00H.

Figure 8-3. Format of 8-Bit Timer H Compare Register 11 (CMP11)

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

Symbol	7	6	5	4	3	2	1	0
CMP11								

Caution In the PWM output mode and carrier generator mode, be sure to set 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).

8.3 Registers Controlling 8-Bit Timer H1

The following four registers are used to control 8-bit timer H1.

- 8-bit timer H mode register 1 (TMHMD1)
- 8-bit timer H carrier control register 1 (TMCYC1)
- Port mode register 1 (PM1)
- Port register 1 (P1)

(1) 8-bit timer H mode register 1 (TMHMD1)

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 sets this register to 00H.

Figure 8-4. Format of 8-Bit Timer H Mode Register 1 (TMHMD1)

Address: FF6CH After reset: 00H R/W

	<7>	6	5	4	3	2	<1>	<0>
TMHMD1	TMHE1	CKS12	CKS11	CKS10	TMMD11	TMMD10	TOLEV1	TOEN1

TMHE1	Timer operation enable
0	Stops timer count operation (counter is cleared to 0)
1	Enables timer count operation (count operation started by inputting clock)

CKS12	CKS11	CKS10	Count clock selection		
				f_{PRS} = 12 MHz	f_{PRS} = 16 MHz
0	0	0	f_{PRS}	12 MHz	16 MHz
0	0	1	$f_{PRS}/2^2$	3 MHz	4 MHz
0	1	0	$f_{PRS}/2^4$	750 kHz	1 MHz
0	1	1	$f_{PRS}/2^6$	187.5 kHz	250 kHz
1	0	0	$f_{PRS}/2^{12}$	2.93 kHz	3.91 kHz
1	0	1	$f_{RL}/2^7$	1.88 kHz (TYP.)	
1	1	0	$f_{RL}/2^9$	0.47 kHz (TYP.)	
1	1	1	f_{RL}	240 kHz (TYP.)	

TMMD11	TMMD10	Timer operation mode
0	0	Interval timer mode
0	1	Carrier generator mode
1	0	PWM output mode
1	1	Setting prohibited

TOLEV1	Timer output level control (in default mode)
0	Low level
1	High level

TOEN1	Timer output control
0	Disables output
1	Enables output

Cautions 1. When TMHE1 = 1, setting the other bits of TMHMD1 is prohibited.

2. In the PWM output mode and carrier generator mode, be sure to set 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.

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 sets this register to 00H.

Figure 8-5. Format of 8-Bit Timer H Carrier Control Register 1 (TMCYC1)

Address: FF6DH After reset: 00H R/W[Note]

	7	6	5	4	3	2	1	<0>
TMCYC1	0	0	0	0	0	RMC1	NRZB1	NRZ1

RMC1	NRZB1	Remote control output
0	0	Low-level output
0	1	High-level output
1	0	Low-level output
1	1	Carrier pulse output

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.

(3) Port mode register 1 (PM1)

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

When using the P16/TOH1 pin for timer output, clear PM16 and the output latches of P16 to 0.

PM1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Figure 8-6. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10

PM1n	P1n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

8.4 Operation of 8-Bit Timer H1

8.4.1 Operation as interval timer/square-wave output

When 8-bit timer counter H1 and compare register 01 (CMP01) match, an interrupt request signal (INTTMH1) is generated and 8-bit timer counter H1 is cleared to 00H.

Compare register 11 (CMP11) is not used in interval timer mode. Since a match of 8-bit timer counter H1 and the CMP11 register is not detected even if the CMP11 register is set, timer output is not affected.

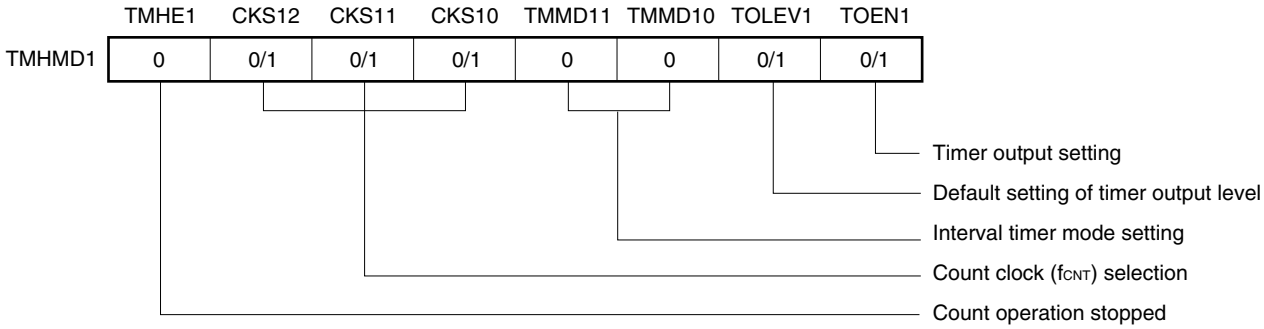
By setting bit 0 (TOENn) of timer H mode register 1 (TMHMD1) to 1, a square wave of any frequency (duty = 50%) is output from TOH1.

Setting

<1> Set each register.

Figure 8-7. Register Setting During Interval Timer/Square-Wave Output Operation

(i) Setting timer H mode register 1 (TMHMD1)



(ii) CMP01 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 TMHE1 = 1.

<3> When the values of 8-bit timer counter H1 and the CMP01 register match, the INTTMH1 signal is generated and 8-bit timer counter H1 is cleared to 00H.

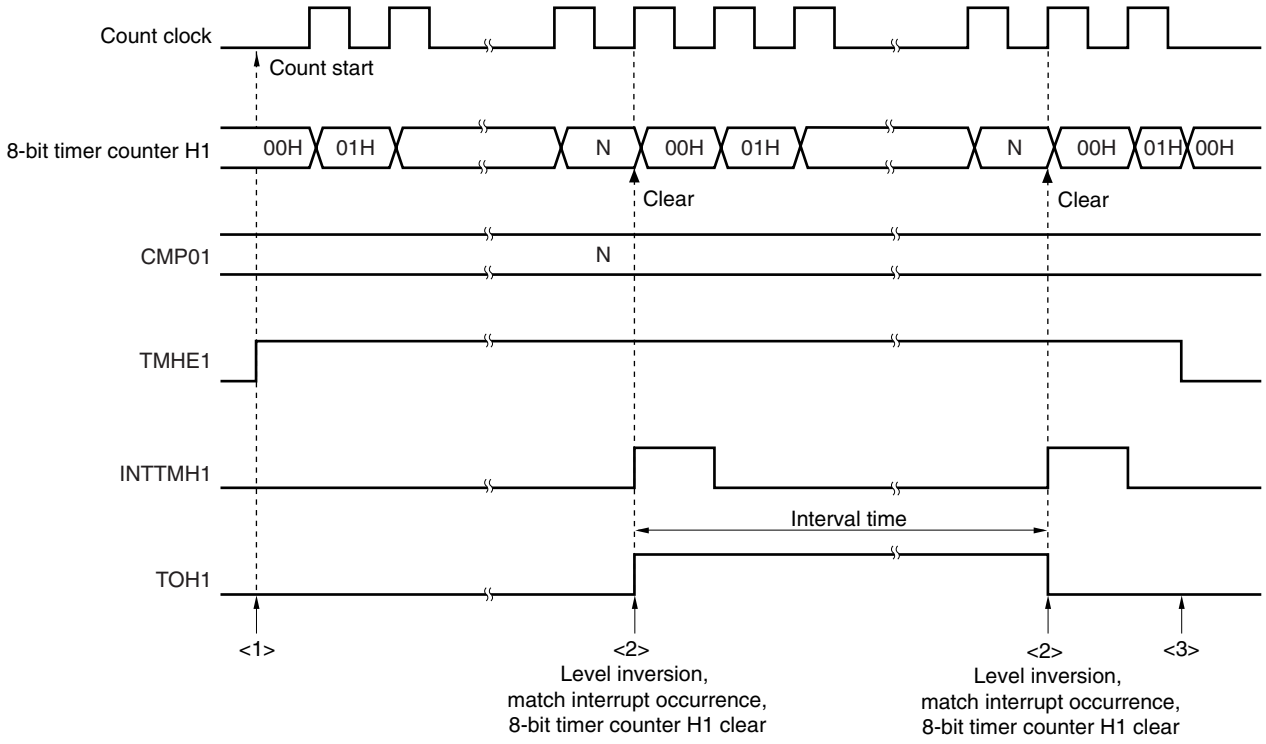
<4> Subsequently, the INTTMH1 signal is generated at the same interval. To stop the count operation, clear TMHE1 to 0.

Remarks 1. For the setting of the output pin, see **8.3 (3) Port mode register 1 (PM1)**.

2. For how to enable the INTTMH1 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

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

(a) Basic operation (Operation When 01H ≤ CMP01 ≤ FEH)



<1> The count operation is enabled by setting the TMHE1 bit to 1. The count clock starts counting no more than 1 clock after the operation is enabled.

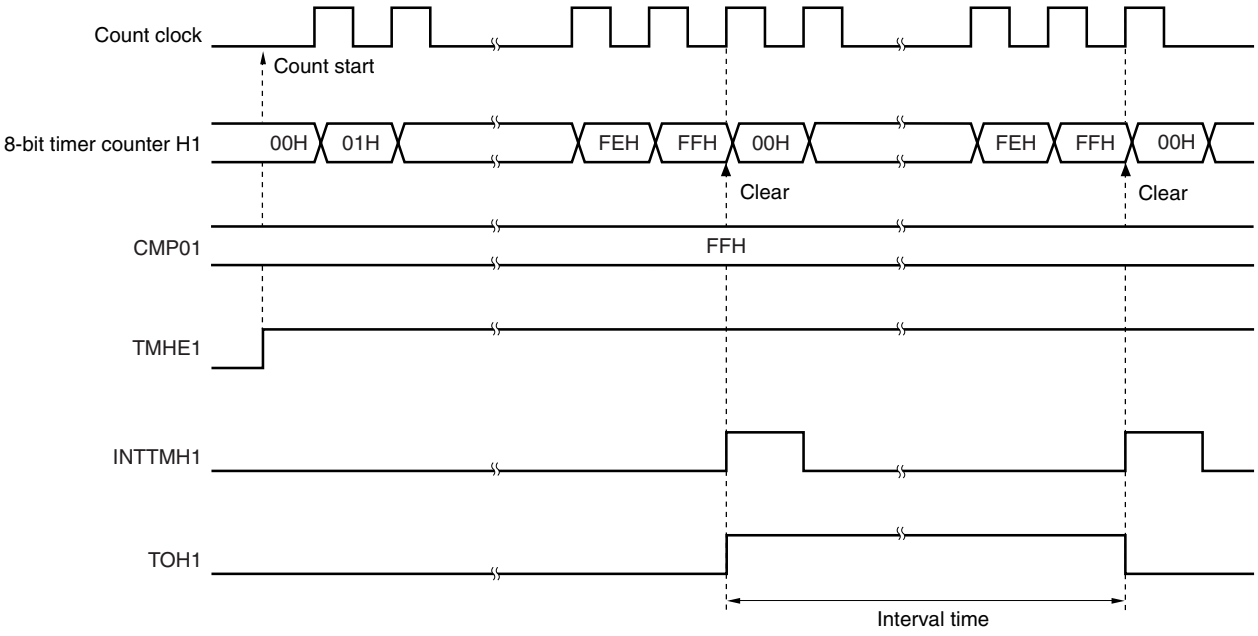
<2> When the value of 8-bit timer counter H1 matches the value of the CMP01 register, the value of the timer counter is cleared, and the level of the TOH1 output is inverted. In addition, the INTTMH1 signal is output at the rising edge of the count clock.

<3> If the TMHE1 bit is cleared to 0 while timer H is operating, the INTTMH1 signal and TOH1 output are set to the default level. If they are already at the default level before the TMHE1 bit is cleared to 0, then that level is maintained.

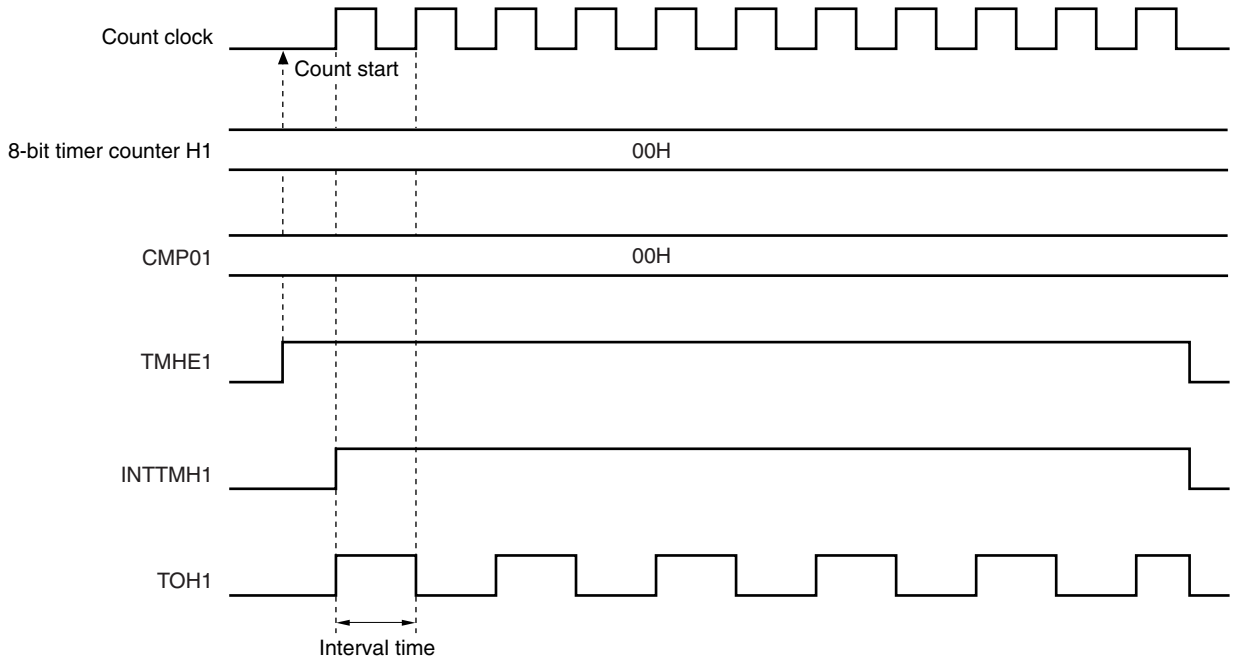
Remark 01H ≤ N ≤ FEH

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

(b) Operation when CMP01 = FFH



(c) Operation when CMP01 = 00H



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 01 (CMP01) controls the cycle of timer output (TOH1). Rewriting the CMP01 register during timer operation is prohibited.

8-bit timer compare register 11 (CMP11) controls the duty of timer output (TOH1). Rewriting the CMP11 register during timer operation is possible.

The operation in PWM output mode is as follows.

The TOH1 output level is inverted and 8-bit timer counter H1 is cleared to 0 when 8-bit timer counter H1 and the CMP01 register match after the timer count is started. The TOH1 output level is inverted when 8-bit timer counter H1 and the CMP11 register match.

Setting

<1> Set each register.

Figure 8-9. Register Setting in PWM Output Mode

(i) Setting timer H mode register 1 (TMHMD1)



(ii) Setting CMP01 register

- Compare value (N): Cycle setting

(iii) Setting CMP11 register

- Compare value (M): Duty setting

Remark 00H ≤ CMP11 (M) < CMP01 (N) ≤ FFH

<2> The count operation starts when TMHE1 = 1.

<3> The CMP01 register is the compare register that is to be compared first after counter operation is enabled. When the values of 8-bit timer counter H1 and the CMP01 register match, 8-bit timer counter H1 is cleared, an interrupt request signal (INTTMH1) is generated, and TOH1 output is inverted. At the same time, the compare register to be compared with 8-bit timer counter H1 is changed from the CMP01 register to the CMP11 register.

<4> When 8-bit timer counter H1 and the CMP11 register match, TOH1 output is inverted and the compare register to be compared with 8-bit timer counter H1 is changed from the CMP11 register to the CMP01 register. At this time, 8-bit timer counter H1 is not cleared and the INTTMH1 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 TMHE1 = 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 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 CMP11 register can be changed while the timer counter is operating. However, this takes a duration of three operating clocks (signal selected by the CKS12 to CKS10 bits of the TMHMD1 register) from when the value of the CMP11 register is changed until the value is transferred to the register.

2. 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).

3. Make sure that the CMP11 register setting value (M) and CMP01 register setting value (N) are within the following range.

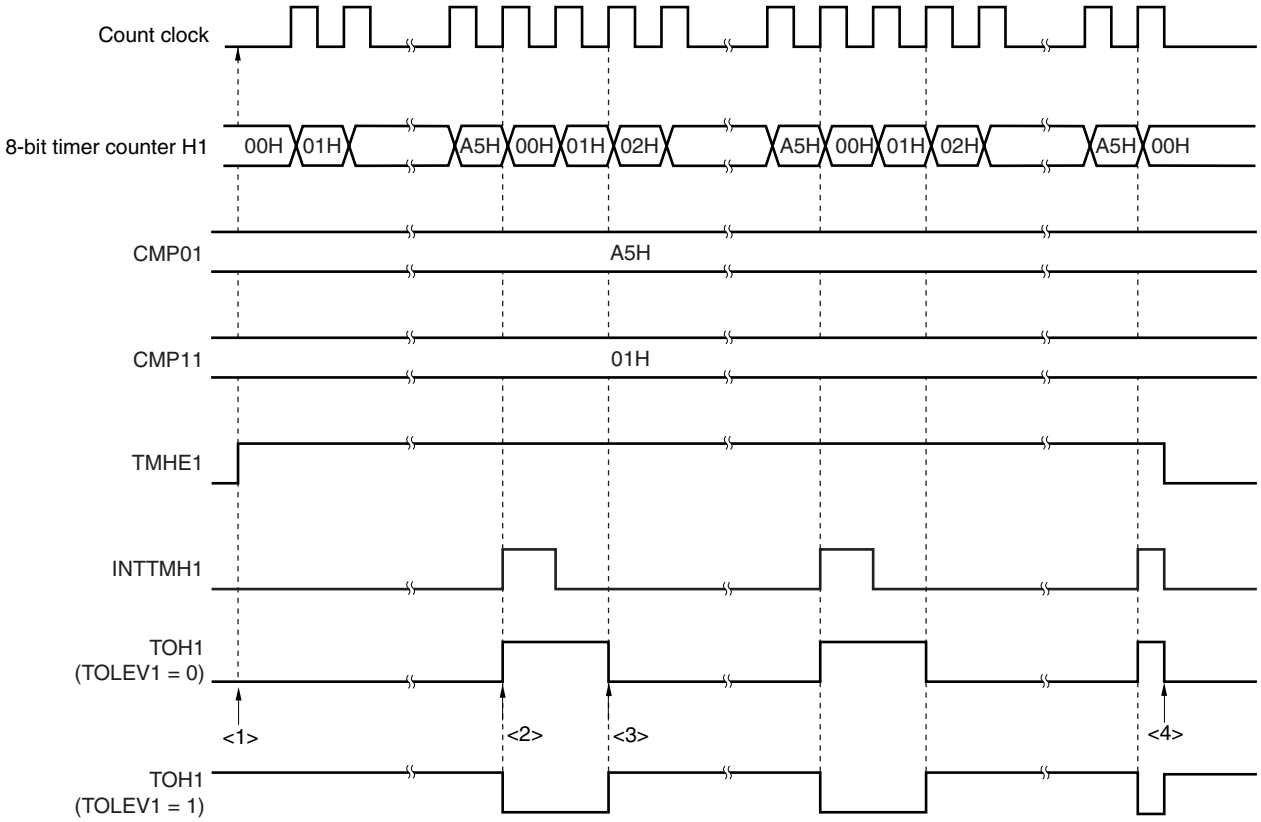
00H ≤ CMP11 (M) < CMP01 (N) ≤ FFH

Remarks 1. For the setting of the output pin, see **8.3 (3) Port mode register 1 (PM1)**.

2. For details on how to enable the INTTMH1 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

Figure 8-10. Operation Timing in PWM Output Mode (1/4)

(a) Basic operation



<1> The count operation is enabled by setting the TMHE1 bit to 1. Start 8-bit timer counter H1 by masking one count clock to count up. At this time, TOH1 output remains the default.

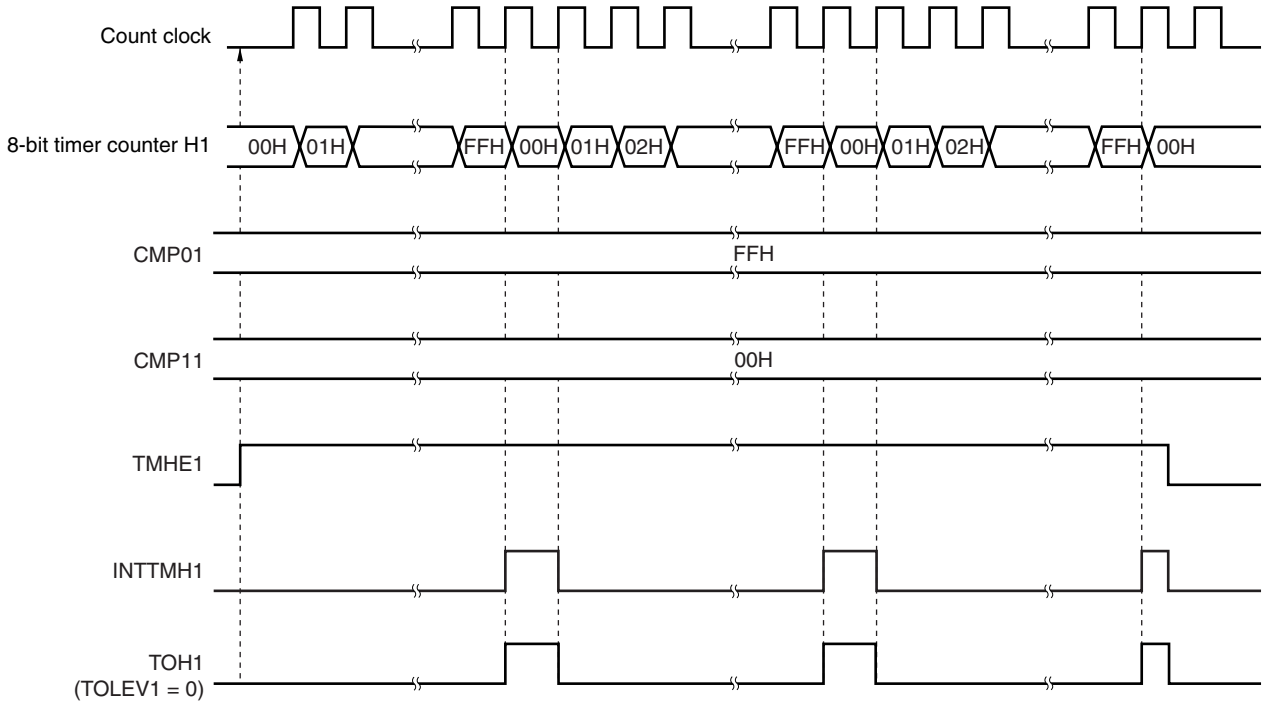
<2> When the values of 8-bit timer counter H1 and the CMP01 register match, the TOH1 output level is inverted, the value of 8-bit timer counter H1 is cleared, and the INTTMH1 signal is output.

<3> When the values of 8-bit timer counter H1 and the CMP11 register match, the TOH1 output level is inverted. At this time, the 8-bit timer counter value is not cleared and the INTTMH1 signal is not output.

<4> Clearing the TMHE1 bit to 0 during timer H1 operation sets the INTTMH1 signal and TOH1 output to the default.

Figure 8-10. Operation Timing in PWM Output Mode (2/4)

(b) Operation when CMP01 = FFH, CMP11 = 00H



(c) Operation when CMP01 = FFH, CMP11 = FEH

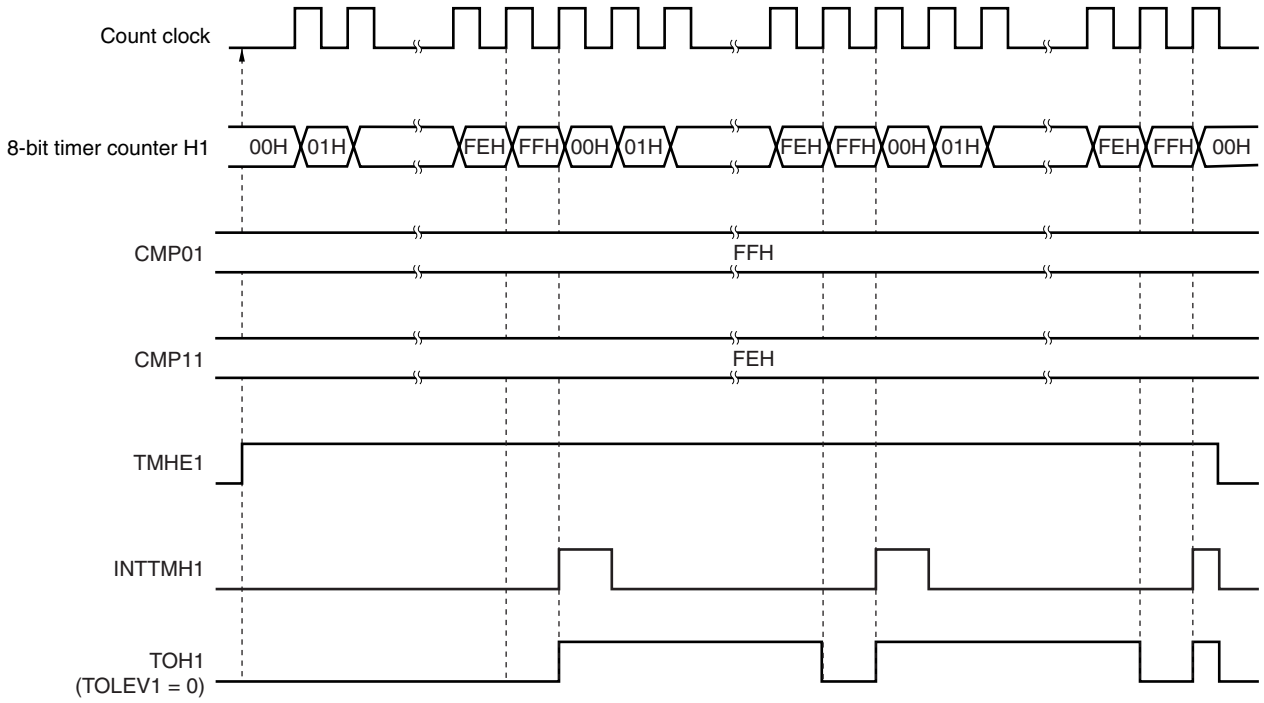


Figure 8-10. Operation Timing in PWM Output Mode (3/4)

(d) Operation when CMP01 = 01H, CMP11 = 00H

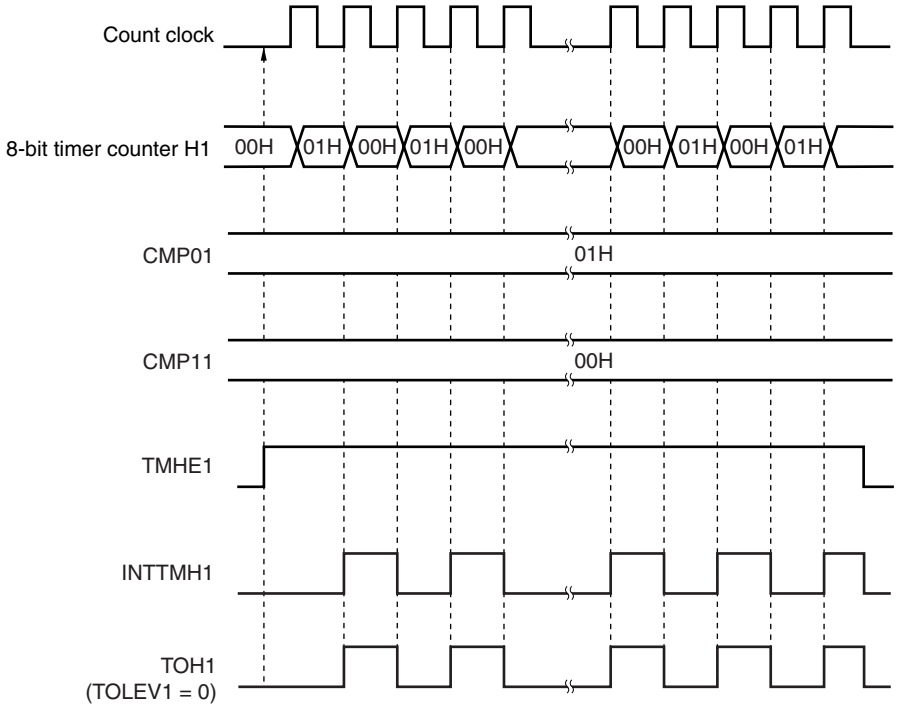
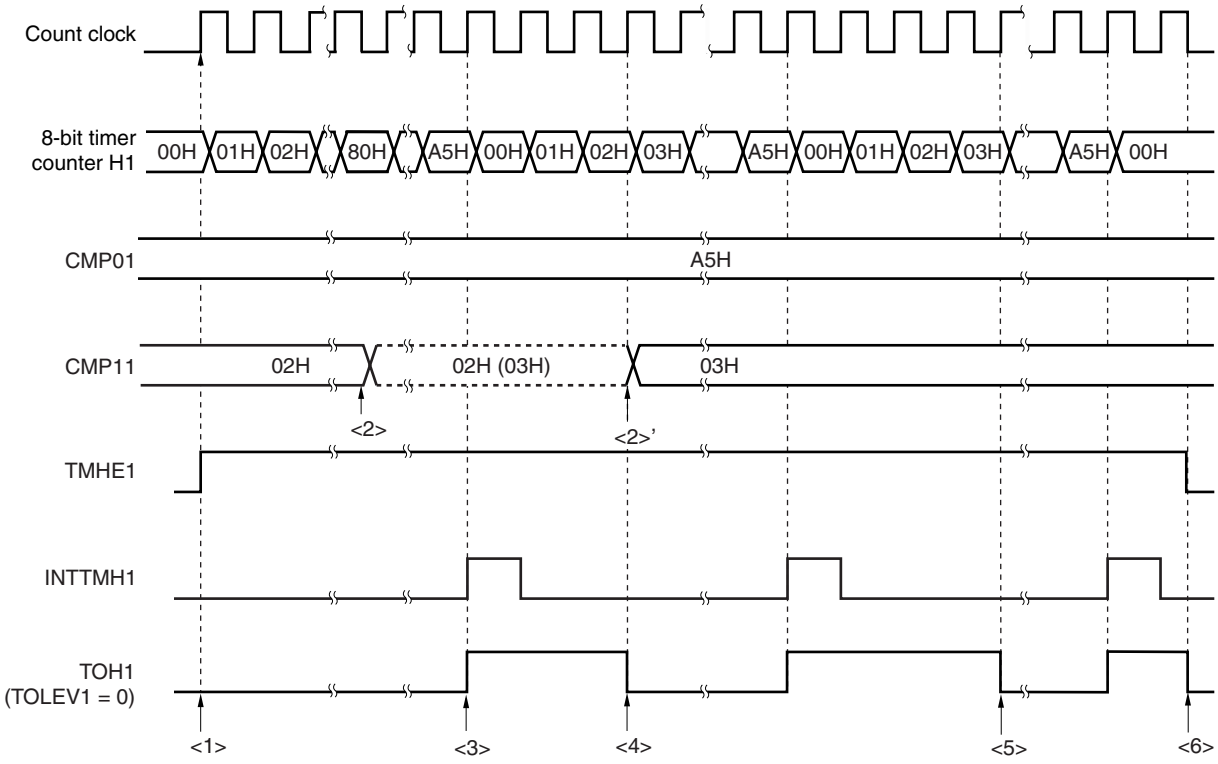


Figure 8-10. Operation Timing in PWM Output Mode (4/4)

(e) Operation by changing CMP11 (CMP11 = 02H → 03H, CMP01 = A5H)



<1> The count operation is enabled by setting TMHE1 = 1. Start 8-bit timer counter H1 by masking one count clock to count up. At this time, the TOH1 output remains default.

<2> The CMP11 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 H1 and the CMP01 register match, the value of 8-bit timer counter H1 is cleared, the TOH1 output level is inverted, and the INTTMH1 signal is output.

<4> If the CMP11 register value is changed, the value is latched and not transferred to the register. When the values of 8-bit timer counter H1 and the CMP11 register before the change match, the value is transferred to the CMP11 register and the CMP11 register value is changed (<2>').

However, three count clocks or more are required from when the CMP11 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 H1 and the CMP11 register after the change match, the TOH1 output level is inverted. 8-bit timer counter H1 is not cleared and the INTTMH1 signal is not generated.

<6> Clearing the TMHE1 bit to 0 during timer H1 operation makes the INTTMH1 signal and TOH1 output default.

8.4.3 Carrier generator operation

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 TOH1 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.

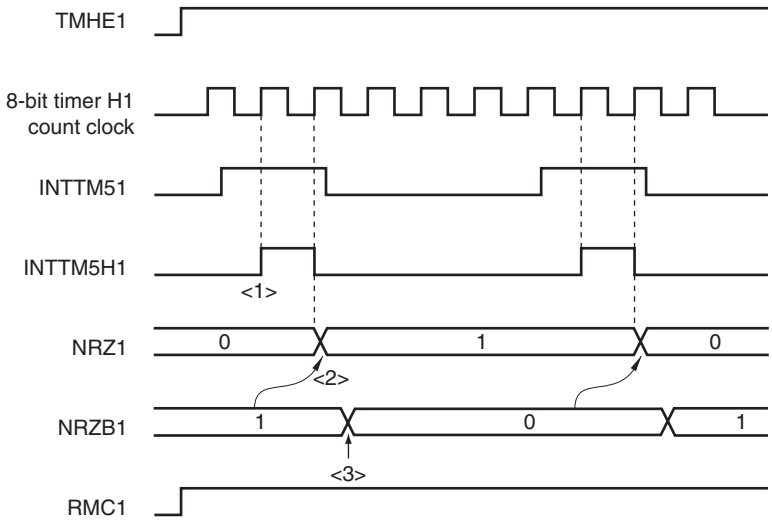
(2) Carrier output control

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 (TMCYC1). The relationship between the outputs is shown below.

RMC1 Bit	NRZB1 Bit	Output
0	0	Low-level output
0	1	High-level output
1	0	Low-level output
1	1	Carrier pulse output

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 8-11. Transfer Timing



<1> The INTTM51 signal is synchronized with the count clock of the 8-bit timer H1 and is output as the INTTM5H1 signal.

<2> The value of the NRZB1 bit is transferred to the NRZ1 bit at the second clock from the rising edge of the INTTM5H1 signal.

<3> Write the next value to the NRZB1 bit in the interrupt servicing program that has been started by the INTTM5H1 interrupt or after timing has been checked by polling the interrupt request flag. Write data to count the next time to the CR51 register.

Cautions 1. Do not rewrite the NRZB1 bit again until at least the second clock after it has been rewritten, or else the transfer from the NRZB1 bit to the NRZ1 bit is not guaranteed.

2. When 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.

Setting

<1> Set each register.

Figure 8-12. Register Setting in Carrier Generator Mode

(i) Setting 8-bit timer H mode register 1 (TMHMD1)



(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 the 8-bit timer H1 and output as the INTTM5H1 signal. The INTTM5H1 signal becomes the data transfer signal for the NRZB1 bit, and the NRZB1 bit value is transferred to the NRZ1 bit.

<8> Write the next value to the NRZB1 bit in the interrupt servicing program that has been started by the INTTM5H1 interrupt or after timing has been checked by polling the interrupt request flag. Write data to count the next time to the CR51 register.

<9> When the NRZ1 bit is high level, a carrier clock is output from the TOH1 pin.

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

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

- Carrier clock output cycle = (N + M + 2)/f_{CNT}
- Duty = High-level width/carrier clock output width = (M + 1)/(N + M + 2)

Cautions 1. Be sure to set the CMP11 register when starting the timer count operation (TMHE1 = 1) after the timer count operation was stopped (TMHE1 = 0) (be sure to set again even if setting the same value to the CMP11 register).

2. Set so that the count clock frequency of TMH1 becomes more than 6 times the count clock frequency of TM51.

3. Set the values of the CMP01 and CMP11 registers in a range of 01H to FFH.

4. The set value of the CMP11 register can be changed while the timer counter is operating. However, it takes the duration of three operating clocks (signal selected by the CKS12 to CKS10 bits of the TMHMD1 register) since the value of the CMP11 register has been changed until the value is transferred to the register.

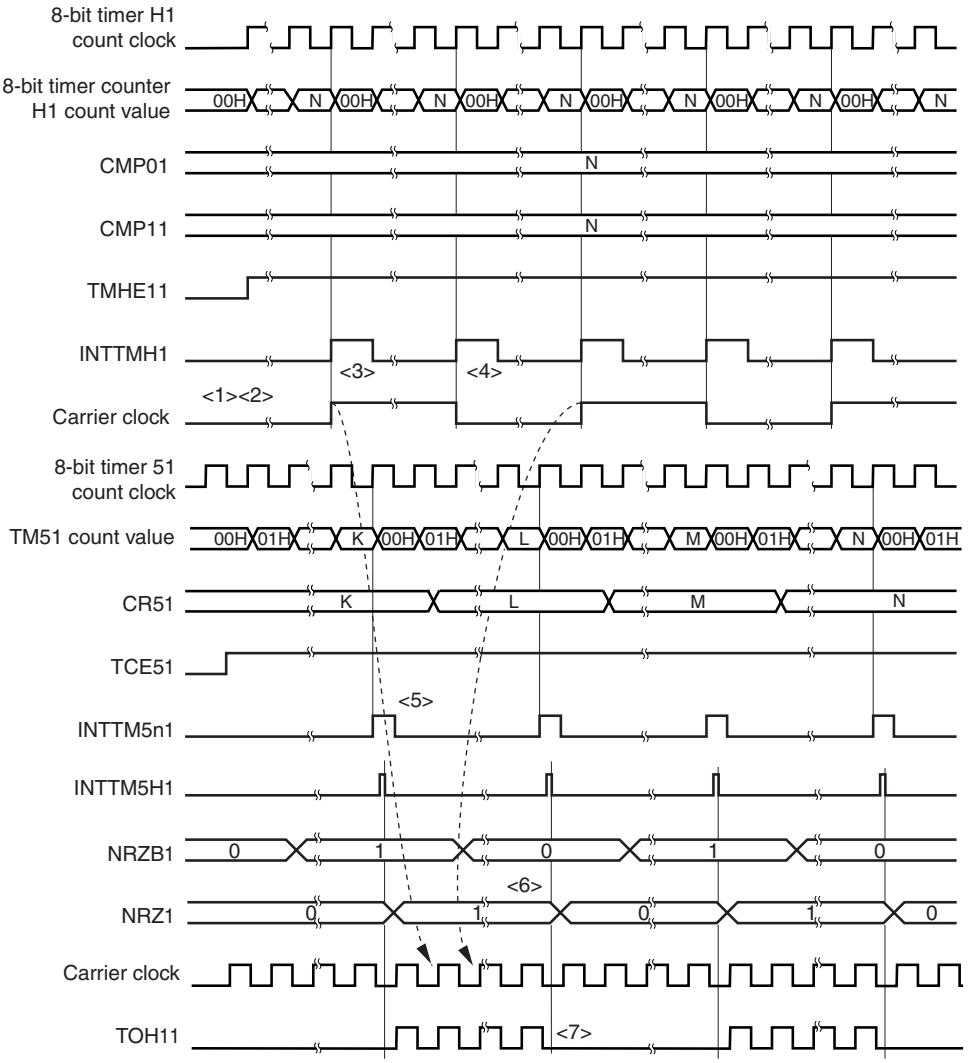
5. Be sure to set the RMC1 bit before the count operation is started.

Remarks 1. For the setting of the output pin, see **8.3 (3) Port mode register 1 (PM1)**.

2. For how to enable the INTTMH1 signal interrupt, see **CHAPTER 13 INTERRUPT FUNCTIONS**.

Figure 8-13. Carrier Generator Mode Operation Timing (1/3)

(a) Operation when CMP01 = N, CMP11 = N



<1> When TMHE1 = 0 and TCE51 = 0, the 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.

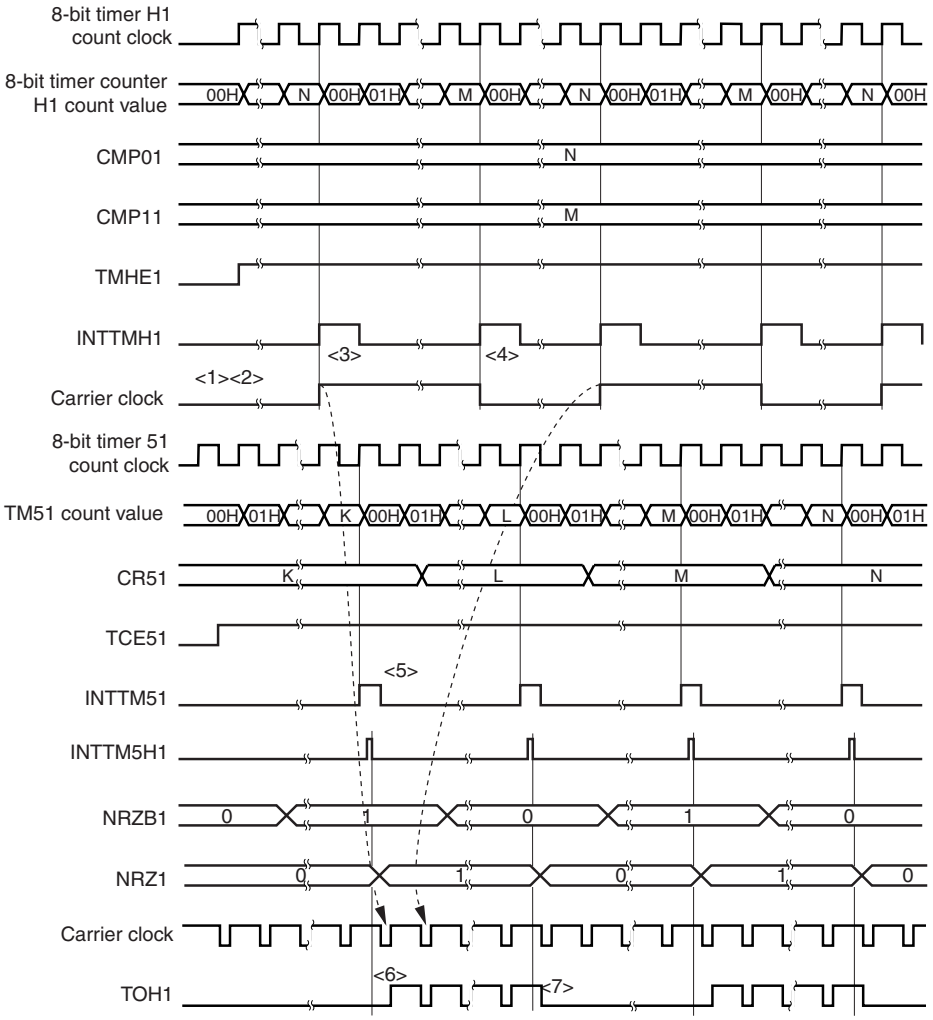
<5> When the INTTM51 signal is generated, it is synchronized with the 8-bit timer H1 count clock and is output as the INTTM5H1 signal.

<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 TOH1 output becomes low level.

Figure 8-13. Carrier Generator Mode Operation Timing (2/3)

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



<1> When TMHE1 = 0 and TCE51 = 0, the 8-bit timer counter H1 operation is stopped.

<2> When TMHE1 = 1 is set, 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.

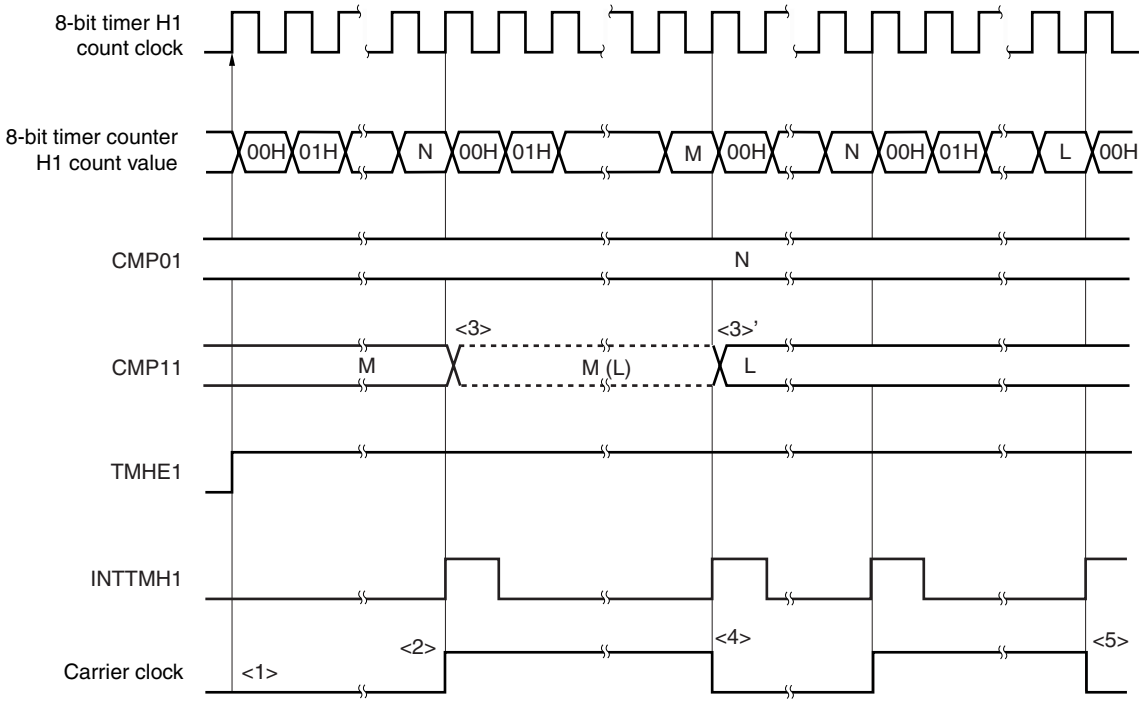
<5> When the INTTM51 signal is generated, it is synchronized with the 8-bit timer H1 count clock and is output as the INTTM5H1 signal.

<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 TOH1 output is held at the high level and is not changed to low level while the carrier clock is high level (from <6> and <7>, the high-level width of the carrier clock waveform is guaranteed).

Figure 8-13. Carrier Generator Mode Operation Timing (3/3)

(c) Operation when CMP11 is changed



<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 and IXS registers (detection of an invalid check while the CPU hangs up)
- If the CPU accesses an area that is not set by the IMS and IXS registers (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 15 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)

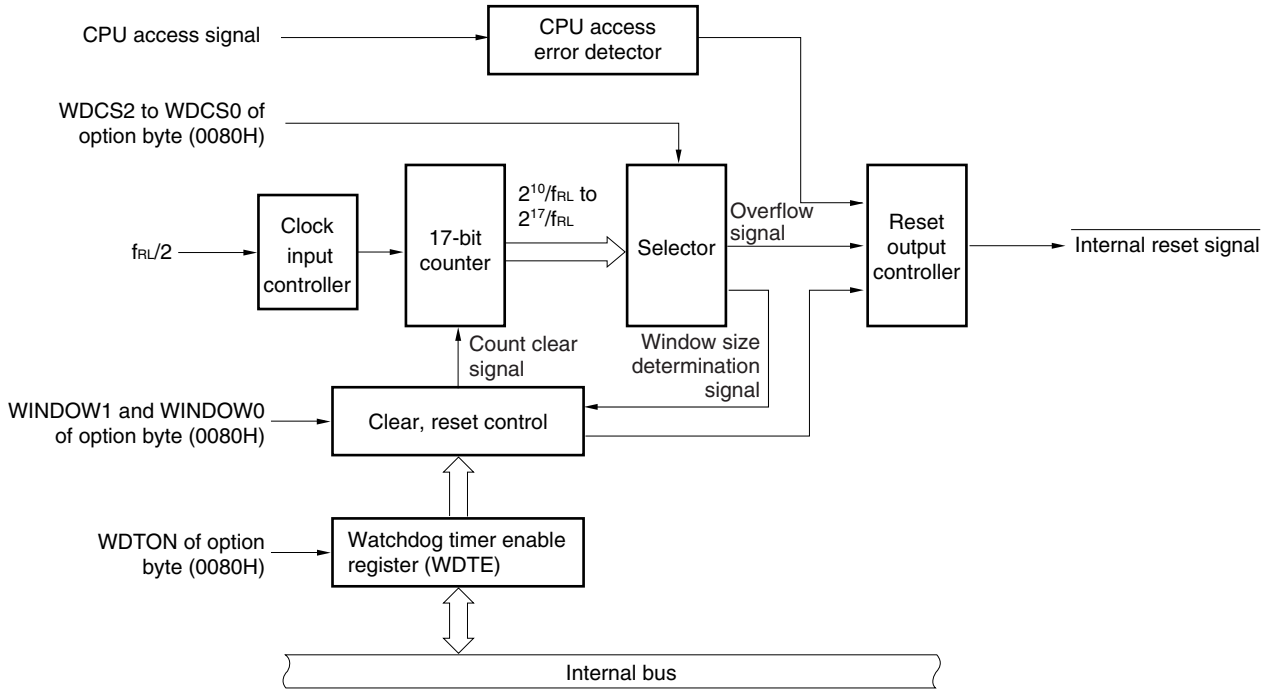
How the counter operation is controlled, overflow time, and window open period are set by the option byte.

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 18 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[Note].

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

Address: FF99H After reset: 9AH/1AH[Note] R/W

Symbol	7	6	5	4	3	2	1	0
WDTE								

Note The WDTE reset value differs depending on the WDTON setting value of the option byte (0080H). To operate watchdog timer, set WDTON to 1.

WDTON Setting Value	WDTE Reset Value
0 (watchdog timer count operation disabled)	1AH
1 (watchdog timer count operation enabled)	9AH

Cautions 1. If a value other than ACH is written to WDTE, an internal reset signal is generated. If the source clock to the watchdog timer is stopped, however, an internal reset signal is generated when the source clock to the watchdog timer resumes operation.

2. If a 1-bit memory manipulation instruction is executed for WDTE, an internal reset signal is generated. If the source clock to the watchdog timer is stopped, however, an internal reset signal is generated when the source clock to the watchdog timer resumes operation.

3. The value read from WDTE is 9AH/1AH (this differs from the written value (ACH)).

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 18**).

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 18**).
   - 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 18**).

2. After a reset release, the watchdog timer starts counting.
3. By writing “ACH” to WDTE after the watchdog timer starts counting and before the overflow time set by the option byte, the watchdog timer is cleared and starts counting again.
4. After that, write WDTE the second time or later after a reset release during the window open period. If WDTE is written during a window close period, an internal reset signal is generated.
5. If the overflow time expires without “ACH” written to WDTE, an internal reset signal is generated.
   A internal reset signal is generated in the following cases.

   - If a 1-bit manipulation instruction is executed on the watchdog timer enable register (WDTE)
   - If data other than “ACH” is written to WDTE
   - If the instruction is fetched from an area not set by the IMS and IXS registers (detection of an invalid check during a CPU program loop)
   - If the CPU accesses an area not set by the IMS and IXS registers (excluding FB00H to 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 (LIOCP) of the option byte.

	LIOCP = 0 (Internal Low-Speed Oscillator Can Be Stopped by Software)	LIOCP = 1 (Internal Low-Speed Oscillator Cannot Be Stopped)
In HALT mode	Watchdog timer operation stops.	Watchdog timer operation continues.
In STOP mode		

If LIOCP = 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 LIOCP = 0, the watchdog timer stops operating. At this time, the counter is not cleared to 0.

5. The watchdog timer does not stop during self-programming of the flash memory and EEPROM™ emulation. 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.

Table 9-3. Setting of Overflow Time of Watchdog Timer

WDCS2	WDCS1	WDCS0	Overflow Time of Watchdog Timer
0	0	0	$2^{10}/f_{RL}$ (3.88 ms)
0	0	1	$2^{11}/f_{RL}$ (7.76 ms)
0	1	0	$2^{12}/f_{RL}$ (15.52 ms)
0	1	1	$2^{13}/f_{RL}$ (31.03 ms)
1	0	0	$2^{14}/f_{RL}$ (62.06 ms)
1	0	1	$2^{15}/f_{RL}$ (124.12 ms)
1	1	0	$2^{16}/f_{RL}$ (248.24 ms)
1	1	1	$2^{17}/f_{RL}$ (496.48 ms)

Caution The watchdog timer does not stop during self-programming of the flash memory and EEPROM emulation. During processing, the interrupt acknowledge time is delayed. Set the overflow time and window size taking this delay into consideration.

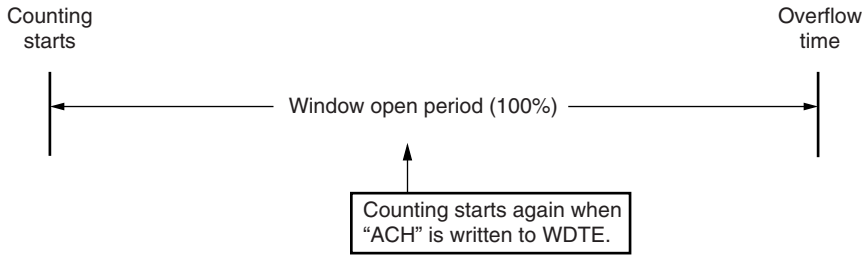
Remarks 1. f_{RL}: Internal low-speed oscillation clock frequency

2. ( ): f_{RL} = 264 kHz (MAX.)

9.4.3 Setting window open period of watchdog timer

In the μPD78F0730, the window open period of the watchdog timer is 100%. Do not set any values other than 1, 1 (default) to bits 6 and 5 (WINDOW1, WINDOW0) of the option byte.

Whenever WDTE is written to during the window open period (100%), as long as it is before the overflow time, the watchdog timer is cleared and starts counting again.



The window open period to be set is as follows.

Table 9-4. Setting Window Open Period of Watchdog Timer

WINDOW1	WINDOW0	Window Open Period of Watchdog Timer
0	0	Setting prohibited
0	1	Setting prohibited
1	0	Setting prohibited
1	1	100% (default)

Caution Only the combination of WINDOW1 and WINDOW0 = 1, 1 is valid. Other settings are prohibited.

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).
- LSB-first communication

Cautions 1. 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.

2. Set POWER6 = 1 and then set TXE6 = 1 (transmission) or RXE6 = 1 (reception) to start communication.

3. TXE6 and RXE6 are synchronized by the base clock (f_{XCLK6}) set by CKSR6. To enable transmission or reception again, set TXE6 or RXE6 to 1 at least two clocks of the base clock after TXE6 or RXE6 has been cleared to 0. If TXE6 or RXE6 is set within two clocks of the base clock, the transmission circuit or reception circuit may not be initialized.

4. Set transmit data to TXB6 at least one base clock (f_{XCLK6}) after setting TXE6 = 1.

5. 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.

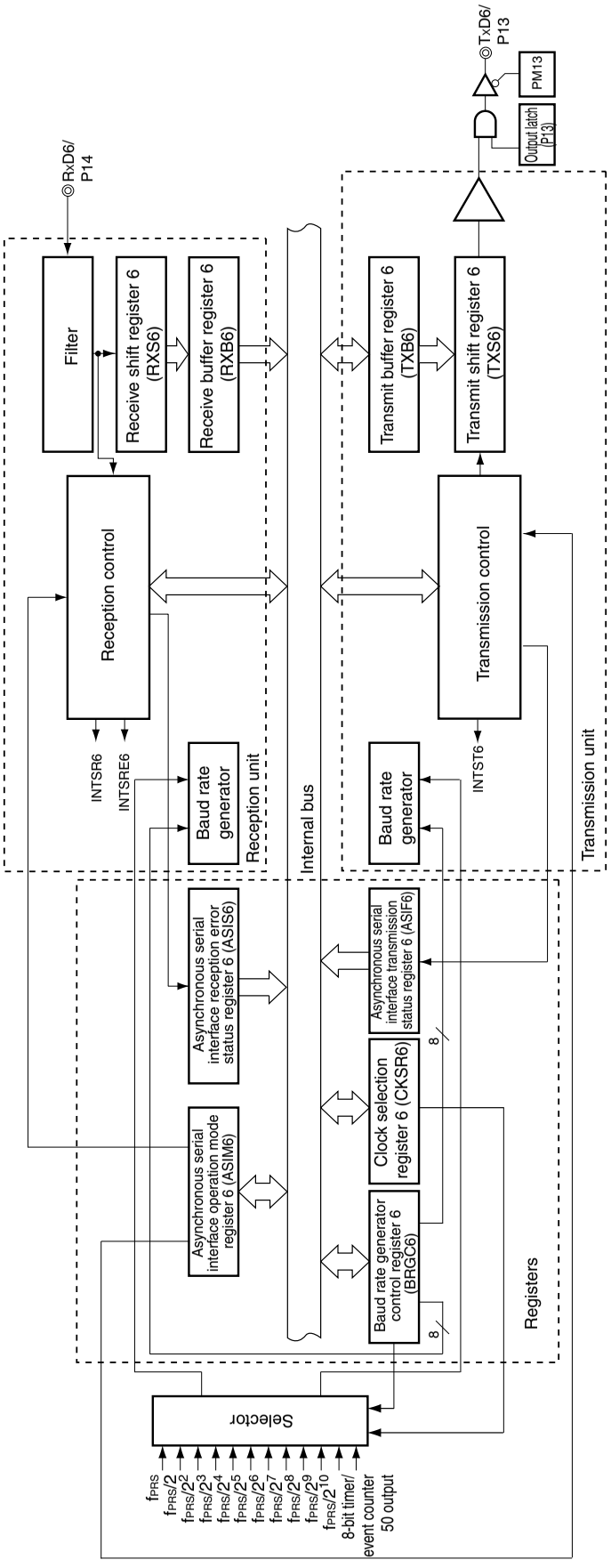
10.2 Configuration of Serial Interface UART6

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) Port mode register 1 (PM1) Port register 1 (P1)

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.

- The receive data is transferred to bits 0 to 6 of RXB6 and the MSB 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 bit 7 (POWER6) and bit 6 (TXE6) of asynchronous serial interface operation mode register 6 (ASIM6) are 1 or when bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 are 1).

3. Set transmit data to TXB6 at least one base clock (f_{XCLK6}) 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 seven 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)
- Port mode register 1 (PM1)
- Port register 1 (P1)

(1) Asynchronous serial interface operation mode register 6 (ASIM6)

This 8-bit register controls the serial communication operations of serial interface UART6.

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

Reset signal generation sets this register to 01H.

Remark ASIM6 can be refreshed (the same value is written) by software during a communication operation (when bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1 or bit 7 (POWER6) and bit 5 (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[Note 1]	Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit[Note 2].
1	Enables operation of the internal operation clock

TXE6	Enables/disables transmission
0	Disables transmission (synchronously resets the transmission circuit).
1	Enables transmission

RXE6	Enables/disables reception
0	Disables reception (synchronously resets the reception circuit).
1	Enables reception

Notes 1. 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[Note]
1	0	Outputs odd parity.	Judges as odd parity.
1	1	Outputs even parity.	Judges as even parity.

CL6	Specifies character length of transmit/receive data
0	Character length of data = 7 bits
1	Character length of data = 8 bits

SL6	Specifies number of stop bits of transmit data
0	Number of stop bits = 1
1	Number of stop bits = 2

ISRM6	Enables/disables occurrence of reception completion interrupt in case of error
0	“INTSRE6” occurs in case of error (at this time, INTSR6 does not occur).
1	“INTSR6” occurs in case of error (at this time, INTSRE6 does not occur).

Note If “reception as 0 parity” is selected, the parity is not judged. Therefore, bit 2 (PE6) of asynchronous serial interface reception error status register 6 (ASIS6) is not set and the error interrupt does not occur.

Cautions
1. **To start the transmission, set POWER6 to 1 and then set TXE6 to 1. To stop the transmission, clear TXE6 to 0, and then clear POWER6 to 0.**
2. **To start the reception, set POWER6 to 1 and then set RXE6 to 1. To stop the reception, clear RXE6 to 0, and then clear POWER6 to 0.**
3. **Set POWER6 to 1 and then set RXE6 to 1 while a high level is input to the RxD6 pin. If POWER6 is set to 1 and RXE6 is set to 1 while a low level is input, reception is started.**
4. **TXE6 and RXE6 are synchronized by the base clock (f_{XCLK6}) set by CKSR6. To enable transmission or reception again, set TXE6 or RXE6 to 1 at least two clocks of the base clock after TXE6 or RXE6 has been cleared to 0. If TXE6 or RXE6 is set within two clocks of the base clock, the transmission circuit or reception circuit may not be initialized.**
5. **Set transmit data to TXB6 at least one base clock (f_{XCLK6}) after setting TXE6 = 1.**
6. **Clear the TXE6 and RXE6 bits to 0 before rewriting the PS61, PS60, and CL6 bits.**
7. **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 sets this register to 00H if bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 0. 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 and 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 and 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 and 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. The first bit of the receive data is checked as the stop bit, regardless of the number of stop bits.

3. If an overrun error occurs, the next receive data is not written to receive buffer register 6 (RXB6) but discarded.

4. If data is read from ASIS6, a wait cycle is generated. For details, see CHAPTER 25 CAUTIONS FOR WAIT.

(3) Asynchronous serial interface transmission status register 6 (ASIF6)

This register indicates the status of transmission by serial interface UART6. It includes two status flag bits (TXBF6 and TXSF6).

Transmission can be continued without disruption even during an interrupt period, by writing the next data to the TXB6 register after data has been transferred from the TXB6 register to the TXS6 register.

This register is read-only by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H if bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 0.

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 sets this register to 00H.

Remark CKSR6 can be refreshed (the same value is written) by software during a communication operation (when bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1 or bit 7 (POWER6) and bit 5 (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_{XCLK6}) selection		
					f_{PRS} = 12 MHz	f_{PRS} = 16 MHz
0	0	0	0	f_{PRS}	12 MHz	16 MHz
0	0	0	1	$f_{PRS}/2$	6 MHz	8 MHz
0	0	1	0	$f_{PRS}/2^2$	3 MHz	4 MHz
0	0	1	1	$f_{PRS}/2^3$	1.5 MHz	2 MHz
0	1	0	0	$f_{PRS}/2^4$	750 kHz	1 MHz
0	1	0	1	$f_{PRS}/2^5$	375 kHz	500 kHz
0	1	1	0	$f_{PRS}/2^6$	187.5 kHz	250 kHz
0	1	1	1	$f_{PRS}/2^7$	93.75 kHz	125 kHz
1	0	0	0	$f_{PRS}/2^8$	46.875 kHz	62.5 kHz
1	0	0	1	$f_{PRS}/2^9$	23.438 kHz	31.25 kHz
1	0	1	0	$f_{PRS}/2^{10}$	11.719 kHz	15.625 kHz
1	0	1	1	TM50 output[Note]		
Other than above				Setting prohibited		

Note Note the following points when selecting the TM50 output as the base clock.

- Mode in which the count clock is cleared and started upon a match of TM50 and CR50 (TMC506 = 0)
  Start the operation of 8-bit timer/event counter 50 first and then enable the timer F/F inversion operation (TMC501 = 1).
- PWM mode (TMC506 = 1)
  Start the operation of 8-bit timer/event counter 50 first and then set the count clock to make the duty = 50%.

It is not necessary to enable the TO50 pin as a timer output pin in any mode.

Caution Make sure POWER6 = 0 when rewriting TPS63 to TPS60.

Remarks 1. f_{PRS}: Peripheral hardware clock frequency

2. TMC506: Bit 6 of 8-bit timer mode control register 50 (TMC50)
TMC501: Bit 1 of TMC50

(5) Baud rate generator control register 6 (BRGC6)

This register sets the division value of the 8-bit counter of serial interface UART6.

BRGC6 can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Remark BRGC6 can be refreshed (the same value is written) by software during a communication operation (when bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1 or bit 7 (POWER6) and bit 5 (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) Port mode register 1 (PM1)

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

When using the P13/TxD6 pin for serial interface data output, clear PM13 to 0 and set the output latch of P13 to 1.

When using the P14/RxD6 pin for serial interface data input, set PM14 to 1. The output latch of P14 at this time may be 0 or 1.

PM1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Figure 10-7. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10

PM1n	P1n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

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[Note 1]	Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit[Note 2].

TXE6	Enables/disables transmission
0	Disables transmission operation (synchronously resets the transmission circuit).

RXE6	Enables/disables reception
0	Disables reception (synchronously resets the reception circuit).

Notes 1. 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/P14 and TxD6/P13 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)
- Port mode register 1 (PM1)
- Port register 1 (P1)

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

<1> Set the CKSR6 register (see **Figure 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 bit 7 (POWER6) of the ASIM6 register to 1.
<5> 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.
<6> Write data to transmit buffer register 6 (TXB6). → Data transmission is started.

Caution Take relationship with the other party of communication when setting the port mode register and port register.

The relationship between the register settings and pins is shown below.

Table 10-2. Relationship Between Register Settings and Pins

POWER6	TXE6	RXE6	PM13	P13	PM14	P14	UART6 Operation	Pin Function	
								TxD6/P13	RxD6/P14
0	0	0	×Note	×Note	×Note	×Note	Stop	P13	P14
1	0	1	×Note	×Note	1	×	Reception	P13	RxD6
	1	0	0	1	×Note	×Note	Transmission	TxD6	P14
	1	1	0	1	1	×	Transmission/ reception	TxD6	RxD6

Note Can be set as port function.

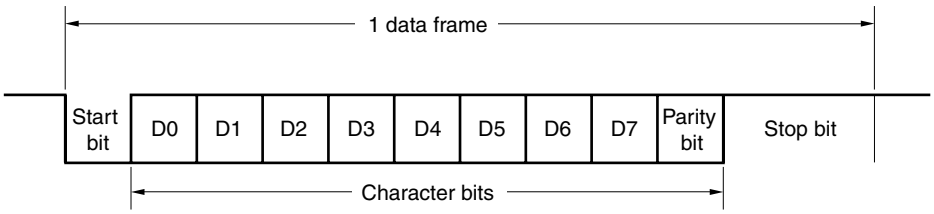
Remark ×: don't care
POWER6: Bit 7 of asynchronous serial interface operation mode register 6 (ASIM6)
TXE6: Bit 6 of ASIM6
RXE6: Bit 5 of ASIM6
PM1×: Port mode register
P1×: Port output latch

(2) Communication operation

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

Figures 10-8 and 10-9 show the format and waveform example of the normal transmit/receive data.

Figure 10-8. Format of Normal UART Transmit/Receive Data



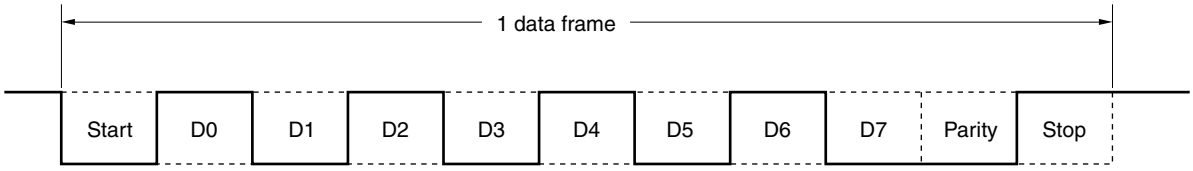
One data frame consists of the following bits.

- Start bit ... 1 bit
- Character bits ... 7 or 8 bits
- Parity bit ... Even parity, odd parity, 0 parity, or no parity
- Stop bit ... 1 or 2 bits

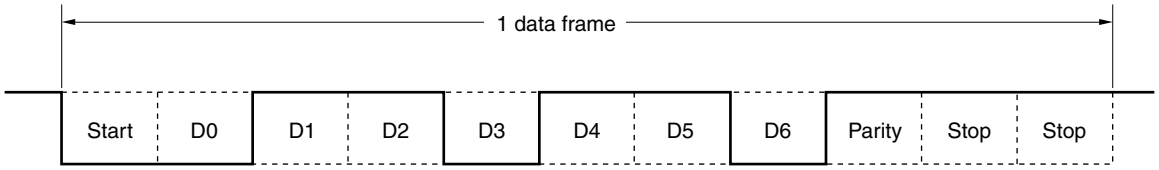
The character bit length, parity, and stop bit length in one data frame are specified by asynchronous serial interface operation mode register 6 (ASIM6).

Figure 10-9. Example of Normal UART Transmit/Receive Data Waveform

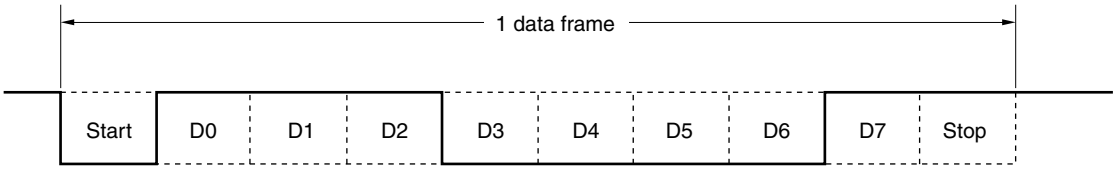
1. Data length: 8 bits, LSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H



2. Data length: 7 bits, LSB first, Parity: Odd parity, Stop bit: 2 bits, Communication data: 36H



3. 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.

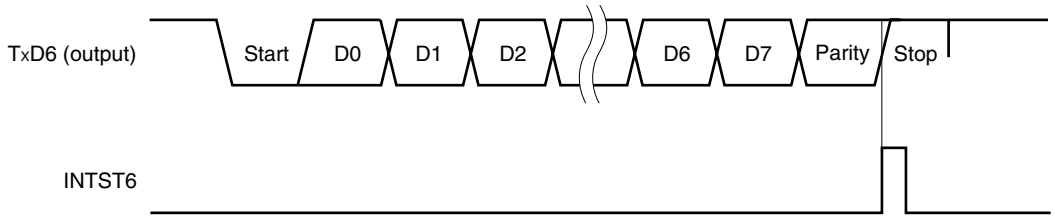
When transmission is started, the data in TXB6 is transferred to transmit shift register 6 (TXS6). After that, the transmit data is sequentially output from TXS6 to the TxD6 pin. When transmission is completed, the parity and stop bits set by ASIM6 are appended and a transmission completion interrupt request (INTST6) is generated.

Transmission is stopped until the data to be transmitted next is written to TXB6.

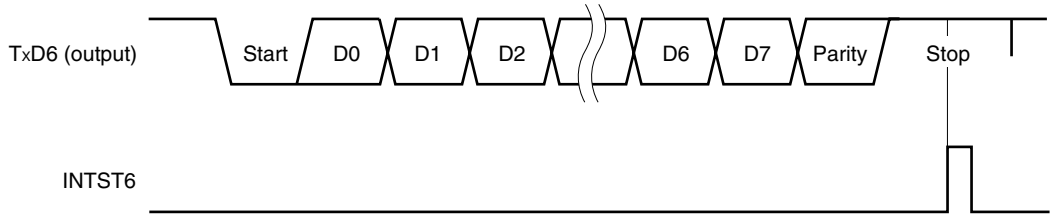
Figure 10-10 shows the timing of the transmission completion interrupt request (INTST6). This interrupt occurs as soon as the last stop bit has been output.

Figure 10-10. 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.

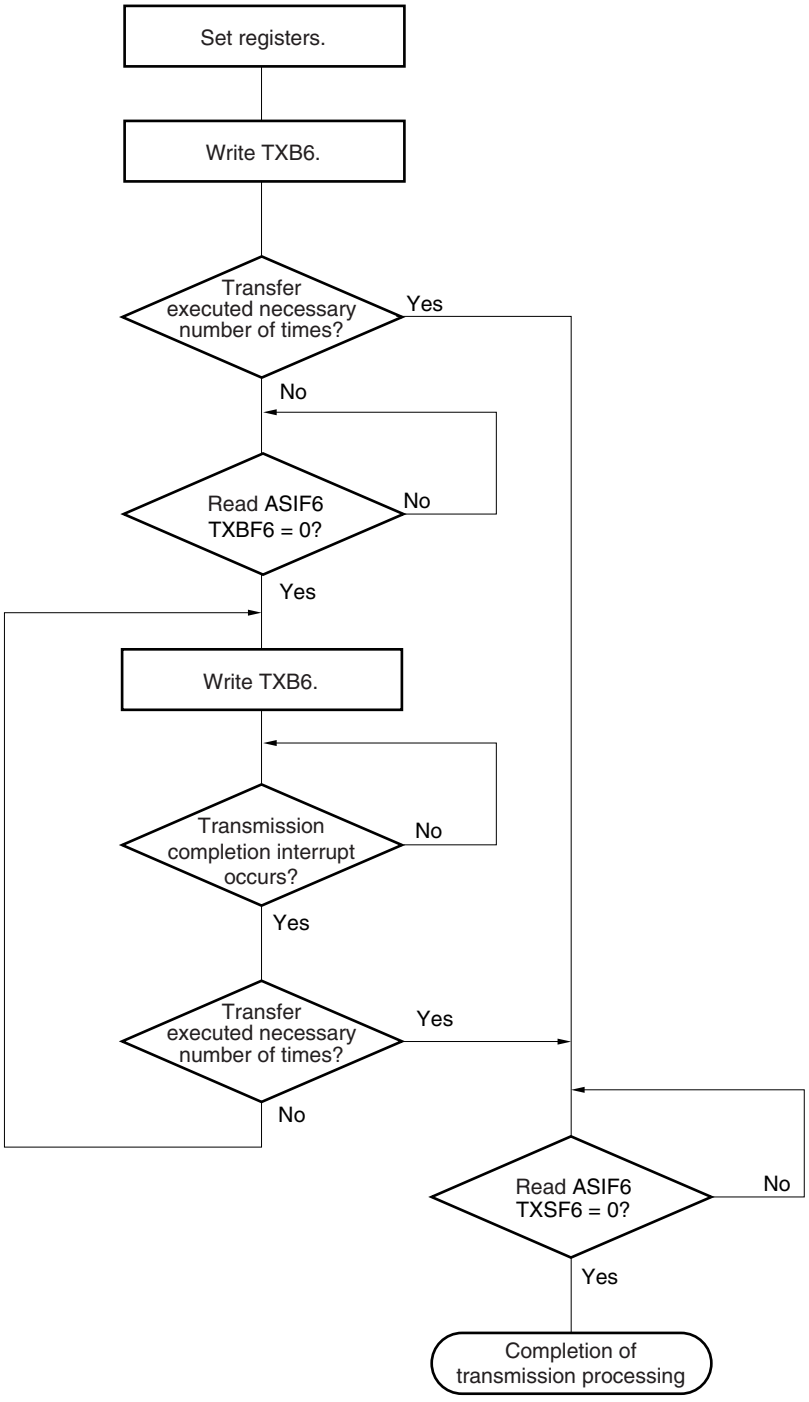
TXSF6	Transmission Status
0	Transmission is completed.
1	Transmission is in progress.

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 10-11 shows an example of the continuous transmission processing flow.

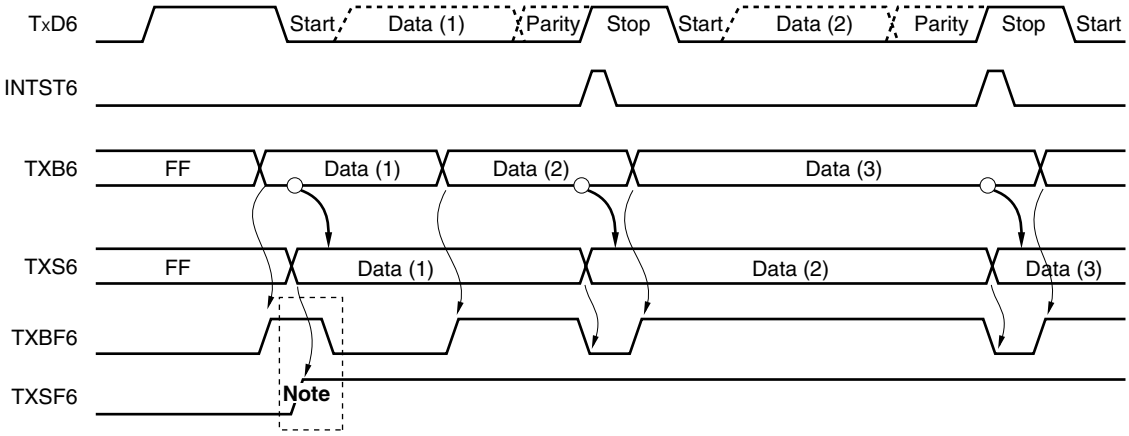
Figure 10-11. Example of Continuous Transmission Processing Flow



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 10-12 shows the timing of starting continuous transmission, and Figure 10-13 shows the timing of ending continuous transmission.

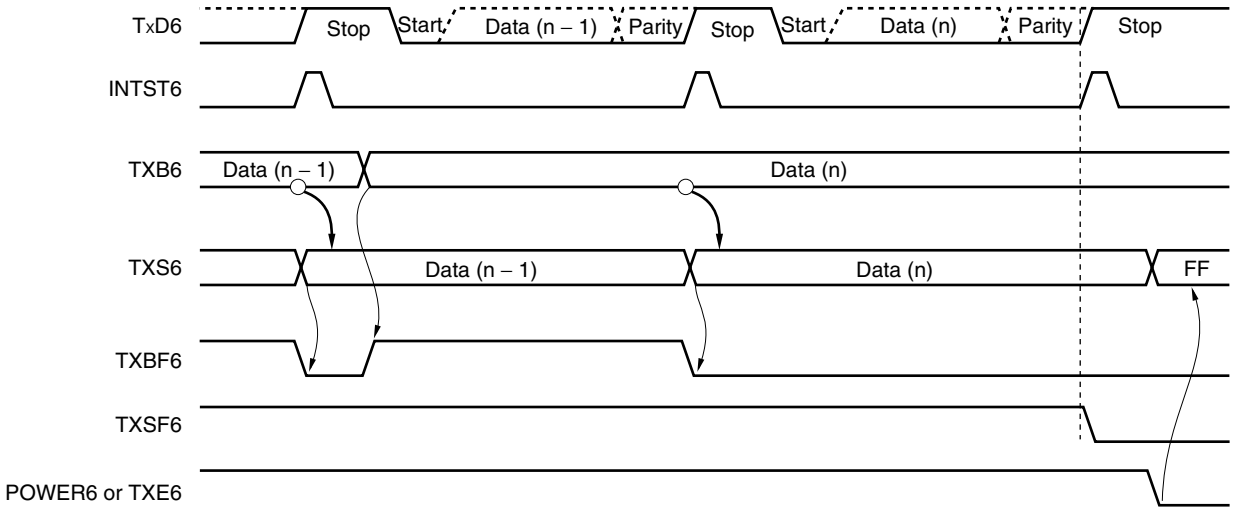
Figure 10-12. Timing of Starting Continuous Transmission



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 10-13. Timing of Ending Continuous Transmission



Remark TxD6: TxD6 pin (output)
INTST6: Interrupt request signal
TXB6: Transmit buffer register 6
TXS6: Transmit shift register 6
ASIF6: Asynchronous serial interface transmission status register 6
TXBF6: Bit 1 of ASIF6
TXSF6: Bit 0 of ASIF6
POWER6: Bit 7 of asynchronous serial interface operation mode register (ASIM6)
TXE6: Bit 6 of asynchronous serial interface operation mode register (ASIM6)

(e) Normal reception

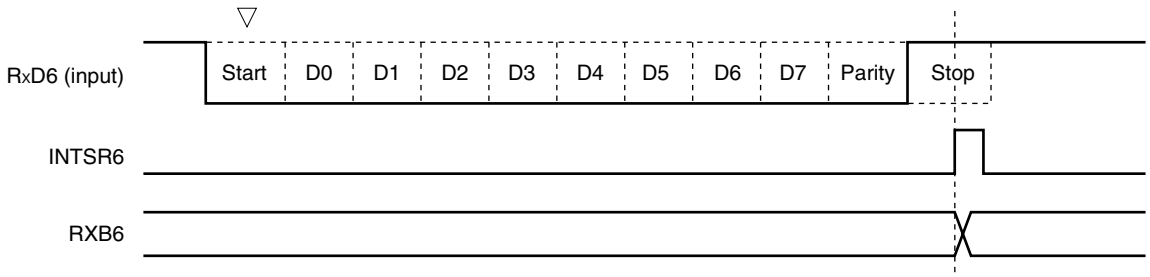
Reception is enabled and the RxD6 pin input is sampled when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1 and then bit 5 (RXE6) of ASIM6 is set to 1.

The 8-bit counter of the baud rate generator starts counting when the falling edge of the RxD6 pin input is detected. When the set value of baud rate generator control register 6 (BRGC6) has been counted, the RxD6 pin input is sampled again (in Figur▽10-14). 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 10-14. 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

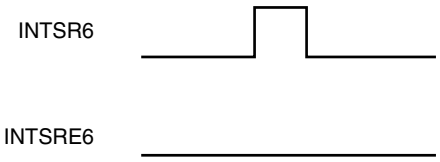
Reception Error	Cause
Parity error	The parity specified for transmission does not match the parity of the receive data.
Framing error	Stop bit is not detected.
Overrun error	Reception of the next data is completed before data is read from receive buffer register 6 (RXB6).

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.

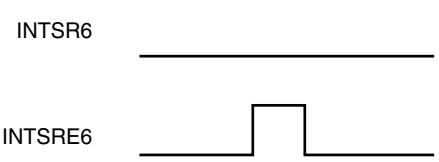
Figure 10-15. Reception Error Interrupt

1. If ISRM6 is cleared to 0 (reception completion interrupt (INTSR6) and error interrupt (INTSRE6) are separated)

(a) No error during reception

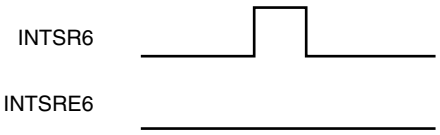


(b) Error during reception

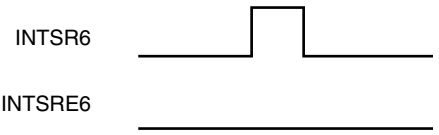


2. If ISRM6 is set to 1 (error interrupt is included in INTSR6)

(a) No error during reception



(b) Error during reception



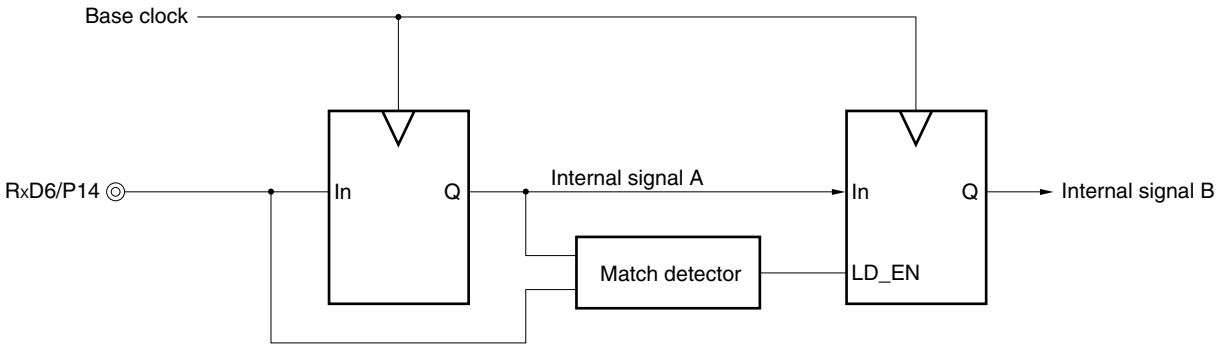
(g) Noise filter of receive data

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 10-16, the internal processing of the reception operation is delayed by two clocks from the external signal status.

Figure 10-16. 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.

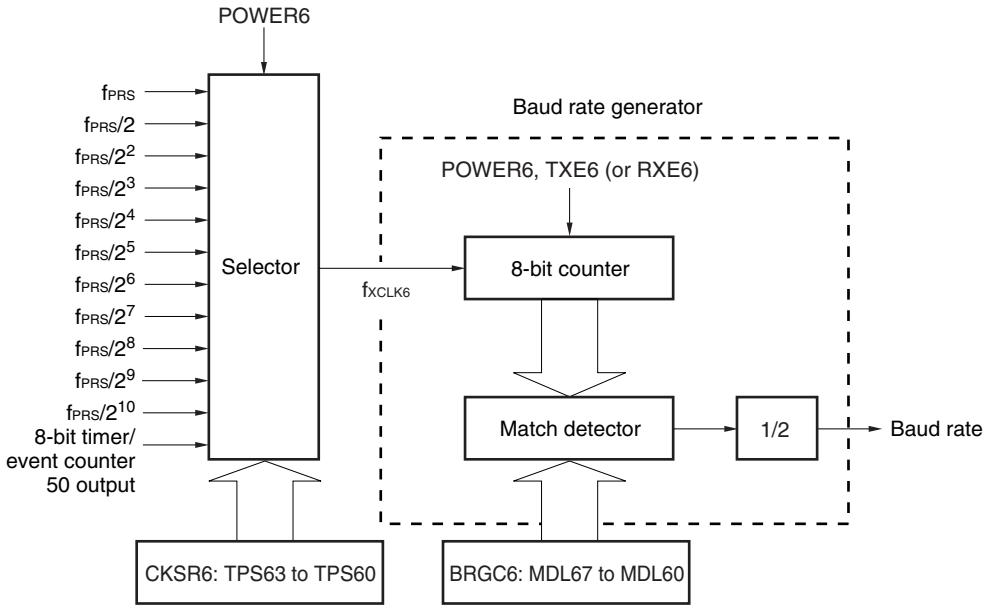
(1) Configuration of baud rate generator

- 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_{XCLK6}. 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 10-17. 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.

Table 10-4. Set Value of TPS63 to TPS60

TPS63	TPS62	TPS61	TPS60	Base clock (f_{XCLK6}) selection		
					f_{PRS} = 12 MHz	f_{PRS} = 16 MHz
0	0	0	0	f_{PRS}	12 MHz	16 MHz
0	0	0	1	$f_{PRS}/2$	6 MHz	8 MHz
0	0	1	0	$f_{PRS}/2^2$	3 MHz	4 MHz
0	0	1	1	$f_{PRS}/2^3$	1.5 MHz	2 MHz
0	1	0	0	$f_{PRS}/2^4$	750 kHz	1 MHz
0	1	0	1	$f_{PRS}/2^5$	375 kHz	500 kHz
0	1	1	0	$f_{PRS}/2^6$	187.5 kHz	250 kHz
0	1	1	1	$f_{PRS}/2^7$	93.75 kHz	125 kHz
1	0	0	0	$f_{PRS}/2^8$	46.875 kHz	62.5 kHz
1	0	0	1	$f_{PRS}/2^9$	23.438 kHz	31.25 kHz
1	0	1	0	$f_{PRS}/2^{10}$	11.719 kHz	15.625 kHz
1	0	1	1	TM50 output		
Other than above				Setting prohibited		

(a) Baud rate

The baud rate can be calculated by the following expression.

- Baud rate = $\frac{f_{XCLK6}}{2 \times k}$ [bps]

f_{XCLK6}: Frequency of base clock selected by TPS63 to TPS60 bits of CKSR6 register

k: Value set by MDL67 to MDL60 bits of BRGC6 register (k = 4, 5, 6, ..., 255)

(b) 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 = 16 MHz = 16,000,000 Hz
Set value of MDL67 to MDL60 bits of BRGC6 register = 01000101B (k = 69)
Target baud rate = 115200 bps

Baud rate = 16 M/(2 × 69)
= 16000000/(2 × 69) = 115942 [bps]

Error = (115942/115200 − 1) × 100
= 0.644 [%]

(3) Example of setting baud rate

Table 10-5. Set Data of Baud Rate Generator

Baud Rate [bps]	f_{PRS} = 12.0 MHz				f_{PRS} = 16.0 MHz			
	TPS63to TPS60	k	Calculated Value	ERR[%]	TPS63to TPS60	k	Calculated Value	ERR[%]
300	9H	39	300.487	0.16	AH	26	300.481	0.16
600	8H	39	600.962	0.16	AH	13	600.962	0.16
1200	7H	39	1201.92	0.16	9H	13	1201.92	0.16
2400	6H	39	2403.85	0.16	8H	13	2403.85	0.16
4800	5H	39	4807.69	0.16	7H	13	4807.69	0.16
9600	4H	39	9615.38	0.16	6H	13	9615.38	0.16
19200	3H	39	19230.8	0.16	5H	13	19230.8	0.16
24000	1H	125	24000	0.00	1H	167	23952.1	–0.20
31250	5H	6	31250	0.00	5H	8	31250	0.00
38400	2H	39	38461.5	0.16	4H	13	38461.5	0.16
48000	0H	125	48000	0.00	0H	167	47904.2	–0.20
76800	1H	39	76923.1	0.16	3H	13	76923.1	0.16
115200	2H	13	115385	0.16	0H	69	115942	0.64
153600	0H	39	153846	0.16	2H	13	153846	0.14
312500	0H	19	315789	1.05	1H	13	307692	–1.54

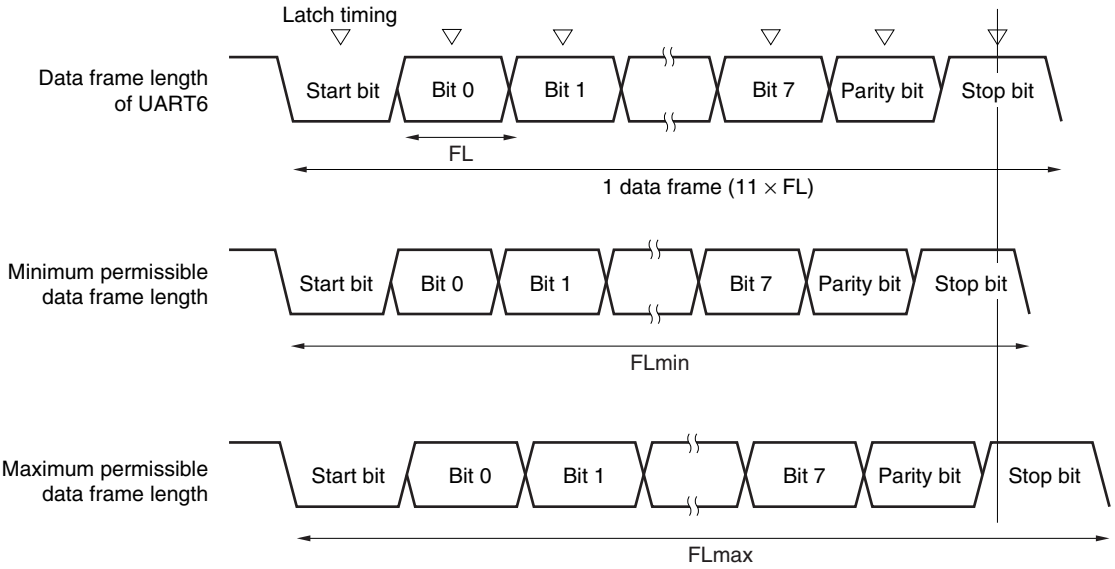
Remark TPS63 to TPS60: Bits 3 to 0 of clock selection register 6 (CKSR6) (setting of base clock (f_{XCLK6}))
k: Value set by MDL67 to MDL60 bits of baud rate generator control register 6 (BRGC6) (k = 4, 5, 6, ..., 255)
f_{PRS}: Peripheral hardware clock frequency
ERR: Baud rate error

(4) Permissible baud rate range during reception

The permissible error from the baud rate at the transmission destination during reception is shown below.

Caution Make sure that the baud rate error during reception is within the permissible error range, by using the calculation expression shown below.

Figure 10-18. Permissible Baud Rate Range During Reception



As shown in Figure 10-18, 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.

$FL = (Brate)^{-1}$

Brate: Baud rate of UART6
k: Set value of BRGC6
FL: 1-bit data length
Margin of latch timing: 2 clocks

Minimum permissible data frame length: $FLmin = 11 \times FL - \frac{k-2}{2k} \times FL = \frac{21k+2}{2k} FL$

Therefore, the maximum receivable baud rate at the transmission destination is as follows.

$$BRmax = (FLmin/11)^{-1} = \frac{22k}{21k+2} Brate$$

Similarly, the maximum permissible data frame length can be calculated as follows.

$$\frac{10}{11} \times FLmax = 11 \times FL - \frac{k+2}{2 \times k} \times FL = \frac{21k-2}{2 \times k} FL$$

$$FLmax = \frac{21k-2}{20k} FL \times 11$$

Therefore, the minimum receivable baud rate at the transmission destination is as follows.

$$BRmin = (FLmax/11)^{-1} = \frac{20k}{21k-2} Brate$$

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
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.

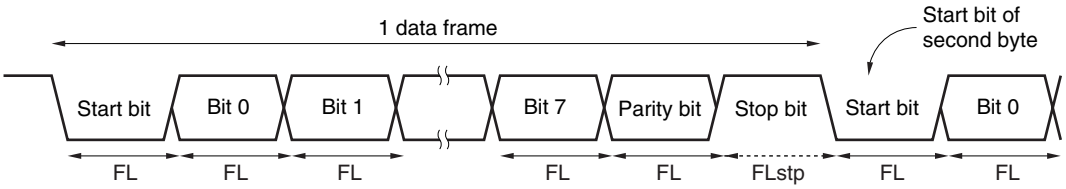
2. k: Set value of BRGC6

10.5 Cautions for Serial Interface UART6

(1) 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 10-19. 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.

$$FLstp = FL + 2/f_{XCLK6}$$

Therefore, the data frame length during continuous transmission is:

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

(2) Operating current in STOP mode

The UART6 operation is stopped in the STOP mode. At this time, the operating current can be reduced by clearing bits 7 (POWER6), 6 (TXE6), and 5 (RXE6) of the asynchronous serial interface operation mode register (ASIM6) to 0. To resume the operation from the standby status, first clear bit 7 (POWER6) of interrupt request flag register 0L (IF0L), bits 1 (STIF6) and 0 (SRIF6) of interrupt request flag register 0H (IF0H) to 1, and then start operation.

CHAPTER 11 SERIAL INTERFACE CSI10

11.1 Functions of Serial Interface CSI10

Serial interface CSI10 has the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

(1) Operation stop mode

This mode is used when serial communication is not performed and can enable a reduction in the power consumption.

For details, see **11.4.1 Operation stop mode**.

(2) 3-wire serial I/O mode (MSB/LSB-first selectable)

This mode is used to communicate 8-bit data using three lines: a serial clock line ($\overline{\text{SCK10}}$) and two serial data lines (SI10 and SO10).

The processing time of data communication can be shortened in the 3-wire serial I/O mode because transmission and reception can be simultaneously executed.

In addition, whether 8-bit data is communicated with the MSB or LSB first can be specified, so this interface can be connected to any device.

The 3-wire serial I/O mode is used for connecting peripheral ICs and display controllers with a clocked serial interface.

For details, see **11.4.2 3-wire serial I/O mode**.

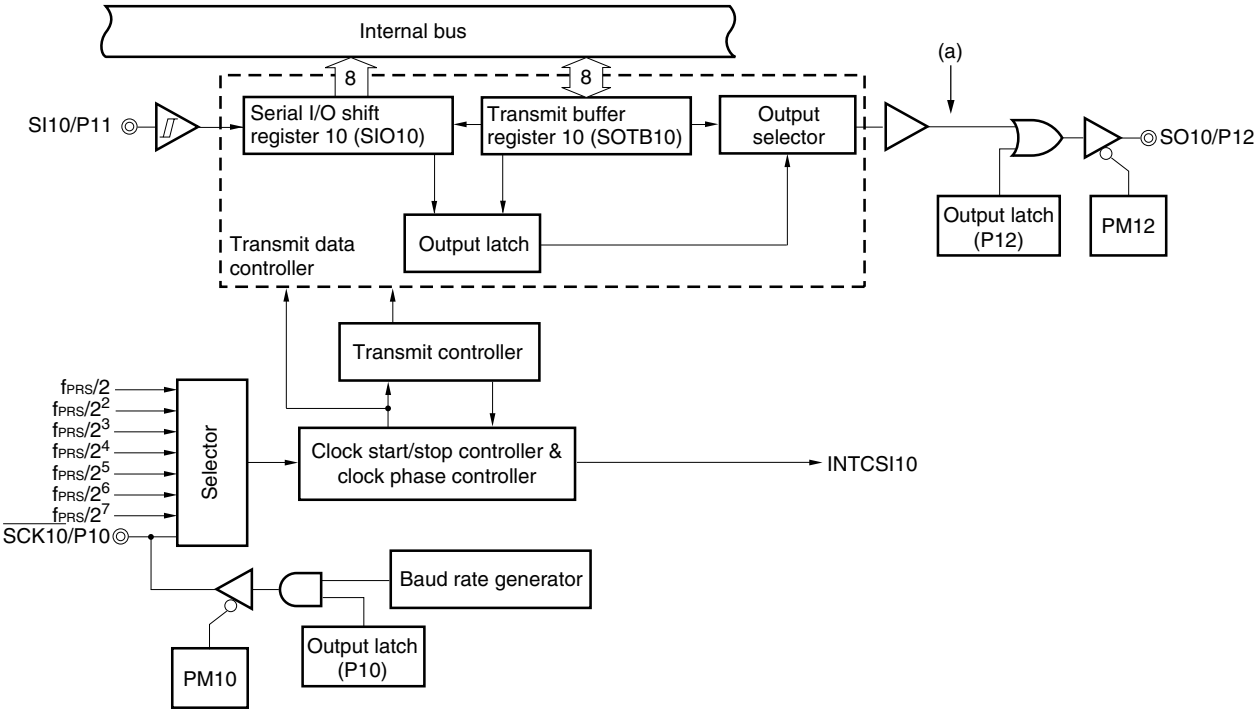
11.2 Configuration of Serial Interface CSI10

Serial interface CSI10 includes the following hardware.

Table 11-1. Configuration of Serial Interface CSI10

Item	Configuration
Controller	Transmit controller Clock start/stop controller & clock phase controller
Registers	Transmit buffer register 10 (SOTB10) Serial I/O shift register 10 (SIO10)
Control registers	Serial operation mode register 10 (CSIM10) Serial clock selection register 10 (CSIC10) Port mode register 0 (PM0) or port mode register 1 (PM1) Port register 0 (P0) or port register 1 (P1)

Figure 11-1. Block Diagram of Serial Interface CSI10



Remark (a): SO10 output

(1) Transmit buffer register 10 (SOTB10)

This register sets the transmit data.

Transmission/reception is started by writing data to SOTB10 when bit 7 (CSIE10) and bit 6 (TRMD10) of serial operation mode register 10 (CSIM10) is 1.

The data written to SOTB10 is converted from parallel data into serial data by serial I/O shift register 10, and output to the serial output pin (SO10).

SOTB10 can be written or read by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

Caution Do not access SOTB10 when CSOT10 = 1 (during serial communication).

(2) Serial I/O shift register 10 (SIO10)

This is an 8-bit register that converts data from parallel data into serial data and vice versa.

This register can be read by an 8-bit memory manipulation instruction.

Reception is started by reading data from SIO10 if bit 6 (TRMD10) of serial operation mode register 10 (CSIM10) is 0.

During reception, the data is read from the serial input pin (SI10) to SIO10.

Reset signal generation sets this register to 00H.

Caution Do not access SIO10 when CSOT10 = 1 (during serial communication).

11.3 Registers Controlling Serial Interface CSI10

Serial interface CSI10 is controlled by the following four registers.

- Serial operation mode register 10 (CSIM10)
- Serial clock selection register 10 (CSIC10)
- Port mode register 1 (PM1)
- Port port register 1 (P1)

(1) Serial operation mode register 10 (CSIM10)

CSIM10 is used to select the operation mode and enable or disable operation.

CSIM10 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

Figure 11-2. Format of Serial Operation Mode Register 10 (CSIM10)

Address: FF80H After reset: 00H R/W[Note 1]

Symbol	<7>	6	5	4	3	2	1	0
CSIM10	CSIE10	TRMD10	0	DIR10	0	0	0	CSOT10

CSIE10	Operation control in 3-wire serial I/O mode
0	Disables operation[Note 2] and asynchronously resets the internal circuit[Note 3].
1	Enables operation

TRMD10[Note 4]	Transmit/receive mode control
0[Note 5]	Receive mode (transmission disabled).
1	Transmit/receive mode

DIR10[Note 6]	First bit specification
0	MSB
1	LSB

CSOT10	Communication status flag
0	Communication is stopped.
1	Communication is in progress.

Notes 1. Bit 0 is a read-only bit.

2. To use P10/$\overline{SCK10}$ and P12/SO10 as general-purpose ports, set CSIM10 in the default status (00H).

3. Bit 0 (CSOT10) of CSIM10 and serial I/O shift register 10 (SIO10) are reset.

4. Do not rewrite TRMD10 when CSOT10 = 1 (during serial communication).

5. The SO10 output (see **(a)** in **Figure 11-1**) is fixed to the low level when TRMD10 is 0. Reception is started when data is read from SIO10.

6. Do not rewrite DIR10 when CSOT10 = 1 (during serial communication).

Caution Be sure to clear bits 1 to 3 and 5 to 0.

(2) Serial clock selection register 10 (CSIC10)

This register specifies the timing of the data transmission/reception and sets the serial clock.

CSIC10 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

Figure 11-3. Format of Serial Clock Selection Register 10 (CSIC10)

Address: FF81H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CSIC10	0	0	0	CKP10	DAP10	CKS102	CKS101	CKS100

CKP10	DAP10	Specification of data transmission/reception timing	Type
0	0	$\overline{SCK10}$, SO10 (D7 D6 D5 D4 D3 D2 D1 D0), SI10 input timing	1
0	1	$\overline{SCK10}$, SO10 (D7 D6 D5 D4 D3 D2 D1 D0), SI10 input timing	2
1	0	$\overline{SCK10}$, SO10 (D7 D6 D5 D4 D3 D2 D1 D0), SI10 input timing	3
1	1	$\overline{SCK10}$, SO10 (D7 D6 D5 D4 D3 D2 D1 D0), SI10 input timing	4

CKS102	CKS101	CKS100	CSI10 serial clock selection			Mode
				f_{PRS} = 12 MHz	f_{PRS} = 16 MHz	
0	0	0	$f_{PRS}/2$	6 MHz	8 MHz	Master mode
0	0	1	$f_{PRS}/2^2$	3 MHz	4 MHz	
0	1	0	$f_{PRS}/2^3$	1.5 MHz	2 MHz	
0	1	1	$f_{PRS}/2^4$	750 kHz	1 MHz	
1	0	0	$f_{PRS}/2^5$	375 kHz	500 kHz	
1	0	1	$f_{PRS}/2^6$	187.5 kHz	250 kHz	
1	1	0	$f_{PRS}/2^7$	93.75 kHz	125 kHz	
1	1	1	External clock input to $\overline{SCK10}$			Slave mode

Cautions 1. Do not write to CSIC10 while CSIE10 = 1 (operation enabled).

2. To use P10/$\overline{SCK10}$ and P12/SO10 as general-purpose ports, set CSIC10 in the default status (00H).

3. The phase type of the data clock is type 1 after reset.

Remark f_{PRS}: Peripheral hardware clock frequency

(3) Port mode register 1 (PM1)

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

When using P10/$\overline{SCK10}$ as the clock output pin of the serial interface, clear PM10 to 0, and set the output latches of P10 to 1.

When using P12/SO10 as the data output pin of the serial interface, clear PM12 and the output latch of P12 to 0.

When using P10/$\overline{SCK10}$ as the clock input pin of the serial interface, and P11/SI10 as the data input pin, set PM10 and PM11 to 1. At this time, the output latches of P10 and P11 may be 0 or 1.

PM1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Figure 11-4. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10

PM1n	P1n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

11.4 Operation of Serial Interface CSI10

Serial interface CSI10 can be used in the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

11.4.1 Operation stop mode

Serial communication is not executed in this mode. Therefore, the power consumption can be reduced. In addition, the P10/$\overline{SCK10}$, P11/SI10, and P12/SO10 pins can be used as ordinary I/O port pins in this mode.

(1) Register used

The operation stop mode is set by serial operation mode register 10 (CSIM10).

To set the operation stop mode, clear bit 7 (CSIE10) of CSIM10 to 0.

(a) Serial operation mode register 10 (CSIM10)

CSIM10 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets CSIM10 to 00H.

- Serial operation mode register 10 (CSIM10)

Address: FF80H After reset: 00H R/W

Symbol	<7>	6	5	4	3	2	1	0
CSIM10	CSIE10	TRMD10	0	DIR10	0	0	0	CSOT10

CSIE10	Operation control in 3-wire serial I/O mode
0	Disables operation[Note 1] and asynchronously resets the internal circuit[Note 2].

Notes 1. To use P10/$\overline{SCK10}$ and P12/SO10 as general-purpose ports, set CSIM10 in the default status (00H).

2. Bit 0 (CSOT10) of CSIM10 and serial I/O shift register 10 (SIO10) are reset.

11.4.2 3-wire serial I/O mode

The 3-wire serial I/O mode is used for connecting peripheral ICs and display controllers with a clocked serial interface.

In this mode, communication is executed by using three lines: the serial clock ($\overline{SCK10}$), serial output (SO10), and serial input (SI10) lines.

(1) Registers used

- Serial operation mode register 10 (CSIM10)
- Serial clock selection register 10 (CSIC10)
- Port mode register 1 (PM1)
- Port register 1 (P1)

The basic procedure of setting an operation in the 3-wire serial I/O mode is as follows.

<1> Set the CSIC10 register (see **Figure 11-3**).
<2> Set bits 0, 4 and 6 (CSOT10, DIR10, and TRMD10) of the CSIM10 register (see **Figure 11-2**).
<3> Set bit 7 (CSIE10) of the CSIM10 register to 1. → Transmission/reception is enabled.
<4> Write data to transmit buffer register 10 (SOTB10). → Data transmission/reception is started.
Read data from serial I/O shift register 10 (SIO10). → Data reception is started.

Caution Take relationship with the other party of communication when setting the port mode register and port register.

The relationship between the register settings and pins is shown below.

Table 11-2. Relationship Between Register Settings and Pins

CSIE10	TRMD10	PM11	P11	PM12	P12	PM10	P10	CSI10 Operation	Pin Function		
									SI10/P11	SO10/P12	$\overline{SCK10}$/P10
0	×	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	Stop	P11	P12	P10Note 2
1	0	1	×	×Note 1	×Note 1	1	×	Slave receptionNote 3	SI10	P12	$\overline{SCK10}$ (input)Note 3
1	1	×Note 1	×Note 1	0	0	1	×	Slave transmissionNote 3	P11	SO10	$\overline{SCK10}$ (input)Note 3
1	1	1	×	0	0	1	×	Slave transmission/ receptionNote 3	SI10	SO10	$\overline{SCK10}$ (input)Note 3
1	0	1	×	×Note 1	×Note 1	0	1	Master reception	SI10	P12	$\overline{SCK10}$ (output)
1	1	×Note 1	×Note 1	0	0	0	1	Master transmission	P11	SO10	$\overline{SCK10}$ (output)
1	1	1	×	0	0	0	1	Master transmission/ reception	SI10	SO10	$\overline{SCK10}$ (output)

Notes 1. Can be set as port function.

2. To use P10/$\overline{SCK10}$ as port pins, clear CKP10 to 0.

3. To use the slave mode, set CKS102, CKS101, and CKS100 to 1, 1, 1.

Remark ×: don't care

CSIE10: Bit 7 of serial operation mode register 10 (CSIM10)

TRMD10: Bit 6 of CSIM10

CKP10: Bit 4 of serial clock selection register 10 (CSIC10)

CKS102, CKS101, CKS100: Bits 2 to 0 of CSIC10

PM1×: Port mode register

P1×: Port output latch

(2) Communication operation

In the 3-wire serial I/O mode, data is transmitted or received in 8-bit units. Each bit of the data is transmitted or received in synchronization with the serial clock.

Data can be transmitted or received if bit 6 (TRMD10) of serial operation mode register 10 (CSIM10) is 1. Transmission/reception is started when a value is written to transmit buffer register 10 (SOTB10). In addition, data can be received when bit 6 (TRMD10) of serial operation mode register 10 (CSIM10) is 0.

Reception is started when data is read from serial I/O shift register 10 (SIO10).

After communication has been started, bit 0 (CSOT10) of CSIM10 is set to 1. When communication of 8-bit data has been completed, a communication completion interrupt request flag (CSIIF1n) is set, and CSOT10 is cleared to 0. Then the next communication is enabled.

Caution Do not access the control register and data register when CSOT10 = 1 (during serial communication).

Figure 11-5. Timing in 3-Wire Serial I/O Mode (1/2)

(a) Transmission/reception timing (Type 1: TRMD10 = 1, DIR10 = 0, CKP10 = 0, DAP10 = 0)

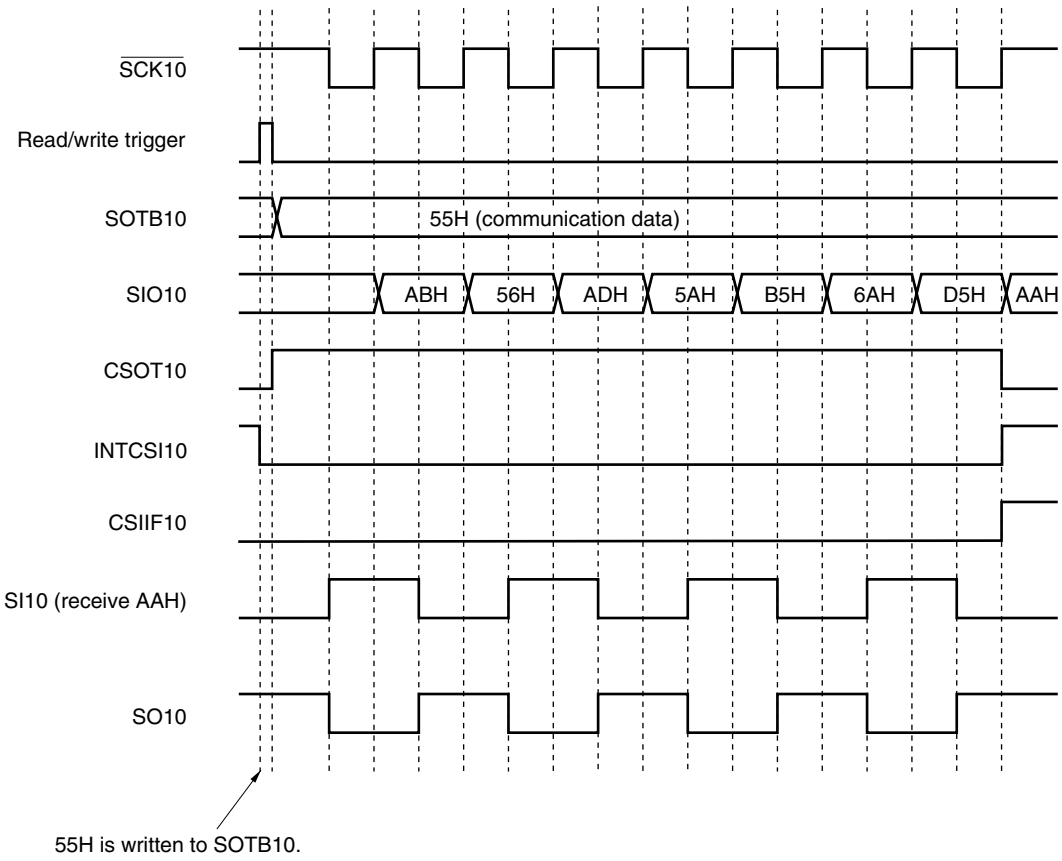


Figure 11-5. Timing in 3-Wire Serial I/O Mode (2/2)

(b) Transmission/reception timing (Type 2: TRMD10 = 1, DIR10 = 0, CKP10 = 0, DAP10 = 1)

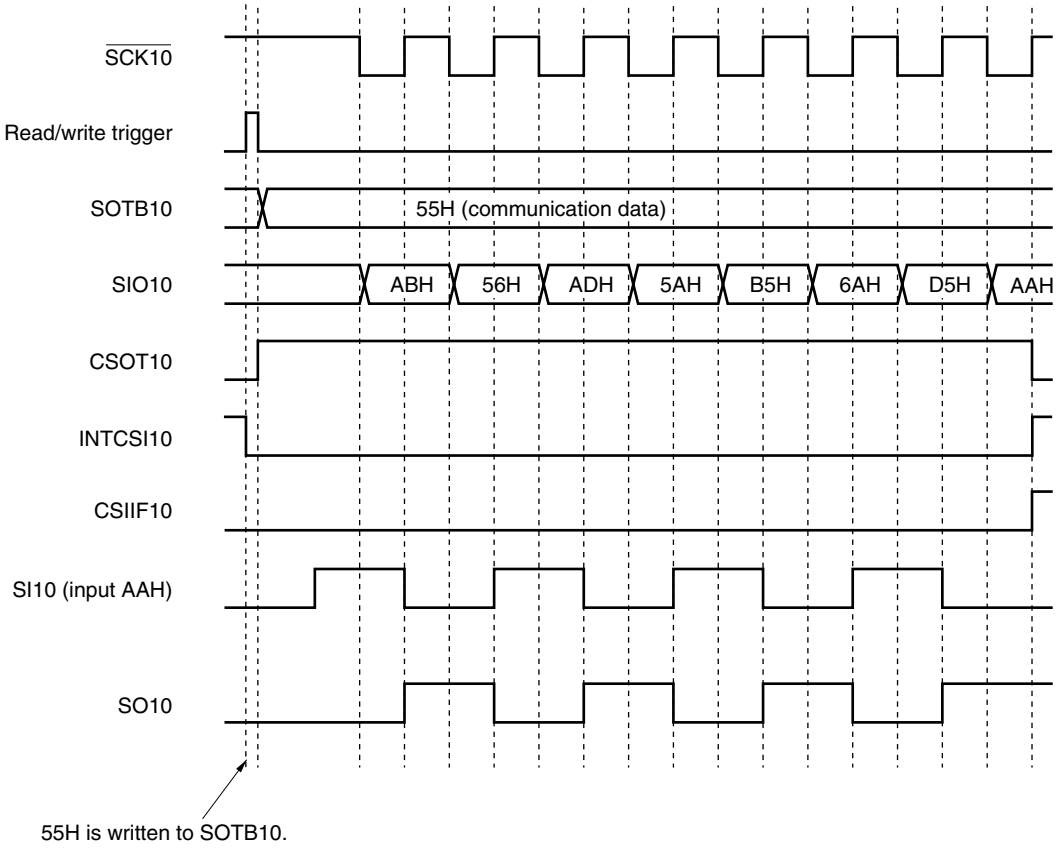
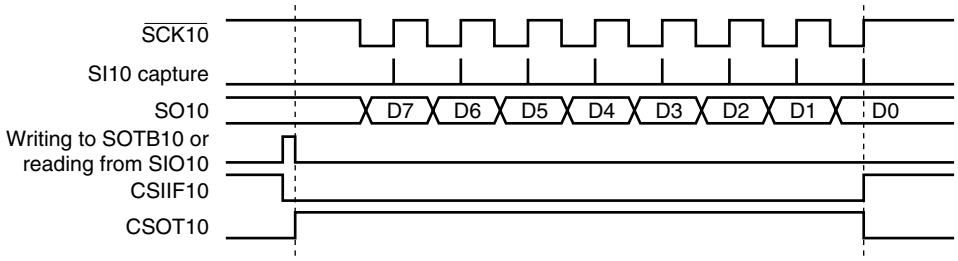
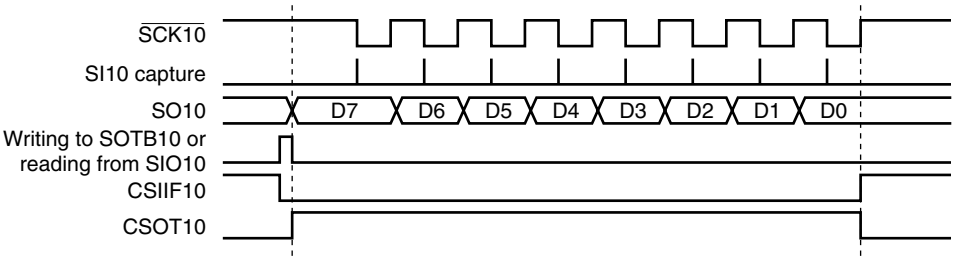


Figure 11-6. Timing of Clock/Data Phase

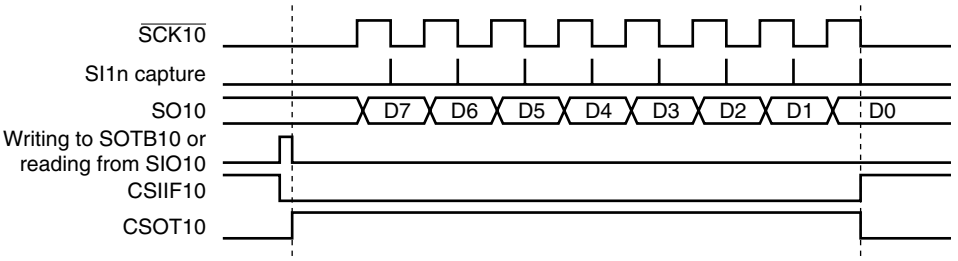
(a) Type 1: CKP10 = 0, DAP10 = 0, DIR10 = 0



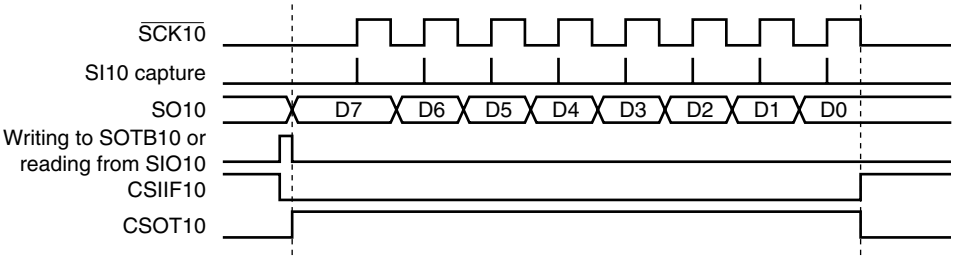
(b) Type 2: CKP10 = 0, DAP10 = 1, DIR10 = 0



(c) Type 3: CKP10 = 1, DAP10 = 0, DIR10 = 0



(d) Type 4: CKP10 = 1, DAP10 = 1, DIR10 = 0

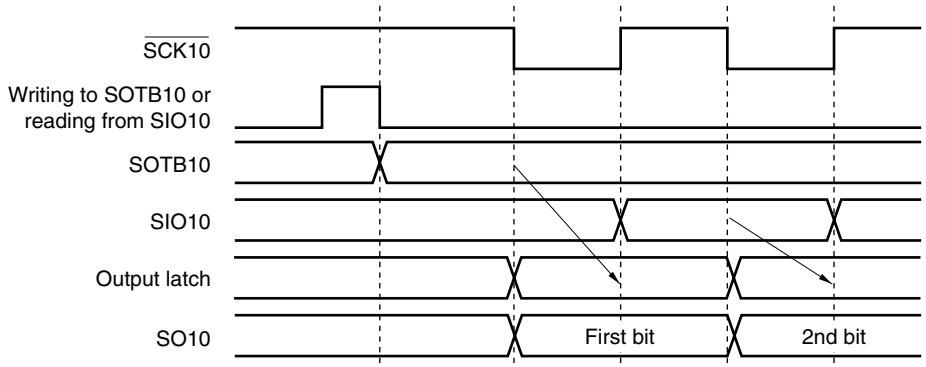


(3) Timing of output to SO10 pin (first bit)

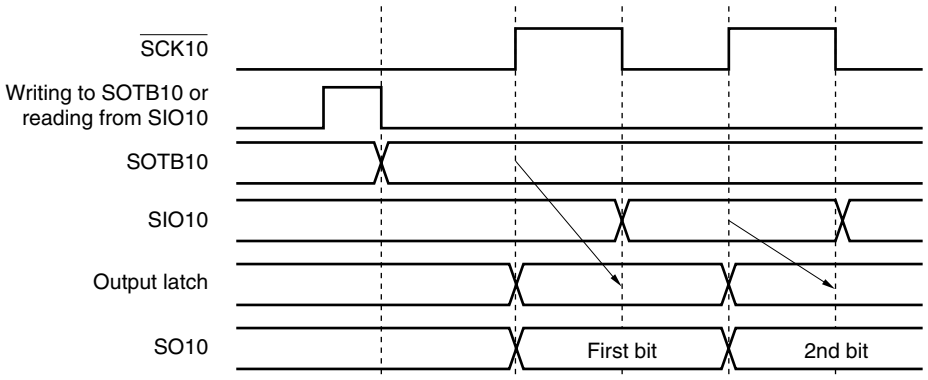
When communication is started, the value of transmit buffer register 10 (SOTB10) is output from the SO10 pin. The output operation of the first bit at this time is described below.

Figure 11-7. Output Operation of First Bit (1/2)

(a) Type 1: CKP10 = 0, DAP10 = 0



(b) Type 3: CKP10 = 1, DAP10 = 0

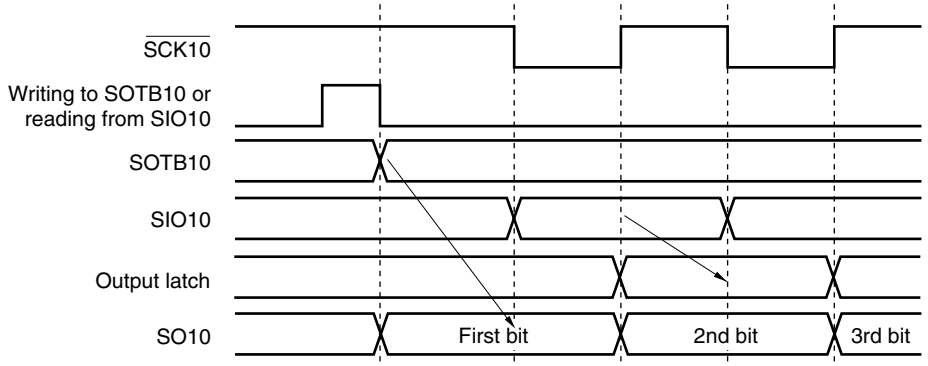


The first bit is directly latched by the SOTB10 register to the output latch at the falling (or rising) edge of $\overline{SCK10}$, and output from the SO10 pin via an output selector. Then, the value of the SOTB10 register is transferred to the SIO10 register at the next rising (or falling) edge of $\overline{SCK10}$, and shifted one bit. At the same time, the first bit of the receive data is stored in the SIO10 register via the SI10 pin.

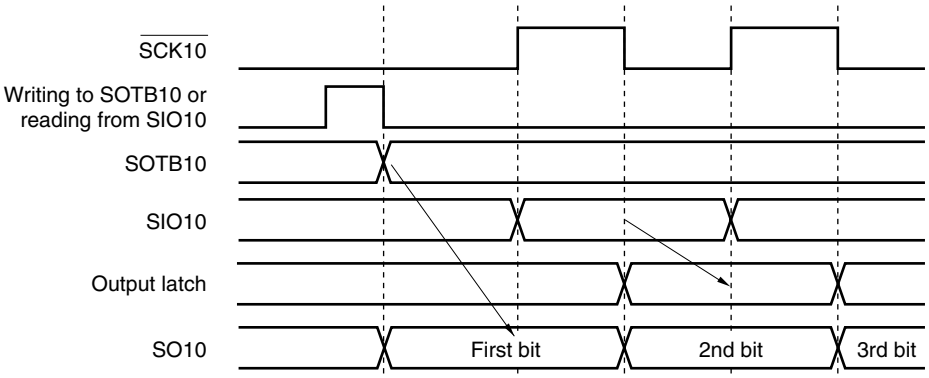
The second and subsequent bits are latched by the SIO10 register to the output latch at the next falling (or rising) edge of $\overline{SCK10}$, and the data is output from the SO10 pin.

Figure 11-7. Output Operation of First Bit (2/2)

(c) Type 2: CKP10 = 0, DAP10 = 1



(d) Type 4: CKP10 = 1, DAP10 = 1



The first bit is directly latched by the SOTB10 register at the falling edge of the write signal of the SOTB10 register or the read signal of the SIO10 register, and output from the SO10 pin via an output selector. Then, the value of the SOTB10 register is transferred to the SIO10 register at the next falling (or rising) edge of $\overline{\text{SCK10}}$, and shifted one bit. At the same time, the first bit of the receive data is stored in the SIO10 register via the SI10 pin.

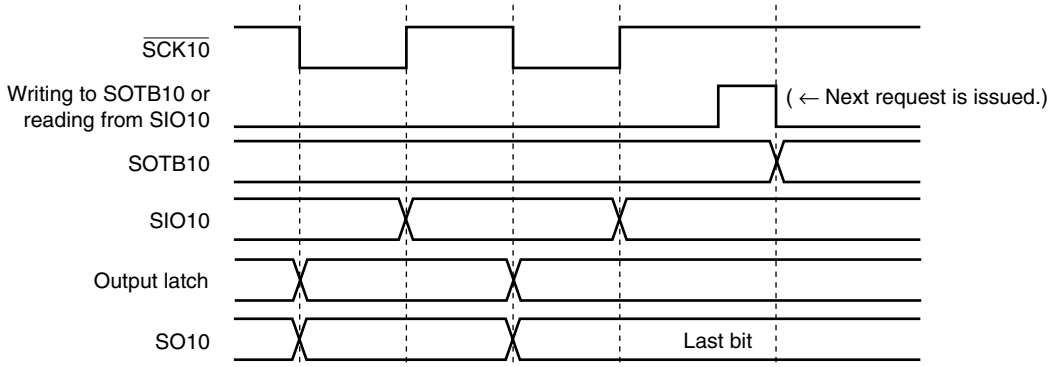
The second and subsequent bits are latched by the SIO10 register to the output latch at the next rising (or falling) edge of $\overline{\text{SCK10}}$, and the data is output from the SO10 pin.

(4) Output value of SO10 pin (last bit)

After communication has been completed, the SO10 pin holds the output value of the last bit.

Figure 11-8. Output Value of SO10 Pin (Last Bit) (1/2)

(a) Type 1: CKP10 = 0, DAP10 = 0



(b) Type 3: CKP10 = 1, DAP10 = 0

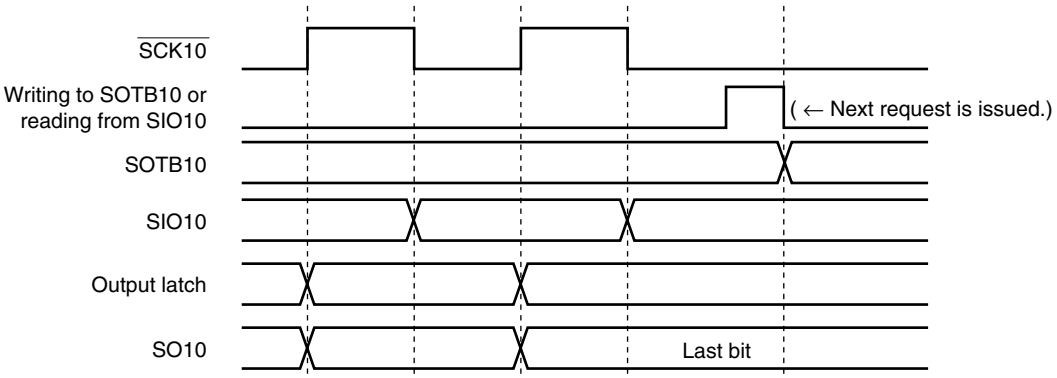
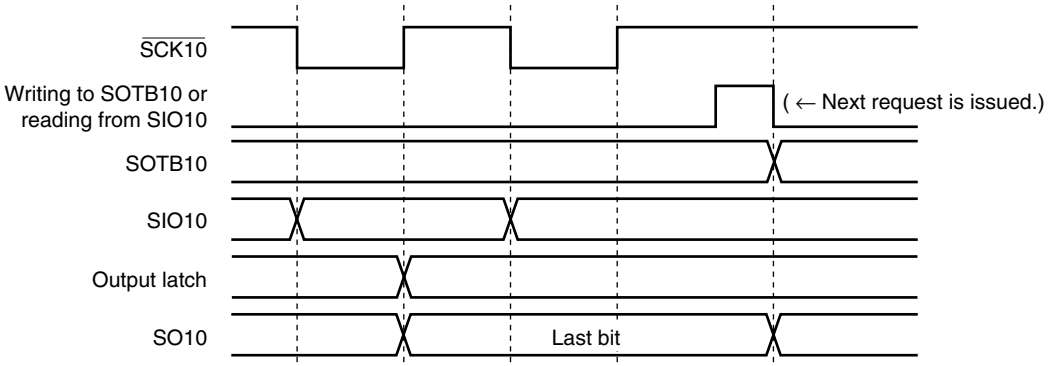
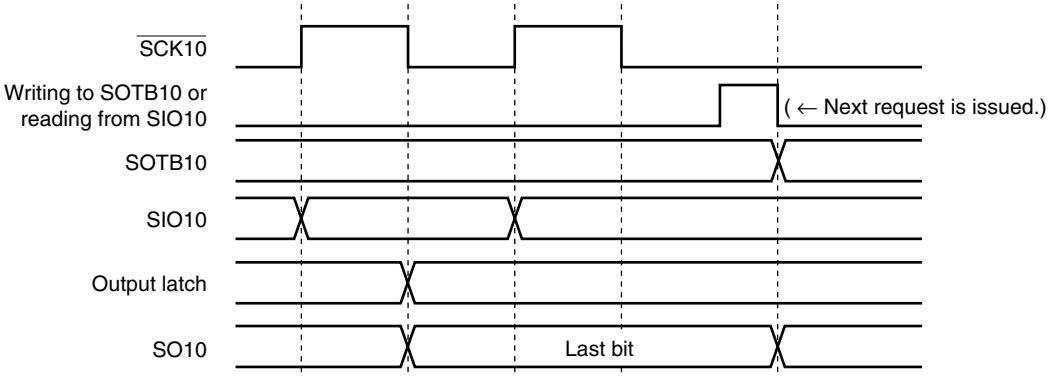


Figure 11-8. Output Value of SO10 Pin (Last Bit) (2/2)

(c) Type 2: CKP10 = 0, DAP10 = 1



(d) Type 4: CKP10 = 1, DAP10 = 1



(5) SO10 output (see (a) in Figure 11-1)

The status of the SO10 output is as follows if bit 7 (CSIE10) of serial operation mode register 10 (CSIM10) is cleared to 0.

Table 11-3. SO10 Output Status

TRMD10	DAP10	DIR10	SO10 Output[Note 1]
TRMD10 = 0[Note 2]	–	–	Outputs low level[Note 2]
TRMD10 = 1	DAP10 = 0	–	Value of SO10 latch (low-level output)
	DAP10 = 1	DIR10 = 0	Value of bit 7 of SOTB10
		DIR10 = 1	Value of bit 0 of SOTB10

Notes 1. The actual output of the SO10/P12 pin is determined according to PM12 and P12, as well as the SO10 output.

2. Status after reset

Caution If a value is written to TRMD10, DAP10, and DIR10, the output value of SO10 changes.

11.5 Caution for Serial Interface CSI10

(1) Standby mode

To resume the operation from the standby status, clear bit 2 (CSIIF10) of interrupt request flag register 0H (IF0H) to 0.

CHAPTER 12 USB FUNCTION CONTROLLER USBF

The μPD78F0730 has an internal USB function controller (USBF) conforming to the Universal Serial Bus Specification.

12.1 Overview

- Conforms to the Universal Serial Bus Specification.
- Supports 12 Mbps (full-speed) transfer
- Endpoint for transfer incorporated

Endpoint Name	FIFO Size (Bytes)	Transfer Type	Remark
Endpoint0 Read	64	Control transfer	–
Endpoint0 Write	64	Control transfer	–
Endpoint1	64 × 2	Bulk 1 transfer (IN)	2-buffer configuration
Endpoint2	64 × 2	Bulk 1 transfer (OUT)	2-buffer configuration

- Clock: f_{USB} = 48 MHz
  (The clock source is selectable. see **5.3 (8) PLL control register**.)

Caution When using the USB function, be sure to supply clock to the USB function controller (see 5.6.4 Example of controlling USB clock). Don't access the registers related to the USB function while clock isn't supplied to the USB function controller.

12.2 Configuration

USB function controller USBF includes the following hardware.

Table 12-1. Configuration of USB Function Controller USBF (1/2)

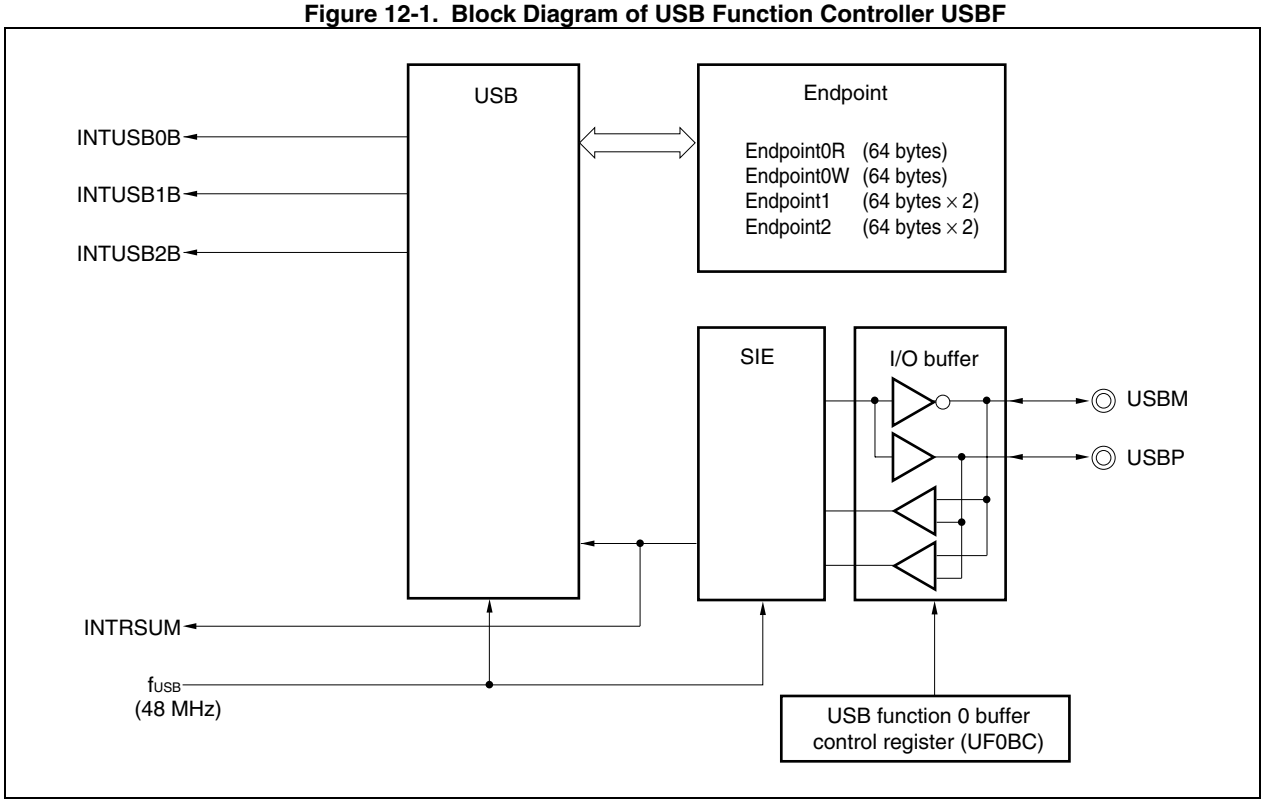
Item	Configuration
USB port pins	USBP (+), USBM (–)
Control registers	UF0 EP0NAK register (UF0E0N) UF0 EP0NAKALL register (UF0E0NA) UF0 EPNAK register (UF0EN) UF0 EPNAK mask register (UF0ENM) UF0 SNDSIE register (UF0SDS) UF0 CLR request register (UF0CLR) UF0 SET request register (UF0SET) UF0 EP status 0 register (UF0EPS0) UF0 EP status 1 register (UF0EPS1) UF0 EP status 2 register (UF0EPS2) UF0 INT status 0 register (UF0IS0) UF0 INT status 1 register (UF0IS1) UF0 INT status 2 register (UF0IS2) UF0 INT status 3 register (UF0IS3) UF0 INT status 4 register (UF0IS4) UF0 INT mask 0 register (UF0IM0) UF0 INT mask 1 register (UF0IM1) UF0 INT mask 2 register (UF0IM2) UF0 INT mask 3 register (UF0IM3) UF0 INT mask 4 register (UF0IM4) UF0 INT clear 0 register (UF0IC0) UF0 INT clear 1 register (UF0IC1) UF0 INT clear 2 register (UF0IC2) UF0 INT clear 3 register (UF0IC3) UF0 INT clear 4 register (UF0IC4) UF0 FIFO clear 0 register (UF0FIC0) UF0 FIFO clear 1 register (UF0FIC1) UF0 data end register (UF0DEND) UF0 GPR register (UF0GPR) UF0 mode control register (UF0MODC) UF0 mode status register (UF0MODS) UF0 active interface number register (UF0AIFN) UF0 active alternative setting register (UF0AAS) UF0 alternative setting status register (UF0ASS) UF0 endpoint 1 interface mapping register (UF0E1IM) UF0 endpoint 2 interface mapping register (UF0E2IM)

Table 12-1. Configuration of USB Function Controller USBF (2/2)

Item	Configuration
Data hold registers	UF0 EP0 read register (UF0E0R) UF0 EP0 length register (UF0E0L) UF0 EP0 setup register (UF0E0ST) UF0 EP0 write register (UF0E0W) UF0 bulk out 1 register (UF0BO1) UF0 bulk out 1 length register (UF0BO1L) UF0 bulk in 1 register (UF0BI1)
Request data registers	UF0 devise status register L (UF0DSTL) UF0 EP0 status register L (UF0E0SL) UF0 EP1 status register L (UF0E1SL) UF0 EP2 status register L (UF0E2SL) UF0 address register (UF0ADRS) UF0 configuration register (UF0CNF) UF0 interface 0 register (UF0IF0) UF0 interface 1 to 4 registers (UF0IF1 to UF0IF4) UF0 descriptor length register (UF0DSCL) UF0 devise descriptor registers 0 to 17 (UF0DD0 to UF0DD17) UF0 configuration/interface/endpoint descriptor registers 0 to 255 (UF0CIE0 to UF0CIE255)
Peripheral control register	USB function 0 buffer control register (UF0BC)

Figure 12-1 shows the block diagram.

Figure 12-1. Block Diagram of USB Function Controller USBF



12.3 Requests

12.3.1 Automatic requests

(1) Decode

The following tables show the request formats and correspondence between requests and decoded values.

Table 12-2. Request Format

Offset	Field Name	
0	bmRequestType	
1	bRequest	
2	wValue	Lower side
3		Higher side
4	wIndex	Lower side
5		Higher side
6	wLength	Lower side
7		Higher side

Table 12-3. Correspondence Between Requests and Decoded Values

Request \ Offset	Decoded Value								Response			Data Stage
	bmRequestType	bRequest	wValue		wIndex		wLength		Df	Ad	Cf	
	0	1	3	2	5	4	7	6				
GET_INTERFACE	81H	0AH	00H	00H	00H	0nH	00H	01H	STALL	STALL	ACK NAK	√
GET_CONFIGURATION	80H	08H	00H	00H	00H	00H	00H	01H	ACK NAK	ACK NAK	ACK NAK	√
GET_DESCRIPTOR Device	80H	06H	01H	00H	00H	00H	XXH	XXH[Note 1]	ACK NAK	ACK NAK	ACK NAK	√
GET_DESCRIPTOR Configuration	80H	06H	02H	00H	00H	00H	XXH	XXH[Note 1]	ACK NAK	ACK NAK	ACK NAK	√
GET_STATUS Device	80H	00H	00H	00H	00H	00H	00H	02H	ACK NAK	ACK NAK	ACK NAK	√
GET_STATUS Endpoint 0	82H	00H	00H	00H	00H	00H 80H	00H	02H	ACK NAK	ACK NAK	ACK NAK	√
GET_STATUS Endpoint X	82H	00H	00H	00H	00H	$$H	00H	02H	STALL	STALL	ACK NAK	√
CLEAR_FEATURE Device[Note 2]	00H	01H	00H	01H	00H	00H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×
CLEAR_FEATURE Endpoint 0[Note 2]	02H	01H	00H	00H	00H	00H 80H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×
CLEAR_FEATURE Endpoint X[Note 2]	02H	01H	00H	00H	00H	$$H	00H	00H	STALL	STALL	ACK NAK	×
SET_FEATURE Device[Note 3]	00H	03H	00H	01H	00H	00H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×
SET_FEATURE Endpoint 0[Note 3]	02H	03H	00H	00H	00H	00H 80H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×
SET_FEATURE Endpoint X[Note 3]	02H	03H	00H	00H	00H	$$H	00H	00H	STALL	STALL	ACK NAK	×
SET_INTERFACE	01H	0BH	00H	0#H	00H	0?H	00H	00H	STALL	STALL	ACK NAK	×
SET_CONFIGURATION[Note 4]	00H	09H	00H	00H 01H	00H	00H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×
SET_ADDRESS	00H	05H	XXH	XXH	00H	00H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×

Remark √: Data stage is provided
×: Data stage is not provided

Notes 1. If the wLength value is less than the prepared value, the wLength value is returned; if the wLength value is greater than the prepared value, the prepared value is returned.

2. The CLEAR_FEATURE request clears UF0 device status register L (UF0DSTL) and UF0 EPn status register L (UF0EnSL) (n = 0 to 2) when ACK is received in the status stage.

Notes 3. The SET_FEATURE request sets the UF0 device status register L (UF0DSTL) and UF0 EPn status register L (UF0EnSL) (n = 0 to 2) when ACK is received in the status stage. If the E0HALT bit of the UF0E0SL register is set, a STALL response is made in the status stage or data stage of control transfer for a request other than the GET_STATUS Endpoint0 request, SET_FEATURE Endpoint0 request, and a request generated by the CPUDEC interrupt request, until the CLEAR_FEATURE Endpoint0 request is received. A STALL response to an unsupported request does not set the E0HALT bit of the UF0E0SL register to 1, and the STALL response is cleared as soon as the next SETUP token has been received.

4. If the wValue is not the default value, an automatic STALL response is made.

Cautions 1. The sequence of control transfer defined by the Universal Serial Bus Specification is not satisfied under the following conditions. The operation is not guaranteed under these conditions.

- **If an IN/OUT token is suddenly received without a SETUP stage**
- **If DATA PID1 is sent in the data phase of the SETUP stage**
- **If a token of 128 addresses or more is received**
- **If the request data transmitted in the SETUP stage is of less than 8 bytes**

2. An ACK response is made even when the host transmits data other than a Null packet in the status stage.

3. If the wLength value is 00H during control transfer (read) of FW processing, a Null packet is automatically transmitted for control transfer (without data). The FW request does not automatically transmit a Null packet.

Remarks 1. Df: Default state, Ad: Addressed state, Cf: Configured state

2. n = 0 to 4

It is determined by the setting of the UF0 active interface number register (UF0AIFN) whether a request with Interface number 1 to 4 is correctly responded to, depending on whether the Interface number of the target is valid or not.

3. $$: Valid endpoint number including transfer direction

The valid endpoint is determined by the currently set Alternate Setting number (see **12.4.1 (33) UF0 active alternative setting register (UF0AAS)**, **(35) UF0 endpoint 1 interface mapping register (UF0E1IM)** to **(36) UF0 endpoint 2 interface mapping register (UF0E2IM)**).

4. ? and #: Value transmitted from host (information on Interface numbers 0 to 4)

It is determined by the UF0 active interface number register (UF0AIFN) and UF0 active alternative setting register (UF0AAS) whether an Alternate Setting request corresponding to each Interface number is correctly responded to or not, depending on whether the Interface number and Alternate Setting of the target are valid or not.

(2) Processing

The processing of an automatic request in the Default state, Addressed state, and Configured state is described below.

Remark Default state: State in which an operation is performed with the Default address
Addressed state: State after an address has been allocated
Configured state: State after SET_CONFIGURATION wValue = 1 has been correctly received

(a) CLEAR_FEATURE() request

A STALL response is made in the status stage if the CLEAR_FEATURE() request cannot be cleared, if FEATURE does not exist, or if the target is an interface or an endpoint that does not exist. A STALL response is also made if the wLength value is other than 0.

- Default state: The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for Endpoint0; otherwise a STALL response is made in the status stage.
- Addressed state: The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for Endpoint0; otherwise a STALL response is made in the status stage.
- Configured state: The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for an endpoint that exists; otherwise a STALL response is made in the status stage.

When the CLEAR_FEATURE() request has been correctly processed, the corresponding bit of the UF0 CLR request register (UF0CLR) is set to 1, the EnHALT bit of the UF0 EPn status register L (UF0EnSL) is cleared to 0, and an interrupt is issued (n = 0 to 2). If the CLEAR_FEATURE() request is received when the subject is an endpoint, the toggle bit (that controls switching between DATA0 and DATA1) of the corresponding endpoint is always re-set to DATA0.

(b) GET_CONFIGURATION() request

A STALL response is made in the data stage if any of wValue, wIndex, or wLength is other than the values shown in Table 12-3.

- Default state: The value stored in the UF0 configuration register (UF0CNF) is returned when the GET_CONFIGURATION() request has been received.
- Addressed state: The value stored in the UF0CNF register is returned when the GET_CONFIGURATION() request has been received.
- Configured state: The value stored in the UF0CNF register is returned when the GET_CONFIGURATION() request has been received.

(c) GET_DESCRIPTOR() request

If the subject descriptor has a length that is a multiple of wMaxPacketSize, a Null packet is returned to indicate the end of the data stage. If the length of the descriptor at this time is less than the wLength value, the entire descriptor is returned; if the length of the descriptor is greater than the wLength value, the descriptor up to the wLength value is returned.

- Default state: The value stored in UF0 device descriptor register n (UF0DDn) and UF0 configuration/interface/endpoint descriptor register m (UF0CIEm) is returned (n = 0 to 17, m = 0 to 255) when the GET_DESCRIPTOR() request has been received.
- Addressed state: The value stored in the UF0DDn register and UF0CIEm register is returned when the GET_DESCRIPTOR() request has been received.
- Configured state: The value stored in the UF0DDn register and UF0CIEm register is returned when the GET_DESCRIPTOR() request has been received.

A descriptor of up to 256 bytes can be stored in the UF0CIEm register. To return a descriptor of more than 256 bytes, set the CDCGDST bit of the UF0MODC register to 1 and process the GET_DESCRIPTOR() request by FW.

Store the value of the total number of bytes of the descriptor set by the UF0CIEm register – 1 in the UF0 descriptor length register (UF0DSCL). The transfer data is controlled by the value of this data + 1 and wLength.

(d) GET_INTERFACE() request

If either of wValue and wLength is other than that shown in Table 12-3, or if wIndex is other than that set by the UF0 active interface number register (UF0AIFN), a STALL response is made in the data stage.

- Default state: A STALL response is made in the data stage when the GET_INTERFACE() request has been received.
- Addressed state: A STALL response is made in the data stage when the GET_INTERFACE() request has been received.
- Configured state: The value stored in the UF0 interface n register (UF0IFn) corresponding to the wIndex value is returned (n = 0 to 4) when the GET_INTERFACE() request has been received.

(e) GET_STATUS() request

A STALL response is made in the data stage if any of wValue, wIndex, or wLength is other than the values shown in Table 12-3. A STALL response is also made in the data stage if the target is an interface or an endpoint that does not exist.

- Default state: The value stored in the target status register[Note] is returned only when the GET_STATUS() request has been received and when the request is for a device or Endpoint0; otherwise a STALL response is made in the data stage.
- Addressed state: The value stored in the target status register[Note] is returned only when the GET_STATUS() request has been received and when the request is for a device or Endpoint0; otherwise a STALL response is made in the data stage.
- Configured state: The value stored in the target status register[Note] is returned only when the GET_STATUS() request has been received and when the request is for a device or an endpoint that exists; otherwise a STALL response is made in the data stage.

Note The target status register is as follows.
- If the target is a device: UF0 device status register L (UF0DSTL)
- If the target is endpoint 0: UF0 EP0 status register L (UF0E0SL)
- If the target is endpoint n: UF0 EPn status register L (UF0EnSL) (n = 1 to 2)

(f) SET_ADDRESS() request

A STALL response is made in the status stage if either of wIndex or wLength is other than the values shown in Table 12-3. A STALL response is also made if the specified device address is greater than 127.

- Default state: The device enters the Addressed state and changes the USB Address value to be input to SIE into a specified address value if the specified address is other than 0 when the SET_ADDRESS() request has been received. If the specified address is 0, the device remains in the Default state.
- Addressed state: The device enters the Default state and returns the USB Address value to be input to SIE to the default address if the specified address is 0 when the SET_ADDRESS() request has been received. If the specified address is other than 0, the device remains in the Addressed state, and changes the USB Address value to be input to SIE into a specified new address value.
- Configured state: The device remains in the Configured state and returns the USB Address value to be input to SIE to the default address if the specified address is 0 when the SET_ADDRESS() request has been received. In this case, the endpoints other than endpoint 0 remain valid, and control transfer (IN), control transfer (OUT), bulk transfer and interrupt transfer for an endpoint other than endpoint 0 are also acknowledged. If the specified address is other than 0, the device remains in the Configured state and changes the USB Address value to be input to SIE into a specified new address value.

(g) SET_CONFIGURATION() request

If any of wValue, wIndex, or wLength is other than the values shown in Table 12-3, a STALL response is made in the status stage.

- Default state: The CONF bit of the UF0 mode status register (UF0MODS) and the UF0 configuration register (UF0CNF) are set to 1 if the specified configuration value is 1 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 0, the CONF bit of the UF0MODS register and UF0CNF register are cleared to 0. In other words, the device skips the Addressed state and moves to the Configured state in which it responds to the Default address.
- Addressed state: The CONF bit of the UF0MODS register and UF0CNF register are set to 1 and the device enters the Configured state if the specified configuration value is 1 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 0, the device remains in the Addressed state.
- Configured state: The CONF bit of the UF0MODS register and UF0CNF register are set to 1 and the device returns to the Addressed state if the specified configuration value is 0 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 1, the device remains in the Configured state.

If the SET_CONFIGURATION() request has been correctly processed, the target bit of the UF0 SET request register (UF0SET) is set to 1, and an interrupt is issued. All Halt Features are cleared after the SET_CONFIGURATION() request has been completed even if the specified configuration value is the same as the current configuration value. If the SET_CONFIGURATION() request has been correctly processed, the data toggle of all endpoints is always initialized to DATA0 again (it is defined that the default status, Alternative Setting 0, is set from when the SET_CONFIGURATION request is received to when the SET_INTERFACE request is received).

(h) SET_FEATURE() request

A STALL response is made in the status stage if the SET_FEATURE() request is for a Feature that cannot be set or does not exist, or if the target is an interface or an endpoint that does not exist. A STALL response is also made if the wLength value is other than 0.

- Default state: The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or Endpoint0; otherwise a STALL response is made in the status stage.
- Addressed state: The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or Endpoint0; otherwise a STALL response is made in the status stage.
- Configured state: The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or an endpoint that exists; otherwise a STALL response is made in the status stage.

When the SET_FEATURE() request has been correctly processed, the target bit of the UF0 SET request register (UF0SET) and the EnHALT bit of the UF0 EPn status register L (UF0EnSL) are set to 1, and an interrupt is issued (n = 0 to 2).

(i) SET_INTERFACE() request

If wLength is other than the values shown in Table 12-3, if wIndex is other than the value set to the UF0 active interface number register (UF0AIFN), or if wValue is other than the value set to the UF0 active alternative setting register (UF0AAS), a STALL response is made in the status stage.

- Default state: A STALL response is made in the status stage when the SET_INTERFACE() request has been received.
- Addressed state: A STALL response is made in the status stage when the SET_INTERFACE() request has been received.
- Configured state: Null packet is transmitted in the status stage when the SET_INTERFACE() request has been received.

When the SET_INTERFACE() request has been correctly processed, an interrupt is issued. All the Halt Features of the endpoint linked to the target Interface are cleared after the SET_INTERFACE() request has been cleared. The data toggle of all the endpoints related to the target Interface number is always initialized again to DATA0. When the currently selected Alternative Setting is to be changed by correctly processing the SET_INTERFACE() request, the FIFO of the endpoint that is affected is completely cleared, and all the related interrupt sources are also initialized.

When the SET_INTERFACE() request has been completed, the FIFO of all the endpoints linked to the target Interface are cleared. At the same time, Halt Feature and Data PID are initialized, and the related UF0 INT status n register (UF0ISn) is cleared to 0 (n = 0 to 4). (Only Halt Feature and Data PID are cleared when the SET_CONFIGURATION request has been completed.)

12.3.2 Other requests

(1) Response and processing

The following table shows how other requests are responded to and processed.

Table 12-4. Response and Processing of Other Requests

Request	Response and Processing
GET_DESCRIPTOR String	Generation of CPUDEC interrupt request
GET_STATUS Interface	Automatic STALL response
CLEAR_FEATURE Interface	Automatic STALL response
SET_FEATURE Interface	Automatic STALL response
all SET_DESCRIPTOR	Generation of CPUDEC interrupt request
All other requests	Generation of CPUDEC interrupt request

12.4 Register Configuration

12.4.1 Control registers

(1) UF0 EP0NAK register (UF0E0N)

This register controls NAK of Endpoint0 (except an automatically executed request).

This register can be read or written in 8-bit units (however, bit 0 can only be read).

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set. If it is necessary to read the status correctly, therefore, separate a write signal that accesses the UF0FIC0 and UF0FIC1 registers from a read signal that accesses the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

While NAK is being transmitted to Endpoint0 Read and Endpoint2, a write access to the EP0NKR bit is ignored.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0N	0	0	0	0	0	0	EP0NKR	EP0NKW	FF60H	00H

Bit position	Bit name	Function
1	EP0NKR	This bit controls NAK to the OUT token to Endpoint0 (except an automatically executed request). It is automatically set to 1 by hardware when Endpoint0 has correctly received data. It is also cleared to 0 by hardware when the data of the UF0E0R register has been read by FW (counter value = 0). 1: Transmit NAK. 0: Do not transmit NAK (default value). Set this bit to 1 by FW when data should not be received from the USB bus for some reason even when USBF is ready for receiving data. In this case, USBF continues transmitting NAK until this bit is cleared to 0 by FW. This bit is also cleared to 0 as soon as the UF0E0R register has been cleared.
0	EP0NKW	This bit indicates how NAK to the IN token to Endpoint0 is controlled (except an automatically executed request). This bit is automatically cleared to 0 by hardware when the data of Endpoint0 is transmitted and the host correctly receives the transmitted data. The data of the UF0E0W register is retained until this bit is cleared. Therefore, it is not necessary to rewrite this bit even in the case of a retransmission request that is made if the host could not receive data correctly. To send a short packet, be sure to set the E0DED bit of the UF0DEND register to 1. This bit is automatically set to 1 when the FIFO is full. As soon as the E0DED bit of the UF0DEND register is set to 1, the EP0NKW bit is automatically set to 1 at the same time. 1: Do not transmit NAK. 0: Transmit NAK (default value). If control transfer enters the status stage while ACK cannot be correctly received in the data stage, this bit is cleared to 0 as soon as the UF0E0W register is cleared. This bit is also cleared to 0 when UF0E0W is cleared by FW.

Next, the procedure of a SETUP transaction that uses IN/OUT tokens is explained below.

(a) When IN token is used (except a request automatically executed by hardware)

FW should be used to clear the PROT bit of the UF0IS1 register to 0 after receiving the CPUDEC interrupt and before reading data from the UF0E0ST register. Next, perform processing in accordance with the request and, if it is necessary to return data by an IN token, write data to the UF0E0W register. Confirm that the PROT bit of the UF0IS1 register is 0 after writing has been completed, and set the E0DED bit of the UF0DEND register to 1. The hardware sends out data at the first IN token after the EP0NKW bit has been set to 1. If the PROT bit of the UF0IS1 register is 1, it indicates that a SETUP transaction has occurred again before completion of control transfer. In this case, clear the PROT bit of the UF0IS1 register to 0 by clearing the PROTC bit of the UF0IC1 register to 0, and then read data from the UF0E0ST register again. A request received later can be read.

(b) When OUT token is used (except a request automatically executed by hardware)

FW should be used to clear the PROT bit of the UF0IS1 register after receiving the CPUDEC interrupt and before reading data from the UF0E0ST register. Confirm that the PROT bit of the UF0IS1 register is 0 before reading data from the UF0E0R register. If the PROT bit is 1, it means that invalid data is retained. Clear the FIFO by FW (the EP0NKR bit is automatically cleared to 0). If the PROT bit of the UF0IS1 register is 0, read the data of the UF0E0L register and read as many data from the UF0E0R register as set. When reading data from the UF0E0R register has been completed (when the counter of the UF0E0R register has been cleared to 0), the hardware automatically clears the EP0NKR bit to 0.

(2) UF0 EP0NAKALL register (UF0E0NA)

This register controls NAK to all the requests of Endpoint0. It is also valid for automatically executed requests.

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

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0NA	0	0	0	0	0	0	0	EP0NKA	FF61H	00H

Bit position	Bit name	Function
0	EP0NKA	This bit controls NAK to a transaction other than a SETUP transaction to Endpoint0 (including an automatically executed request). This bit is manipulated by FW. 1: Transmit NAK. 0: Do not transmit NAK (default value). This register is used to prevent a conflict between a write access by FW and a read access from SIE when the data used for an automatically executed request is to be changed. It postpones reflecting a write access on this bit from FW while an access from SIE is being made. Before rewriting the request data register from FW, confirm that this bit has been correctly set to 1. Setting this bit to 1 is reflected only in the following cases. • Immediately after USBF has been reset and a SETUP token has never been received • Immediately after reception of Bus Reset and a SETUP token has never been received • PID of a SETUP token has been detected • The stage has been changed to the status stage Clearing this bit to 0 is reflected immediately, except while an IN token is being received and a NAK response is being made. Setting the EP0NKA bit to 1 is reflected in the above four cases during Endpoint0 transfer, but it is reflected immediately after data has been written to the bit while Endpoint0 is transferring no data.

(3) UF0 EPNAK register (UF0EN)

This register controls NAK of endpoints other than Endpoint0.

This register can be read or written in 8-bit units (however, bit 0 can only be read).

The BKO2NK bit can be written only when the BKO2NKM bit of the UF0ENM register is 1 and the BKO1NK bit can be written only when the BKO1NKM bit of the UF0ENM register is 1.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 2) and the current setting of the interface.

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set. If it is necessary to read the status correctly, therefore, separate a write signal that accesses the UF0FIC0 and UF0FIC1 registers from a read signal that accesses the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

While NAK is being transmitted to Endpoint0 Read and Endpoint2, a write access to the BKO1NK and BKO2NK bits is ignored.

Be sure to clear bits 7 to 3 and 1. If these bits are set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
UF0EN	0	0	0	0	0	BKO1NK	0	BKI1NK	FF62H	00H

Bit position	Bit name	Function
2	BKO1NK	This bit controls NAK to Endpoint2 (bulk 1 transfer (OUT)). 1: Transmit NAK. 0: Do not transmit NAK (default value). This bit is set to 1 only when the FIFO connected to the SIE side of the UF0BO1 register (64-byte FIFO of bank configuration) cannot receive data. It is cleared to 0 when a toggle operation is performed. The bank is changed (toggle operation) when the following conditions are satisfied. • Data correctly received is stored in the FIFO connected to the SIE side. • The value of the FIFO counter connected to the CPU side is 0 (completion of reading). FW should be used to read data of the UF0BO1L register when it has received the BKO1DT interrupt request and read as many data from the UF0BO1 register as the value of that data. To not receive data from the USB bus for some reason even if USBF is ready to receive data, set this bit to 1 by FW. In this case, USBF keeps transmitting NAK until the FW clears this bit to 0. This bit is also cleared to 0 as soon as the UF0BO1 register has been cleared.

(2/2)

Bit position	Bit name	Function
0	BKI1NK	This bit controls NAK to Endpoint1 (bulk 1 transfer (IN)). 1: Do not transmit NAK. 0: Transmit NAK (default value). This bit is cleared to 0 only when the FIFO connected to the SIE side of the UF0BI1 register (64-byte FIFO of bank configuration) cannot receive data. It is set to 1 when a toggle operation is performed (the data of the UF0BI1 register is retained until transmission has been correctly completed). The bank is changed (toggle operation) when the following conditions are satisfied. • Data is correctly written to the FIFO connected to the CPU bus side (writing has been completed and the FIFO is full or the UF0DEND register is set). • The value of the FIFO counter connected to the SIE side is 0. This bit is automatically set to 1 and data transmission is started when the FIFO on the CPU side becomes full and a FIFO toggle operation is performed as a result of writing data to the FIFO. To send a short packet that does not make the FIFO on the CPU side full, set the BKI1DED bit to 1 after completing writing data. When the BKI1DED bit is set to 1, a toggle operation is performed and at the same time, this bit is automatically set to 1. This bit is also cleared to 0 as soon as the UF0BI1 register has been cleared.

(4) UF0 EPNAK mask register (UF0ENM)

This register controls masking a write access to the UF0EN register.

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

Be sure to clear bits 7 to 3, 1, and 0. If these bits are set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
UF0ENM	0	0	0	0	0	BKO1NKM	0	0	FF63H	00H

Bit position	Bit name	Function
2	BKO1NKM	This bit specifies whether a write access to bit 2 (BKO1NK) of the UF0EN register is masked or not. 1: Do not mask. 0: Mask (default value).

(5) UF0 SNDSIE register (UF0SDS)

This register performs manipulation such as no handshake. It can directly manipulate the pins of SIE.

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

Be sure to clear bit 2. If it is set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
UF0SDS	0	0	0	0	SNDSTL	0	0	RSUMIN	FF64H	00H

Bit position	Bit name	Function
3	SNDSTL	This bit makes Endpoint0 issue a STALL handshake. Setting this bit to 1 if a request for CPUDEC processing is not supported by the system results in a STALL handshake response. If an unsupported wValue is sent by the SET_CONFIGURATION or SET_INTERFACE request, the hardware sets this bit to 1. If a problem occurs in Endpoint0 due to overrun of an automatically executed request, this bit is also set to 1. However, the E0HALT bit of the UF0E0SL register is not set to 1. 1: Respond with STALL handshake. 0: Do not respond with STALL handshake (default value). This bit is cleared to 0 and the handshake response to the bus is other than STALL when the next SETUP token is received. To set the SNDSTL bit to 1 by FW, do not write data to the UF0E0W register. Depending on the timing of setting this bit, the STALL response is not made in time, and it may be made to the next transfer after a NAK response has been made. Setting this bit is valid only while an FW-executed request is under execution when this bit is set to 1. It is automatically cleared to 0 when the next SETUP token is received. **Remark** The SNDSTL bit is valid only for an FW-executed request.
0	RSUMIN	This bit outputs the Resume signal onto the USB bus. Writing this bit is invalid unless the RMWK bit of the UF0DSTL register is set to 1. 1: Generate the Resume signal. 0: Do not generate the Resume signal (default value). While this bit is set to 1, the Resume signal continues to be generated. Clear this bit to 0 by FW after a specific time has elapsed. Because the signal is internally sampled at the clock, the operation is guaranteed only while CLK is supplied. Care must be exercised when CLK of the system is stopped.

(6) UF0 CLR request register (UF0CLR)

This register indicates the target of the received CLEAR_FEATURE request.

This register is read-only, in 8-bit units.

This register is meaningful only when an interrupt request is generated. Each bit is set to 1 after completion of the status stage, and automatically cleared to 0 when this register is read.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 2) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0CLR	0	0	0	0	CLREP2	CLREP1	CLREP0	CLRDEV	FF65H	00H

Bit position	Bit name	Function
3 to 1	CLREPn	These bits indicate that a CLEAR_FEATURE Endpoint n request is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)
0	CLRDEV	This bit indicates that a CLEAR_FEATURE Device request is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)

Remark n = 2 to 0

(7) UF0 SET request register (UF0SET)

This register indicates the target of the automatically processed SET_XXXX (except SET_INTERFACE) request.

This register is read-only, in 8-bit units.

This register is meaningful only when an interrupt request is generated. Each bit is set to 1 after completion of the status stage, and automatically cleared to 0 when this register is read.

	7	6	5	4	3	2	1	0	Address	After reset
UF0SET	SETCON	0	0	0	0	SETEP	0	SETDEV	FF66H	00H

Bit position	Bit name	Function
7	SETCON	This bit indicates that a SET_CONFIGURATION request is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)
2	SETEP	This bit indicates that a SET_FEATURE Endpoint n request (n = 0 to 2) is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)
0	SETDEV	This bit indicates that a SET_FEATURE Device request is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)

(8) UF0 EP status 0 register (UF0EPS0)

This register indicates the USB bus status and the presence or absence of register data.

This register is read-only, in 8-bit units.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 2) and the current setting of the interface.

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set. If it is necessary to read the status correctly, therefore, separate writing to the UF0FIC0 and UF0FIC1 registers from reading from the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

	7	6	5	4	3	2	1	0	Address	After reset
UF0EPS0	0	0	0	BKOUT1	0	BKIN1	EP0W	EP0R	FF67H	00H

Bit position	Bit name	Function
4	BKOUT1	This bit indicates that data is in the UF0BO1 register (FIFO) connected to the CPU side. When the FIFO configuring the UF0BO1 register is toggled, this bit is automatically set to 1 by hardware. It is automatically cleared to 0 by hardware when reading the UF0BO1 register (FIFO) connected to the CPU side has been completed (counter value = 0). It is not set to 1 when Null data is received (toggling the FIFO does not take place either). 1: Data is in the register. 0: No data is in the register (default value).
2	BKIN1	This bit indicates that data is in the UF0BI1 register (FIFO) connected to the CPU side. By setting the BKI1DED bit of the UF0DEND register to 1, the status in which data is in the UF0BI1 register can be created even if data is not written to the register (Null data transmission). As soon as the BKI1DED bit of the UF0DEND register has been set to 1 while the counter of the UF0BI1 register is 0, this bit is set to 1 by hardware. It is cleared to 0 when a toggle operation is performed. 1: Data is in the register. 0: No data is in the register (default value).
1	EP0W	This bit indicates that data is in the UF0E0W register (FIFO). By setting the E0DED bit of the UF0DEND register to 1, the status in which data is in the UF0E0W register can be created even if data is not written to the register (Null data transmission). As soon as the E0DED bit of the UF0DEND register is set to 1 even when the counter of the UF0E0W register is 0, this bit is set to 1 by hardware. It is cleared to 0 after correct transmission. 1: Data is in the register. 0: No data is in the register (default value).
0	EP0R	This bit indicates that data is in the UF0E0R register (FIFO). It is automatically cleared to 0 by hardware when reading the UF0E0R register (FIFO) has been completed (counter value = 0). It is not set to 1 if Null data is received. 1: Data is in the register. 0: No data is in the register (default value).

(9) UF0 EP status 1 register (UF0EPS1)

This register indicates the USB bus status and the presence or absence of register data.

This register is read-only, in 8-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
UF0EPS1	RSUM	0	0	0	0	0	0	0	FF68H	00H

Bit position	Bit name	Function
7	RSUM	This bit indicates that the USB bus is in the Resume status. This bit is meaningful only when an interrupt request is generated. 1: Suspend status 0: Resume status (default value) Because sampling is internally performed with the clock, the operation is guaranteed only when CLK is supplied. Care must be exercised when CLK of the system is stopped. The INTRSUM signal of SIE operates even when CLK is stopped. It can therefore be supported by making the RSUMIF bit of interrupt request flag register 1L (IF1L) valid or lowering the frequency of CLK to the USBF. This bit is automatically cleared to 0 when it is read.

(10) UF0 EP status 2 register (UF0EPS2)

This register indicates the USB bus status and the presence or absence of register data.

This register is read-only, in 8-bit units.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 2) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0EPS2	0	0	0	0	0	HALT2	HALT1	HALT0	FF69H	00H

Bit position	Bit name	Function
2 to 0	HALTn	These bits indicate that Endpoint n is currently stalled. These bits are set to 1 when a stall condition, such as occurrence of an overrun and reception of an undefined request, is satisfied. These bits are automatically set to 1 by hardware. 1: Endpoint is stalled. 0: Endpoint is not stalled (default value). The SNDSTL bit is set to 1 as soon as the HALT0 bit has been set to 1 as a result of occurrence of an overrun or reception of an undefined request. If the next SETUP token is received in this status, the SNDSTL bit is cleared to 0 and, therefore, the HALT0 bit is also cleared to 0. If Endpoint0 is stalled by the SET_FEATURE Endpoint0 request, this bit is not cleared to 0 until the CLEAR_FEATURE Endpoint0 request is received or Halt Feature is cleared by FW. If the GET_STATUS Endpoint0, CLEAR_FEATURE Endpoint0, or SET_FEATURE Endpoint0 request is received, or if a request to be processed by FW is received due to the CPUDEC interrupt request, the HALT0 bit is masked and cleared to 0, until the next SETUP token is received. The HALTn bit is not cleared to 0 until Endpoint n receives the CLEAR_FEATURE Endpoint request, Halt Feature is cleared by the SET_INTERFACE or SET_CONFIGURATION request to the interface to which the endpoint is linked, or Halt Feature is cleared by FW. When the SET_INTERFACE or SET_CONFIGURATION request is correctly processed, the Halt Feature of all the target endpoints, except Endpoint0, is cleared after the request has been processed, even if the wValue is the same as the currently set value, and these bits are also cleared to 0. Halt Feature of Endpoint0 cannot be cleared if it is set because the STALL response is made in response to the SET_INTERFACE and SET_CONFIGURATION requests.

Remark n = 2 to 0

(11) UF0 INT status 0 register (UF0IS0)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB0B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSB0B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC0 register.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS0	BUSRST	RSUSPD	0	0	0	SETRQ	CLRRQ	EPHALT	FF27H	00H

Bit position	Bit name	Function
7	BUSRST	This bit indicates that Bus Reset has occurred. 1: Bus Reset has occurred (interrupt request is generated). 0: Not Bus Reset status (default value)
6	RSUSPD	This bit indicates that the Resume or Suspend status has occurred. Reference bit 7 of the UF0EPS1 register by FW. 1: Resume or Suspend status has occurred (interrupt request is generated). 0: Resume or Suspend status has not occurred (default value).
2	SETRQ	This bit indicates that the SET_XXXX request to be automatically processed has been received and automatically processed (XXXX = CONFIGURATION or FEATURE). 1: SET_XXXX request to be automatically processed has been received (interrupt request is generated). 0: SET_XXXX request to be automatically processed has not been received (default value). This bit is set to 1 after completion of the status stage. Reference the UF0SET register to identify what is the target of the request. This bit is not automatically cleared to 0 even if the UF0SET register is read by FW. The EPHALT bit is also set to 1 when the SET_FEATURE Endpoint request has been received.
1	CLRRQ	This bit indicates that the CLEAR_FEATURE request has been received and automatically processed. 1: CLEAR_FEATURE request has been received (interrupt request is generated). 0: CLEAR_FEATURE request has not been received (default value). This bit is set to 1 after completion of the status stage. Reference the UF0CLR register to identify what is the target of the request. This bit is not automatically cleared to 0 even if the UF0CLR register is read by FW.

(2/2)

Bit position	Bit name	Function
0	EPHALT	This bit indicates that an endpoint has stalled. 1: Endpoint has stalled (interrupt request is generated). 0: Endpoint has not stalled (default value). This bit is also set to 1 when an endpoint has stalled by setting FW. Identify the endpoint that has stalled, by referencing the UF0EPS2 register. This bit is not automatically cleared to 0 even when the CLEAR_FEATURE Endpoint, SET_INTERFACE, or SET_CONFIGURATION request is received. It is not automatically cleared to 0, either, if the next SETUP token is received in case of overrun of Endpoint0. **Caution Even if Halt Feature of Endpoint0 is set and this interrupt request is generated, bit 0 of the UF0EPS2 register is masked and cleared to 0 between when a SET_FEATURE Endpoint0, CLEAR_FEATURE Endpoint0, or GET_STATUS Endpoint0 request, or FW-processed request is received and when a SETUP token other than the above is received.**

(12) UF0 INT status 1 register (UF0IS1)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB0B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSB0B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC1 register. However, the SUCES and STG bits of the UF0IS1 register are automatically cleared to 0 when the next SETUP token has been received.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS1	0	E0IN	E0INDT	E0ODT	SUCES	STG	PROT	CPU DEC	FF28H	00H

Bit position	Bit name	Function
6	E0IN	This bit indicates that an IN token for Endpoint0 has been received and that the hardware has automatically transmitted NAK. 1: IN token is received and NAK is transmitted (interrupt request is generated). 0: IN token is not received (default value).
5	E0INDT	This bit indicates that data has been correctly transmitted from the UF0E0W register. 1: Transmission from UF0E0W register is completed (interrupt request is generated). 0: Transmission from UF0E0W register is not completed (default value). Data is transmitted in synchronization with the IN token next to the one that set the EP0NKW bit of the UF0E0N register to 1. This bit is automatically set to 1 by hardware when the host correctly receives that data. It is also set to 1 even if the data is a Null packet. This bit is automatically cleared to 0 by hardware when the first write access is made to the UF0E0W register.
4	E0ODT	This bit indicates that data has been correctly received in the UF0E0R register. 1: Data is in UF0E0R register (interrupt request is generated). 0: Data is not in UF0E0R register (default value). This bit is automatically set to 1 by hardware when data has been correctly received. At the same time, EP0R bit of the UF0EPS0 register is also set to 1. If a Null packet has been received, this bit is not set to 1. It is automatically cleared to 0 by hardware when the FW reads the UF0E0R register and the value of the UF0E0L register becomes 0.
3	SUCES	This bit indicates that either an FW-processed or hardware-processed request has been received and that the status stage has been correctly completed. 1: Control transfer has been correctly processed (interrupt request is generated). 0: Control transfer has not been processed correctly (default value). This bit is set to 1 upon completion of the status stage. It is automatically cleared to 0 by hardware when the next SETUP token is received. This bit is also set to 1 when data with Data PID of 0 (Null data) is received in the status stage of control transfer.

(2/2)

Bit position	Bit name	Function
2	STG	This bit is set to 1 when the stage of control transfer has changed to the status stage. It is valid for both FW-processed and hardware-processed requests. This bit is also set to 1 when the stage of control transfer (without data) has changed to the status stage. 1: Status stage (interrupt request is generated) 0: Not status stage (default value) This bit is automatically cleared to 0 by hardware when the next SETUP token is received. It is also set to 1 when the stage of control transfer has changed to the status stage while ACK cannot be correctly received in the data stage. In this case, the EP0NKW bit of the UF0E0N register is also cleared to 0 as soon as the UF0E0W register has been cleared, if the FW is processing control transfer (read).
1	PROT	This bit indicates that a SETUP token has been received. It is valid for both FW-processed and hardware-processed requests. 1: SETUP token is correctly received (interrupt request is generated). 0: SETUP token is not received (default value). This bit is set to 1 when data has been correctly received in the UF0E0ST register. Clear this bit to 0 by FW when the first read access is made to the UF0E0ST register. If it is not cleared to 0 by FW, reception of the next SETUP token cannot be correctly recognized. This bit is used to accurately recognize that a SETUP transaction has been executed again during control transfer. If the SETUP transaction is re-executed during control transfer and if a second request is executed by hardware, the CPUDEC bit is not set to 1, but the PROT bit can be used for recognition of the re-execution.
0	CPUDEC	This bit indicates that the UF0E0ST register has a request that is to be decoded by FW. 1: Data is in UF0E0ST register (interrupt request is generated). 0: Data is not in UF0E0ST register (default value). This bit is automatically cleared to 0 by hardware when all the data of the UF0E0ST register is read.

(13) UF0 INT status 2 register (UF0IS2)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB1B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSB1B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC2 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E1IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS2	0	0	BKI1IN	BKI1DT	0	0	0	0	FF29H	00H

Bit position	Bit name	Function
5	BKI1IN	This bit indicates that an IN token has been received in the UF0BI1 register (Endpoint 1) and that NAK has been returned. 1: IN token is received and NAK is transmitted (interrupt request is generated). 0: IN token is not received (default value).
4	BKI1DT	This bit indicates that the FIFO of the UF0BI1 register (Endpoint 1) has been toggled. This means that data can be written to Endpoint 1. 1: FIFO has been toggled (interrupt request is generated). 0: FIFO has not been toggled (default value). The data written to Endpoint 1 is transmitted in synchronization with the IN token next to the one that set the BKI1NK bit of the UF0EN register to 1. When the FIFO has been toggled and then data can be written from the CPU, this bit is automatically set to 1 by hardware. It is also set to 1 when the FIFO has been toggled, even if the data is a Null packet. This bit is automatically cleared to 0 by hardware when the first write access is made to the UF0BI1 register.

(14) UF0 INT status 3 register (UF0IS3)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB1B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSB1B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC3 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E2IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS3	0	0	0	0	BKO1FL	BKO1NL	BKO1NAK	BKO1DT	FF2AH	00H

Bit position	Bit name	Function
3	BKO1FL	This bit indicates that data has been correctly received in the UF0BO1 register (Endpoint 2) and that both the FIFOs of the CPU and SIE hold the data. 1: Received data is in both the FIFOs of the UF0BO1 register (interrupt request is generated). 0: Received data is not in the FIFO on the SIE side of the UF0BO1 register (default value). If data is held in both the FIFOs of the CPU and SIE, this bit is automatically set to 1 by hardware. This bit is automatically cleared to 0 by hardware when the FIFO is toggled.
2	BKO1NL	This bit indicates that a Null packet (packet with a length of 0) has been received in the UF0BO1 register (Endpoint 2). 1: Null packet is received (interrupt request is generated). 0: Null packet is not received (default value). This bit is set to 1 immediately after reception of a Null packet when the FIFO is empty. This bit is set to 1 when the FIFO on the CPU side has been completely read if data is in that FIFO.
1	BKO1NAK	This bit indicates that an OUT token has been received to the UF0BO1 register (Endpoint 2) and that NAK has been returned. 1: OUT token is received and NAK is transmitted (interrupt request is generated). 0: OUT token is not received (default value).
0	BKO1DT	This bit indicates that data has been correctly received in the UF0BO1 register (Endpoint 2). 1: Reception has been completed correctly (interrupt request is generated). 0: Reception has not been completed (default value). This bit is automatically set to 1 by hardware when data has been correctly received and the FIFO has been toggled. At the same time, the corresponding bit of the UF0EPS0 register is also set to 1. This bit is not set to 1 when the data is a Null packet. This bit is automatically cleared to 0 by hardware when the value of the UF0BO1L register becomes 0 as a result of reading the UF0BO1 register by FW. This bit is automatically cleared to 0 when all the contents of the FIFO on the CPU side have been read. However, the interrupt request is not cleared if data is in the FIFO on the SIE side at this time, and the INTUSB1B signal does not become inactive. The signal is kept active if data is successively received.

(15) UF0 INT status 4 register (UF0IS4)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB2B signal becomes active.

This register is read-only, in 8-bit units.

If an interrupt request (INTUSB2B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC4 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 2) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS4	0	0	SETINT	0	0	0	0	0	FF2BH	00H

Bit position	Bit name	Function
5	SETINT	This bit indicates that the SET_INTERFACE request has been received and automatically processed. 1: The request has been automatically processed (interrupt request is generated). 0: The request has not been automatically processed (default value). The current setting of this bit can be identified by reading the UF0ASS or UF0IFn register (n = 0 to 4).

(16) UF0 INT mask 0 register (UF0IM0)

This register controls masking of the interrupt sources indicated by the UF0IS0 register.

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

FW can mask occurrence of an interrupt request (INTUSB0B) from USBF by writing 1 to the corresponding bit of this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM0	BUS RSTM	RSU SPDM	0	0	0	SET RQM	CLR RQM	EP HALTM	FF37H	00H

Bit position	Bit name	Function
7	BUSRSTM	This bit masks the Bus Reset interrupt. 1: Mask 0: Do not mask (default value)
6	RSUSPDM	This bit masks the Resume/Suspend interrupt. 1: Mask 0: Do not mask (default value)
2	SETRQM	This bit masks the SET_RQ interrupt. 1: Mask 0: Do not mask (default value)
1	CLRRQM	This bit masks the CLR_RQ interrupt. 1: Mask 0: Do not mask (default value)
0	EPHALTM	This bit masks the EP_Halt interrupt. 1: Mask 0: Do not mask (default value)

(17) UF0 INT mask 1 register (UF0IM1)

This register controls masking of the interrupt sources indicated by the UF0IS1 register.

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

FW can mask occurrence of an interrupt request (INTUSB0B) from USBF by writing 1 to the corresponding bit of this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM1	0	E0INM	E0 INDTM	E0 ODTM	SUCESM	STGM	PROTM	CPU DECM	FF38H	00H

Bit position	Bit name	Function
6	E0INM	This bit masks the EP0IN interrupt. 1: Mask 0: Do not mask (default value)
5	E0INDTM	This bit masks the EP0INDT interrupt. 1: Mask 0: Do not mask (default value)
4	E0ODTM	This bit masks the EP0OUTDT interrupt. 1: Mask 0: Do not mask (default value)
3	SUCESM	This bit masks the Success interrupt. 1: Mask 0: Do not mask (default value)
2	STGM	This bit masks the Stg interrupt. 1: Mask 0: Do not mask (default value)
1	PROTM	This bit masks the Protect interrupt. 1: Mask 0: Do not mask (default value)
0	CPUDECM	This bit masks the CPUDEC interrupt. 1: Mask 0: Do not mask (default value)

(18) UF0 INT mask 2 register (UF0IM2)

This register controls masking of the interrupt sources indicated by the UF0IS2 register.

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

FW can mask occurrence of an interrupt request (INTUSB1B) from USBF by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E1IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM2	0	0	BKI1INM	BKI1DTM	0	0	0	0	FF39H	00H

Bit position	Bit name	Function
5	BKI1INM	This bit masks the BKI1IN interrupt. 1: Mask 0: Do not mask (default value)
4	BKI1DTM	This bit masks the BLKI1DT interrupt. 1: Mask 0: Do not mask (default value)

(19) UF0 INT mask 3 register (UF0IM3)

This register controls masking of the interrupt sources indicated by the UF0IS3 register.

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

FW can mask occurrence of an interrupt request (INTUSB1B) from USBF by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E2IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM3	0	0	0	0	BKO1FLM	BKO1NLM	BKO1NAKM	BKO1DTM	FF3AH	00H

Bit position	Bit name	Function
3	BKO1FLM	This bit masks the BKO1FL interrupt. 1: Mask 0: Do not mask (default value)
2	BKO1NLM	This bit masks the BKO1NL interrupt. 1: Mask 0: Do not mask (default value)
1	BKO1NAKM	This bit masks the BKO1NK interrupt. 1: Mask 0: Do not mask (default value)
0	BKO1DTM	This bit masks the BKO1DT interrupt. 1: Mask 0: Do not mask (default value)

(20) UF0 INT mask 4 register (UF0IM4)

This register controls masking of the interrupt sources indicated by the UF0IS4 register.

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

FW can mask occurrence of an interrupt request (INTUSB2B) from USBF by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 2) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM4	0	0	SETINTM	0	0	0	0	0	FF3BH	00H

Bit position	Bit name	Function
5	SETINTM	This bit masks the SET_INT interrupt. 1: Mask 0: Do not mask (default value)

(21) UF0 INT clear 0 register (UF0IC0)

This register controls clearing the interrupt sources indicated by the UF0IS0 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC0	BUS RSTC	RSU SPDC	1	1	1	SET RQC	CLR RQC	EP HALTC	FF4AH	FFH

Bit position	Bit name	Function
7	BUSRSTC	This bit clears the Bus Reset interrupt. 0: Clear
6	RSUSPDC	This bit clears the Resume/Suspend interrupt. 0: Clear
2	SETRQC	This bit clears the SET_RQ interrupt. 0: Clear
1	CLRRQC	This bit clears the CLR_RQ interrupt. 0: Clear
0	EPHALTC	This bit clears the EP_Halt interrupt. 0: Clear

(22) UF0 INT clear 1 register (UF0IC1)

This register controls clearing the interrupt sources indicated by the UF0IS1 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC1	1	E0INC	E0 INDTC	E0ODTC	SUCESC	STGC	PROTC	CPU DECC	FF4BH	FFH

Bit position	Bit name	Function
6	E0INC	This bit clears the EP0IN interrupt. 0: Clear
5	E0INDTC	This bit clears the EP0INDT interrupt. 0: Clear
4	E0ODTC	This bit clears the EP0OUTDT interrupt. 0: Clear
3	SUCESC	This bit clears the Success interrupt. 0: Clear
2	STGC	This bit clears the Stg interrupt. 0: Clear
1	PROTC	This bit clears the Protect interrupt. 0: Clear
0	CPUDECC	This bit clears the CPUDEC interrupt. 0: Clear

(23) UF0 INT clear 2 register (UF0IC2)

This register controls clearing the interrupt sources indicated by the UF0IS2 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E1IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC2	1	1	BKI1INC	BKI1DTC	1	1	1	1	FF4CH	FFH

Bit position	Bit name	Function
5	BKI1INC	This bit clears the BKInIN interrupt. 0: Clear
4	BKI1DTC	This bit clears the BKInDT interrupt. 0: Clear

(24) UF0 INT clear 3 register (UF0IC3)

This register controls clearing the interrupt sources indicated by the UF0IS3 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E2IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC3	1	1	1	1	BKO1FLC	BKO1NLC	BKO1NAKC	BKO1DTC	FF4DH	FFH

Bit position	Bit name	Function
3	BKO1FLC	This bit clears the BKO1FL interrupt. 0: Clear
2	BKO1NLC	This bit clears the BKO1NL interrupt. 0: Clear
1	BKO1NAKC	This bit clears the BKO1NK interrupt. 0: Clear
0	BKO1DTC	This bit clears the BKO1DT interrupt. 0: Clear

(25) UF0 INT clear 4 register (UF0IC4)

This register controls clearing the interrupt sources indicated by the UF0IS4 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 2) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC4	1	1	SETINTC	1	1	1	1	1	FF4EH	FFH

Bit position	Bit name	Function
5	SETINTC	This bit clears the SET_INT interrupt. 0: Clear

(26) UF0 FIFO clear 0 register (UF0FIC0)

This register clears each FIFO.

This register is write-only, in 8-bit units. If this register is read, 00H is read.

FW can clear the target FIFO by writing 1 to the corresponding bit of this register. The bit to which 1 has been written is automatically cleared to 0. Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E1IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0FIC0	0	0	BKI1SC	BKI1CC	0	0	EP0WC	EP0RC	FF79H	00H

Bit position	Bit name	Function
5	BKI1SC	This bit clears only the FIFO on the SIE side of the UF0BI1 register (reset the counter). 1: Clear Writing this bit is invalid while an IN token for Endpoint 1 is being processed with the BKI1NK bit set to 1. The BKI1NK bit is automatically cleared to 0 by clearing the FIFO. Make sure that the FIFO on the CPU side is empty when this bit is used.
4	BKI1CC	This bit clears only the FIFO on the CPU side of the UF0BI1 register (reset the counter). 1: Clear
1	EP0WC	This bit clears the UF0E0W register (resets the counter). 1: Clear Writing to this bit is invalid while an IN token for Endpoint 0 is being processed with the EP0NKW bit set to 1. The EP0NKW bit is automatically cleared to 0 by clearing the FIFO.
0	EP0RC	This bit clears the UF0E0R register (resets the counter). 1: Clear When the EP0NKR bit is set to 1 (except when it has been set by FW), the EP0NKR bit is automatically cleared to 0 by clearing the FIFO.

(27) UF0 FIFO clear 1 register (UF0FIC1)

This register clears each FIFO.

This register is write-only, in 8-bit units. If this register is read, 00H is read.

FW can clear the target FIFO by writing 1 to the corresponding bit of this register. The bit to which 1 has been written is automatically cleared to 0. Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E2IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0FIC1	0	0	0	0	0	0	BKO1C	BKO1CC	FF7AH	00H

Bit position	Bit name	Function
1	BKO1C	This bit clears the FIFOs on both the SIE and CPU sides of the UF0BO1 register (reset the counter). 1: Clear When the BKO1NK bit is set to 1 (except when it has been set by FW), the BKO1NK bit is automatically cleared to 0 by clearing the FIFO.
0	BKO1CC	This bit clears only the FIFO on the CPU side of the UF0BO1 register (reset the counter). 1: Clear When the BKO1NK bit is set to 1 (except when it has been set by FW), the BKO1NK bit is automatically cleared to 0 by clearing the FIFO.

(28) UF0 data end register (UF0DEND)

This register reports the end of writing to the transmission system.

This register is write-only, in 8-bit units (however, bits 7 and 6 can be read and written). If this register is read, 00H is read.

FW can start data transfer of the target endpoint by writing 1 to the corresponding bit of this register. The bit to which 1 has been written is automatically cleared to 0. Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0E1IM register and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0DEND	0	0	0	0	0	0	BKI1DED	E0DED	FF75H	00H

Bit position	Bit name	Function
1	BKI1DED	Set this bit to 1 when writing transmit data to the UF0BI1 register has been completed. When this bit is set to 1, the FIFO is toggled as soon as possible, the BKI1NK bit is set to 1, and data is transferred. 1: Transmit a short packet. 0: Do not transmit a short packet (default value). This bit controls the FIFO on the CPU side. If the BKI1CC bit of the UF0FIC0 register is set to 1 and then this bit is set to 1 (counter of UF0BI1 register = 0), a Null packet (with a data length of 0) is transmitted. If data exists in the UF0BI1 register and if this bit is set to 1 (counter of UF0BI1 register ≠ 0), and if the FIFO is not full, a short packet is transmitted. If the FIFO on the CPU side of the UF0BI1 register becomes full, the hardware starts data transmission even if this bit is not set to 1.
0	E0DED	Set this bit to 1 to transmit data of the UF0E0W register. When this bit is set to 1, the EP0NKW bit is set to 1 and data is transferred. 1: Transmit a short packet. 0: Do not transmit a short packet (default value). If the EP0WC bit of the UF0FIC0 register is set to 1 and if this bit is set to 1 (counter of UF0E0W register = 0 and bit 1 of UF0EPS0 register = 1), a Null packet (with a data length of 0) is transmitted. If data exists in the UF0E0W register and if this bit is set to 1 (counter of UF0E0W register ≠ 0 and bit 1 of the UF0EPS0 register = 1), and if the FIFO is not full, a short packet is transmitted.

(29) UF0 GPR register (UF0GPR)

This register controls USBF and the USB interface.

This register is write-only, in 8-bit units. If this register is read, 00H is read. Be sure to clear bits 7 to 2.

FW can reset the USBF by writing 1 to bit 0 of this register. This bit is automatically cleared to 0 after 1 has been written to it. Writing 0 to this bit is invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0GPR	0	0	0	0	0	0	CONNECT	MRST	FF2DH	00H

Bit position	Bit name	Function
1	CONNECT	This bit sets the output level of the USBPUC pin, which controls connection of the pull-up resistor connected to D+. 0: USBPUC pin is low level 1: USBPUC pin is high level For the connection of the USBPUC pin, refer to **12.8.2 USB connection example**.
0	MRST	Set this bit to 1 to reset USBF. 1: Reset Actually, USBF is reset two USB clocks after this bit has been set to 1 by FW and the write signal has become inactive. Resetting USBF by the MRST bit while the system clock is operating has the same result as resetting by the $\overline{\text{RESET}}$ pin (hardware reset) (register value back to default value). However, the UF0CS and UF0BC registers are not reset by the MRST bit.

(30) UF0 mode control register (UF0MODC)

This register controls CPUDEC processing.

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

By setting each bit of this register, the setting of the UF0MODS register can be changed. The bit of this register is automatically cleared to 0 only at hardware reset and when the MRST bit of the UF0GRP register has been set to 1.

Even if the bit of this register has automatically been set to 1 by hardware, the setting by FW takes precedence.

Be sure to clear bits 7 and 5 to 0. If these bits are set to 1, the operation is not guaranteed.

Caution This register is provided for debugging purposes. Usually, do not set this register except for verifying the operation or when a special mode is used.

	7	6	5	4	3	2	1	0	Address	After reset
UF0MODC	0	CDC GDST	0	0	0	0	0	0	FF2EH	00H

Bit position	Bit name	Function
6	CDCGDST	Set this bit to 1 to switch the GET_DESCRIPTOR Configuration request to CPUDEC processing. By setting this bit to 1, the CDCGD bit of the UF0MODS register can be forcibly set to 1. 1: Forcibly change the GET_DESCRIPTOR Configuration request to CPUDEC processing (sets the CDCGD bit of the UF0MODS register to 1). 0: Automatically process the GET_DESCRIPTOR Configuration request (default value).

(31) UF0 mode status register (UF0MODS)

This register indicates the configuration status.

This register is read-only, in 8-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
UF0MODS	0	CDCGD	0	MPACK	DFLT	CONF	0	0	FF2FH	00H

Bit position	Bit name	Function
6	CDCGD	This bit specifies whether CPUDEC processing is performed for the GET_DESCRIPTOR Configuration request. 1: Forcibly change the GET_DESCRIPTOR Configuration request to CPUDEC processing. 0: Automatically process the GET_DESCRIPTOR Configuration request (default value).
4	MPACK	This bit indicates the transmit packet size of Endpoint0. 1: Transmit a packet of other than 8 bytes. 0: Transmit a packet of 8 bytes (default value). This bit is automatically set to 1 by hardware after the GET_DESCRIPTOR Device request has been processed (on normal completion of the status stage). It is not cleared to 0 until the USBF has been reset (it is not cleared to 0 by Bus Reset). If this bit is not set to 1, the hardware transfers only the automatically-executed request in 8-byte units. Therefore, even if data of more than 8 bytes is sent by the OUT token to be processed by FW before completion of the GET_DESCRIPTOR Device request, the data is correctly received. This bit is ignored if the size of Endpoint0 is 8 bytes.
3	DFLT	This bit indicates the default status (DFLT bit = 1). 1: Enables response. 0: Disables response (always no response) (default value). This bit is automatically set to 1 by Bus Reset. The transaction for all the endpoints is not responded to until this bit is set to 1.
2	CONF	This bit indicates whether the SET_CONFIGURATION request has been completed. 1: SET_CONFIGURATION request has been completed. 0: SET_CONFIGURATION request has not been completed (default value). This bit is set to 1 when Configuration value = 1 is received by the SET_CONFIGURATION request. Unless this bit is set to 1, access to an endpoint other than Endpoint0 is ignored. This bit is cleared to 0 when Configuration value = 0 is received by the SET_CONFIGURATION request. It is also cleared to 0 when Bus Reset is detected.

(32) UF0 active interface number register (UF0AIFN)

This register sets the valid Interface number that correctly responds to the GET/SET_INTERFACE request. Because Interface 0 is always valid, Interfaces 1 to 4 can be selected.

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

	7	6	5	4	3	2	1	0	Address	After reset
UF0AIFN	ADDIF	0	0	0	0	0	IFNO1	IFNO0	FF70H	00H

Bit position	Bit name	Function
7	ADDIF	This bit allows use of Interfaces numbered other than 0. 1: Support up to the Interface number specified by the IFNO1 and IFNO0 bits. 0: Support only Interface 0 (default value). Setting bits 1 and 0 of this register is invalid when this bit is not set to 1.
1, 0	IFNO1, IFNO0	These bits specify the range of Interface numbers to be supported.

IFNO1	IFNO0	Valid Interface No.
1	1	0, 1, 2, 3, 4
1	0	0, 1, 2, 3
0	1	0, 1, 2
0	0	0, 1

(33) UF0 active alternative setting register (UF0AAS)

This register specifies a link between the Interface number and Alternative Setting.

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

USBF of the µPD78F0730 can set a five-series Alternative Setting (Alternate Setting 0, 1, 2, 3, and 4 can be defined) and a two-series Alternative Setting (Alternative Setting 0 and 1 can be defined) for one Interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0AAS	ALT2	IFAL21	IFAL20	ALT2EN	ALT5	IFAL51	IFAL50	ALT5EN	FF71H	00H

Bit position	Bit name	Function
7, 3	ALTn	These bits specify whether an n-series Alternative Setting is linked with Interface 0. When these bits are set to 1, the setting of the IFALn1 and IFALn0 bits is invalid. 1: Link n-series Alternative Setting with Interface 0. 0: Do not link n-series Alternative Setting with Interface 0 (default value).
6, 5, 2, 1	IFALn1, IFALn0	These bits specify the Interface number to be linked with the n-series Alternative Setting. If the linked Interface number is outside the range specified by the UF0AIFN register, the n-series Alternative Setting is invalid (ALTnEN bit = 0).

IFALn1	IFALn0	Interface number to be linked
1	1	Links Interface 4.
1	0	Links Interface 3.
0	1	Links Interface 2.
0	0	Links Interface 1.

Do not link a five-series Alternative Setting and a two-series Alternative Setting with the same Interface number.

4, 0	ALTnEN	These bits validate the n-series Alternative Setting. Unless these bits are set to 1, the setting of the ALTn, IFALn1, and IFALn0 bits is invalid. 1: Validate the n-series Alternative Setting. 0: Do not validate the n-series Alternative Setting (default value).

Remark n = 2, 5

For example, when the UF0AIFN register is set to 82H and the UF0AAS register is set to 15H, Interfaces 0, 1, 2, and 3 are valid. Interfaces 0 and 2 support only Alternative Setting 0. Interface 1 supports Alternative Setting 0 and 1, and Interface 3 supports Alternative Setting 0, 1, 2, 3, and 4. With this setting, requests GET_INTERFACE wIndex = 0/1/2/3, SET_INTERFACE wValue = 0 & wIndex = 0/2, SET_INTERFACE wValue = 0/1 & wIndex = 1, and SET_INTERFACE wValue = 0/1/2/3/4 & wIndex = 3 are automatically responded to, and a STALL response is made to the other GET/SET_INTERFACE requests.

(34) UF0 alternative setting status register (UF0ASS)

This register indicates the current status of the Alternative Setting.

This register is read-only, in 8-bit units.

Check this register when the SET_INT interrupt request has been issued. The value received by the SET_INTERFACE request is reflected on the UF0IFn register (n = 0 to 4) as well as on this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0ASS	0	0	0	0	AL5ST3	AL5ST2	AL5ST1	AL2ST	FF72H	00H

Bit position	Bit name	Function
3 to 1	AL5ST3 to AL5ST1	These bits indicate the current status of the five-series Alternative Setting.
0	AL2ST	This bit indicates the current status of the two-series Alternative Setting (selected Alternative Setting number). 1: Alternative Setting 1 0: Alternative Setting 0

AL5ST3	AL5ST2	AL5ST1	Selected Alternative Setting number
1	0	0	Alternative Setting 4
0	1	1	Alternative Setting 3
0	1	0	Alternative Setting 2
0	0	1	Alternative Setting 1
0	0	0	Alternative Setting 0

(35) UF0 endpoint 1 interface mapping register (UF0E1IM)

This register specifies for which Interface and Alternative Setting Endpoint1 is valid.

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

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint1 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint1 request and the IN transaction to Endpoint1 are responded to, and whether the related bits are valid or invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E1IM	E1EN2	E1EN1	E1EN0	E12AL1	E15AL4	E15AL3	E15AL2	E15AL1	FF73H	00H

Bit position	Bit name	Function
7 to 5	E1EN2 to E1EN0	These bits set a link between the Interface of Endpoint1 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.

E1EN2	E1EN1	E1EN0	Link status
1	1	1	Not linked with Interface
1	1	0	
1	0	1	Linked with Interface 4 and Alternative Setting 0
1	0	0	Linked with Interface 3 and Alternative Setting 0
0	1	1	Linked with Interface 2 and Alternative Setting 0
0	1	0	Linked with Interface 1 and Alternative Setting 0
0	0	1	Linked with Interface 0 and Alternative Setting 0
0	0	0	Not linked with Interface (default value)

When these bits are set to 110 or 111, they are invalid even if the E12AL1 bit is cleared to 0.

If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint1 is valid.

4	E12AL1	This bit validates Endpoint1 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1. 1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1. 0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value). This bit is valid when the E15AL4 to E15AL1 bits are 0000.
3 to 0	E15ALn	These bits validate Endpoint1 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n. 1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1. 0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value).

Remark n = 1 to 4

(36) UF0 endpoint 2 interface mapping register (UF0E2IM)

This register specifies for which Interface and Alternative Setting Endpoint2 is valid.

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

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint2 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint2 request and the OUT transaction to Endpoint2 are responded to, and whether the related bits are valid or invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E2IM	E2EN2	E2EN1	E2EN0	E22AL1	E25AL4	E25AL3	E25AL2	E25AL1	FF74H	00H

Bit position	Bit name	Function
7 to 5	E2EN2 to E2EN0	These bits set a link between the Interface of Endpoint2 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.

E2EN2	E2EN1	E2EN0	Link status
1	1	1	Not linked with Interface
1	1	0	
1	0	1	Linked with Interface 4 and Alternative Setting 0
1	0	0	Linked with Interface 3 and Alternative Setting 0
0	1	1	Linked with Interface 2 and Alternative Setting 0
0	1	0	Linked with Interface 1 and Alternative Setting 0
0	0	1	Linked with Interface 0 and Alternative Setting 0
0	0	0	Not linked with Interface (default value)

When these bits are set to 110 or 111, they are invalid even if the E22AL1 bit is cleared to 0.

If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint2 is valid.

4	E22AL1	This bit validates Endpoint2 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1. 1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1. 0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value). This bit is valid when the E25AL4 to E25AL1 bits are 0000.
3 to 0	E25ALn	These bits validate Endpoint2 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n. 1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1. 0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value).

Remark n = 1 to 4

12.4.2 Data hold registers

(1) UF0 EP0 read register (UF0E0R)

The UF0E0R register is a 64-byte FIFO that stores the OUT data sent from the host in the data stage of control transfer to/from Endpoint0.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The hardware automatically transfers data to the UF0E0R register when it has received the data from the host. When the data has been correctly received, the E0ODT bit of the UF0IS1 register is set to 1. The UF0E0L register holds the quantity of the received data, and an interrupt request (INTUSB0B) is issued. The UF0E0L register always updates the length of the received data while it is receiving data. If the final transfer is correct reception, the interrupt request is generated. If the reception is abnormal, the UF0E0L register is cleared to 0 and the interrupt request is not generated.

The data held by the UF0E0R register must be read by FW up to the value of the amount of data read by the UF0E0L register. Check that all data has been read by using the EP0R bit of the UF0EPS0 register (EP0R = 0 when all data has been read). If the value of the UF0E0L register is 0, the EP0NKR bit of the UF0E0N register is cleared to 0, and the UF0E0R register is ready for reception. The UF0E0R register is cleared when the next SETUP token has been received.

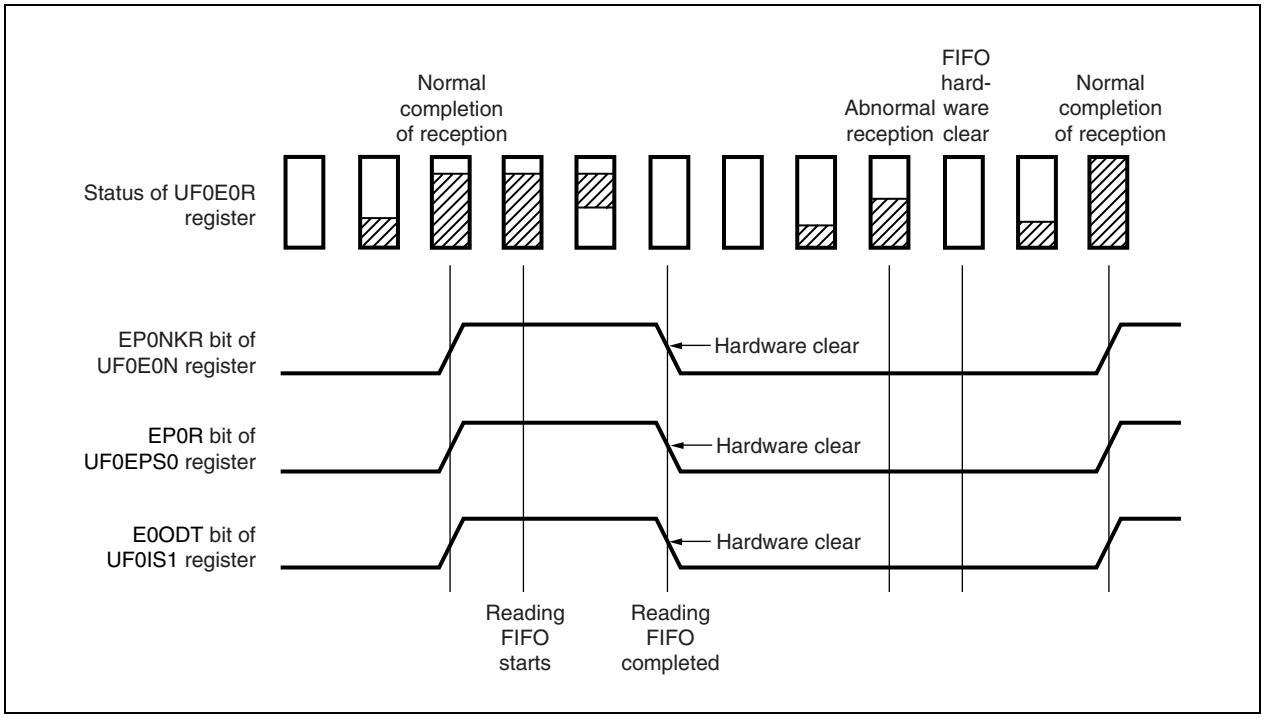
Caution Read all the data stored. Clear the FIFO to discard some data.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0R	E0R7	E0R6	E0R5	E0R4	E0R3	E0R2	E0R1	E0R0	FF02H	Undefined

Bit position	Bit name	Function
7 to 0	E0R7 to E0R0	These bits store the OUT data sent from the host in the data stage of control transfer to/from Endpoint0.

The operation of the UF0E0R register is illustrated below.

Figure 12-2. Operation of UF0E0R Register



(2) UF0 EP0 length register (UF0E0L)

The UF0E0L register stores the data length held by the UF0E0R register.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The UF0E0L register always updates the length of the received data while it is receiving data. If the final transfer is abnormal reception, the UF0E0L register is cleared to 0 and the interrupt request is not generated. The interrupt request is generated only when the reception is normal, and the FW can read as many data from the UF0E0R register as the value read from the UF0E0L register. The value of the UF0E0L register is decremented each time the UF0E0R register has been read.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0L	E0L7	E0L6	E0L5	E0L4	E0L3	E0L2	E0L1	E0L0	FF76H	00H

Bit position	Bit name	Function
7 to 0	E0L7 to E0L0	These bits store the data length held by the UF0E0R register.

(3) UF0 EP0 setup register (UF0E0ST)

The UF0E0ST register holds the SETUP data sent from the host.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The UF0E0ST register always writes data when a SETUP transaction has been received. The hardware sets the PROT bit of the UF0IS1 register when it has correctly received the SETUP transaction. It sets the CPUDEC bit of the UF0IS1 register in the case of an FW-processed request. Then an interrupt request (INTUSB0B) is issued. In the case of an FW-processed request, be sure to read the request in 8-byte units. If it is not read in 8-byte units, the subsequent requests cannot be correctly decoded. The read counter of the UF0E0ST register is not cleared even when Bus Reset is received. Always read this counter in 8-byte units regardless of whether Bus Reset is received or not.

Because the UF0E0ST register always enables writing, the hardware overwrites data to this register even if a SETUP transaction is received while the data of the register is being read. Even if the SETUP transaction cannot be correctly received, the CPUDEC interrupt request and Protect interrupt request are not generated, but the previous data is discarded. If a SETUP token of less than 8 bytes is received, however, the received SETUP token is discarded, and the previously received SETUP data is retained. If the SETUP token is received more than once when control transfer is executed once, be sure to check the PROT bit of the UF0IS1 register under the conditions below. If PROT bit = 1, read the UF0E0ST register again because the SETUP transaction has been received more than once.

<1> If a request is decoded by FW and the UF0E0R register is read or the UF0E0W register is written

<2> When preparing for a STALL response for the request to which the decode result does not correspond

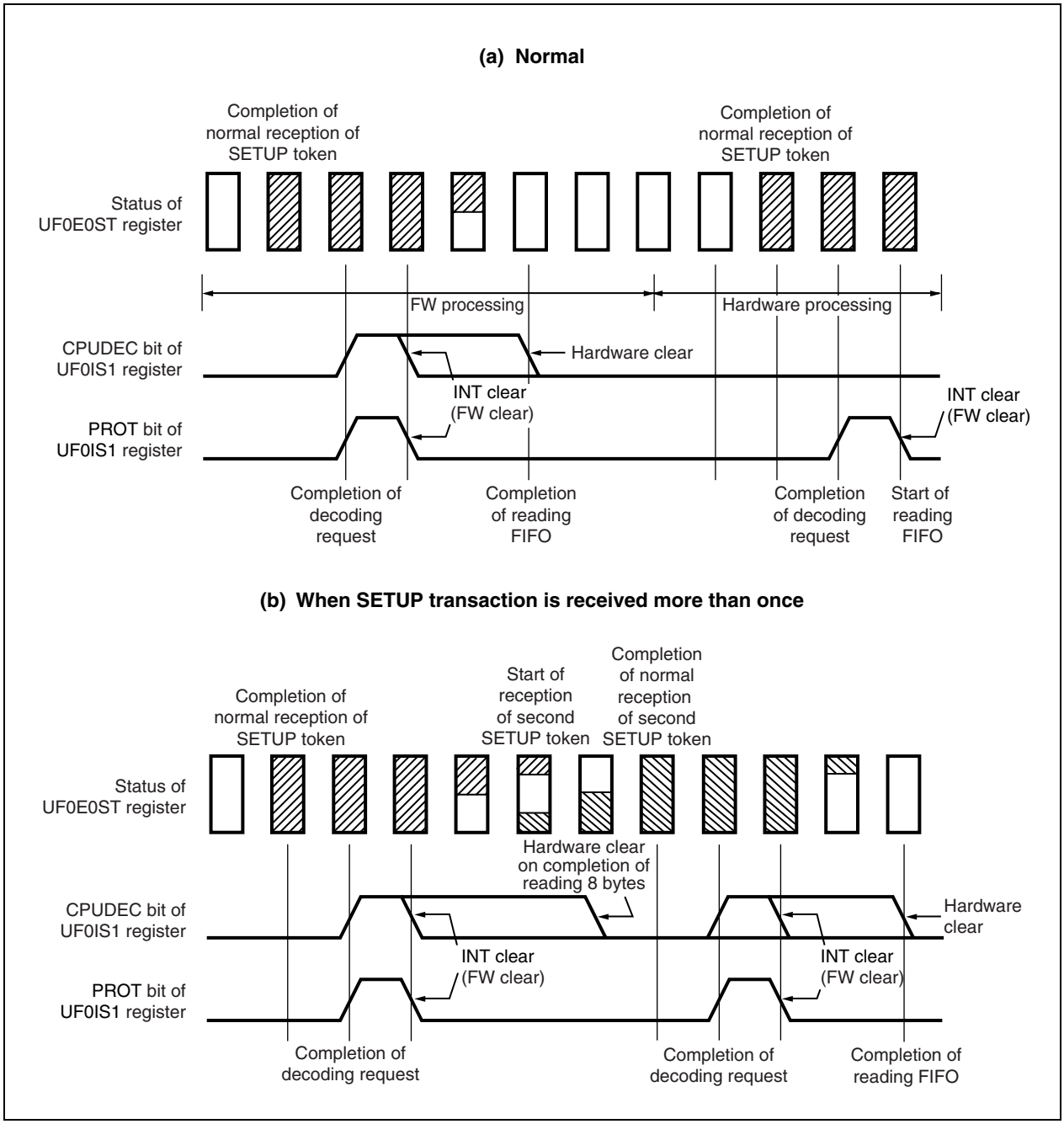
Caution Be sure to read all the stored data. The UF0E0ST register is always updated by the request in the SETUP transaction.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0ST	E0S7	E0S6	E0S5	E0S4	E0S3	E0S2	E0S1	E0S0	FF18H	00H

Bit position	Bit name	Function
7 to 0	E0S7 to E0S0	These bits hold the SETUP data sent from the host.

The operation of the UF0E0ST register is illustrated below.

Figure 12-3. Operation of UF0E0ST Register



(4) UF0 EP0 write register (UF0E0W)

The UF0E0W register is a 64-byte FIFO that stores the IN data (passes it to SIE) sent to the host in the data stage to Endpoint0.

This register is write-only, in 8-bit units. When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with an IN token only when the EP0NKW bit of the UF0E0N register is set to 1 (when NAK is not transmitted). When data is transmitted and when the host correctly receives the data, the EP0NKW bit of the UF0E0N register is automatically cleared to 0 by hardware. A short packet is transmitted when data is written to the UF0E0W register and the E0DED bit of the UF0DEND register is set to 1 (EP0W bit of the UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0E0W register is cleared and the E0DED bit of the UF0DEND register is set to 1 (EP0W bit of the UF0EPS0 register = 1 (data exists)).

The UF0E0W register is cleared to 0 when the next SETUP token is received while transmission has not been completed yet. If the stage of control transfer (read) changes to the status stage while ACK has not been correctly received in the data stage, the UF0E0W register is automatically cleared to 0. At the same time, it is also cleared to 0 if the EP0NKW bit of the UF0E0N register is 1.

If the UF0E0W register is read while no data is in it, 00H is read.

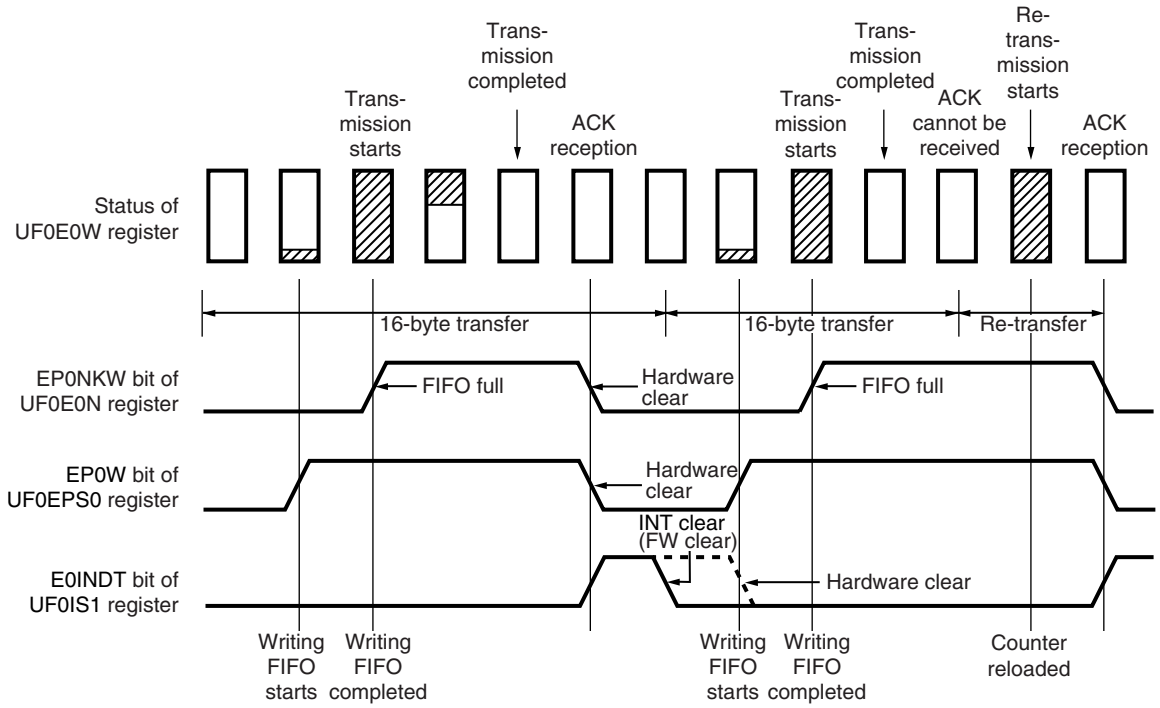
	7	6	5	4	3	2	1	0	Address	After reset
UF0E0W	E0W7	E0W6	E0W5	E0W4	E0W3	E0W2	E0W1	E0W0	FF19H	Undefined

Bit position	Bit name	Function
7 to 0	E0W7 to E0W0	These bits store the IN data sent to the host in the data stage to Endpoint0.

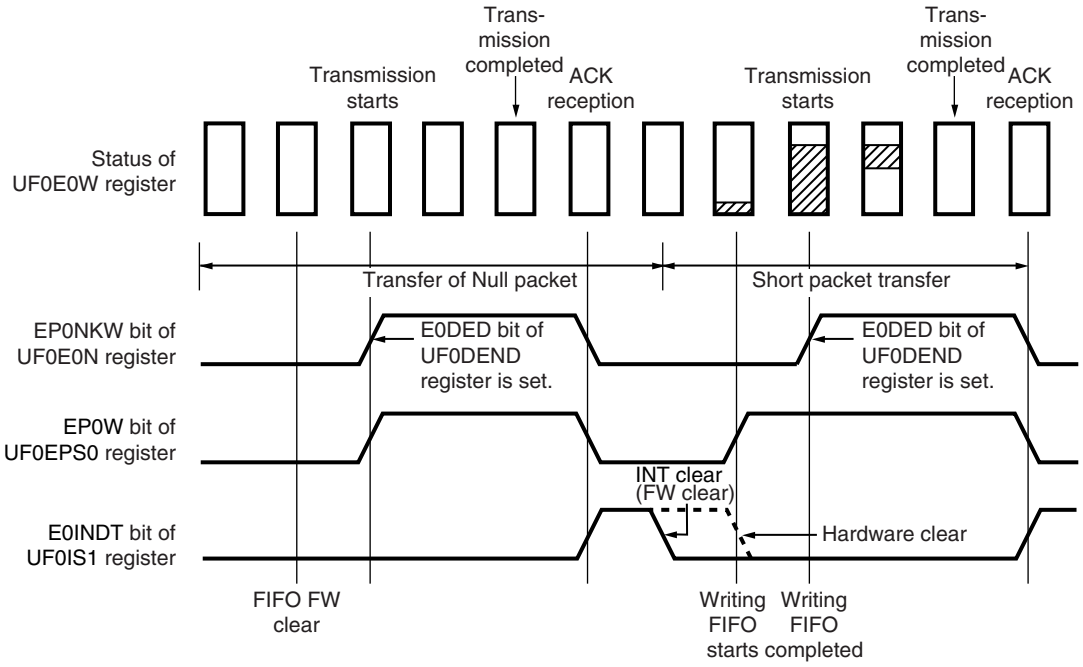
The operation of the UF0E0W register is illustrated below.

Figure 12-4. Operation of UF0E0W Register

(a) 16-byte transmission



(b) When Null packet or short packet is transmitted



(5) UF0 bulk out 1 register (UF0BO1)

The UF0BO1 register is a 64-byte × 2 FIFO that stores data for Endpoint2. This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides. The toggle operation takes place when data is in the FIFO on the SIE side and when no data is in the FIFO on the CPU side (counter value = 0).

This register is read-only, in 8-bit units. A write access to this register is ignored.

When the hardware receives data for Endpoint2 from the host, it automatically transfers the data to the UF0BO1 register. When the register correctly receives the data, a FIFO toggle operation occurs. As a result, the BKO1DT bit of the UF0IS3 register is set to 1, the quantity of the received data is held by the UF0BO1L register, and an interrupt request is issued to the CPU.

Read the data held by the UF0BO1 register by FW, up to the value of the amount of data read by the UF0BO1L register. When the correct received data is held by the FIFO connected to the SIE side and the value of the UF0BO1L register reaches 0, the toggle operation of the FIFO occurs, and the BKO1NK bit of the UF0EN register is automatically cleared to 0. If data greater than the value of the UF0BO1L register is read and if the FIFO toggle condition is satisfied, the toggle operation of the FIFO occurs. As a result, the next packet may be read by mistake. Note that, if the toggle condition is not satisfied, the first data is repeatedly read.

If overrun data is received while data is held by the FIFO connected to the CPU side, Endpoint2 stalls, and the FIFO on the CPU side is cleared.

When the UF0BO1 register is read while no data is in it, an undefined value is read.

Caution Be sure to read all the data stored in this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0BO1	BKO17	BKO16	BKO15	BKO14	BKO13	BKO12	BKO11	BKO10	FF0DH	Undefined

Bit position	Bit name	Function
7 to 0	BKO17 to BKO10	These bits store data for Endpoint2.

The operation of the UF0BO1 register is illustrated below.

Figure 12-5. Operation of UF0BO1 Register (1/2)

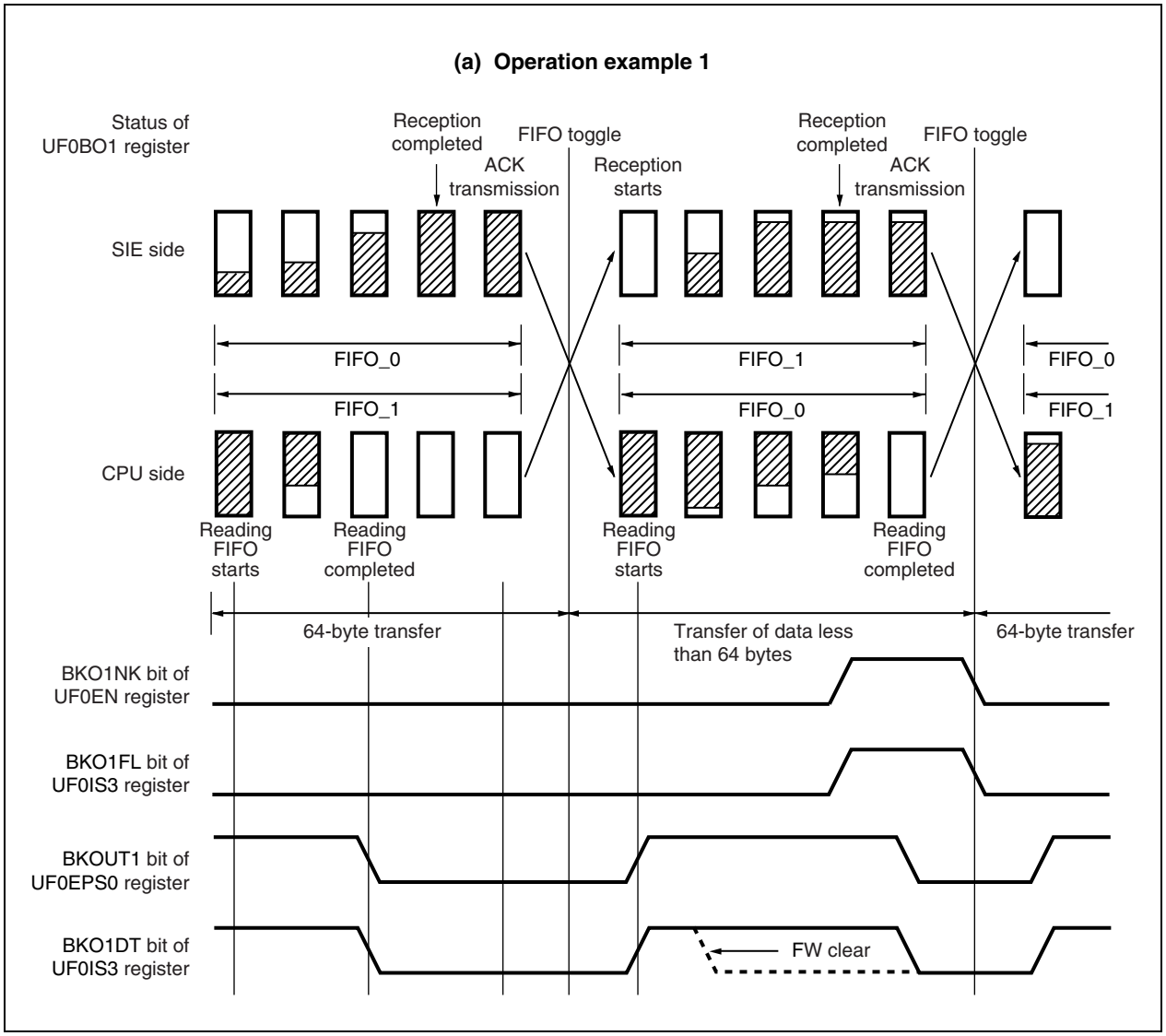
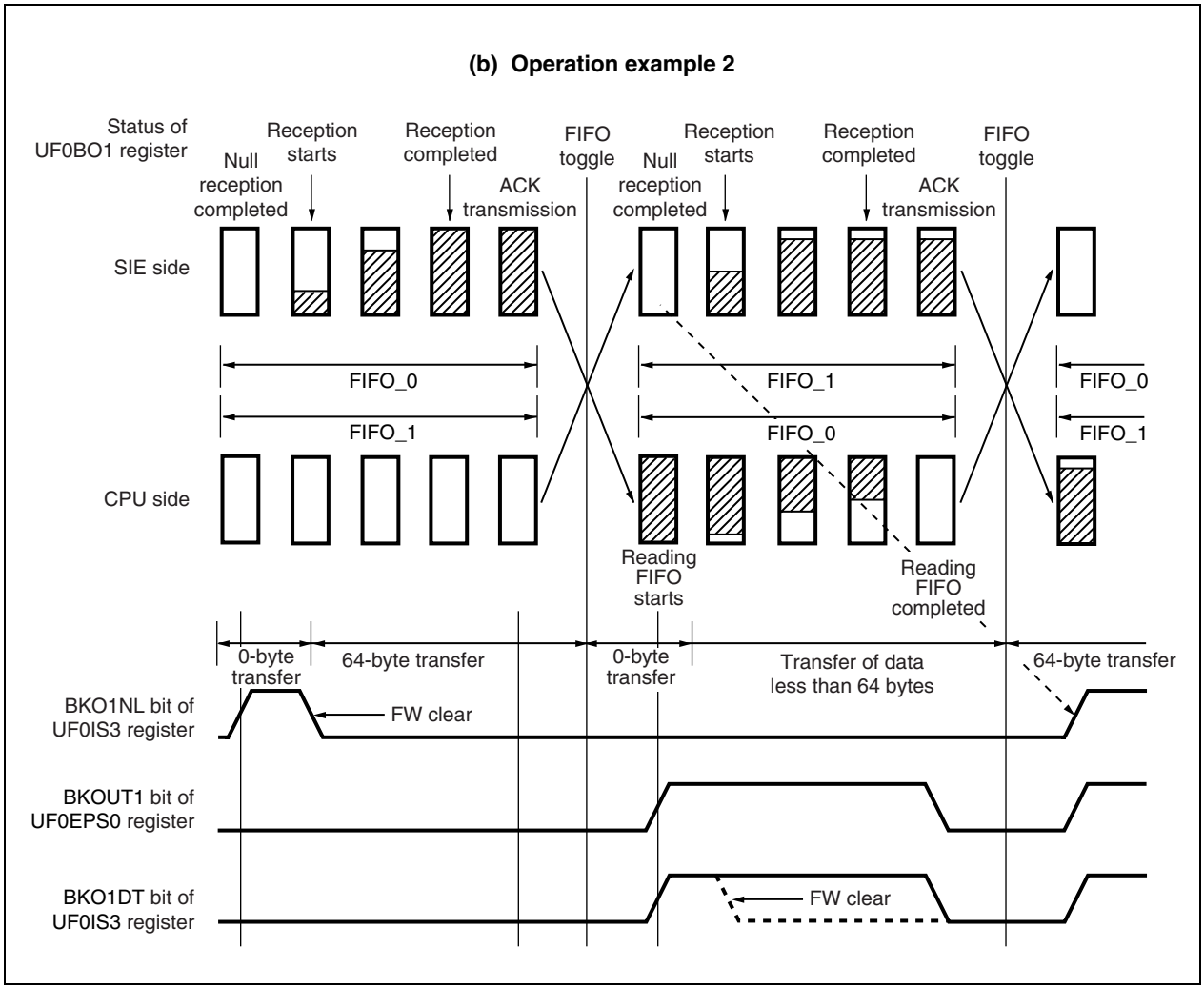


Figure 12-5. Operation of UF0BO1 Register (2/2)



(6) UF0 bulk out 1 length register (UF0BO1L)

The UF0BO1L register stores the length of the data held by the UF0BO1 register.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The UF0BO1L register always updates the received data length while it is receiving data. If the final transfer is abnormal reception, the UF0BO1L register is cleared to 00H, and an interrupt request is not generated. Only if the reception is normal, the interrupt request is generated, and FW can read as much data from the UF0BO1 register as the value read from the UF0BO1L register. The value of the UF0BO1L register is decremented each time the UF0BO1 register has been read.

	7	6	5	4	3	2	1	0	Address	After reset
UF0BO1L	BKO1L7	BKO1L6	BKO1L5	BKO1L4	BKO1L3	BKO1L2	BKO1L1	BKO1L0	FF77H	00H

Bit position	Bit name	Function
7 to 0	BKO1L7 to BKO1L0	These bits store the length of the data held by the UF0BO1 register.

(7) UF0 bulk in 1 register (UF0BI1)

The UF0BI1 register is a 64-byte × 2 FIFO that stores data for Endpoint1. This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides. The toggle operation takes place when no data is in the FIFO on the SIE side (counter value = 0) and when the FIFO on the CPU side is correctly written (FIFO full or BKI1DED bit = 1).

This register is write-only, in 8-bit units. When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with the IN token for Endpoint1 only when the BKI1NK bit of the UF0EN register is set to 1 (when NAK is not transmitted). The address at which data is to be written or read is managed by the hardware. Therefore, FW can transmit data to the host only by writing the data to the UF0BI1 register sequentially. A short packet is transmitted when data is written to the UF0BI1 register and the BKI1DED bit of the UF0DEND register is set to 1 (BKIN1 bit of UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0BI1 register is cleared and the BKI1DED bit of the UF0DEND register is set to 1 (BKIN1 bit of the UF0EPS0 register = 1 (data exists)). When the register correctly transmits the data, a FIFO toggle operation occurs. As a result, the BKI1DT bit of the UF0IS2 register is set to 1, and an interrupt request is issued to the CPU.

	7	6	5	4	3	2	1	0	Address	After reset
UF0BI1	BKI17	BKI16	BKI15	BKI14	BKI13	BKI12	BKI11	BKI10	FF0EH	Undefined

Bit position	Bit name	Function
7 to 0	BKI17 to BKI10	These bits store data for Endpoint1.

The operation of the UF0BI1 register is illustrated below.

Figure 12-6. Operation of UF0BI1 Register (1/3)

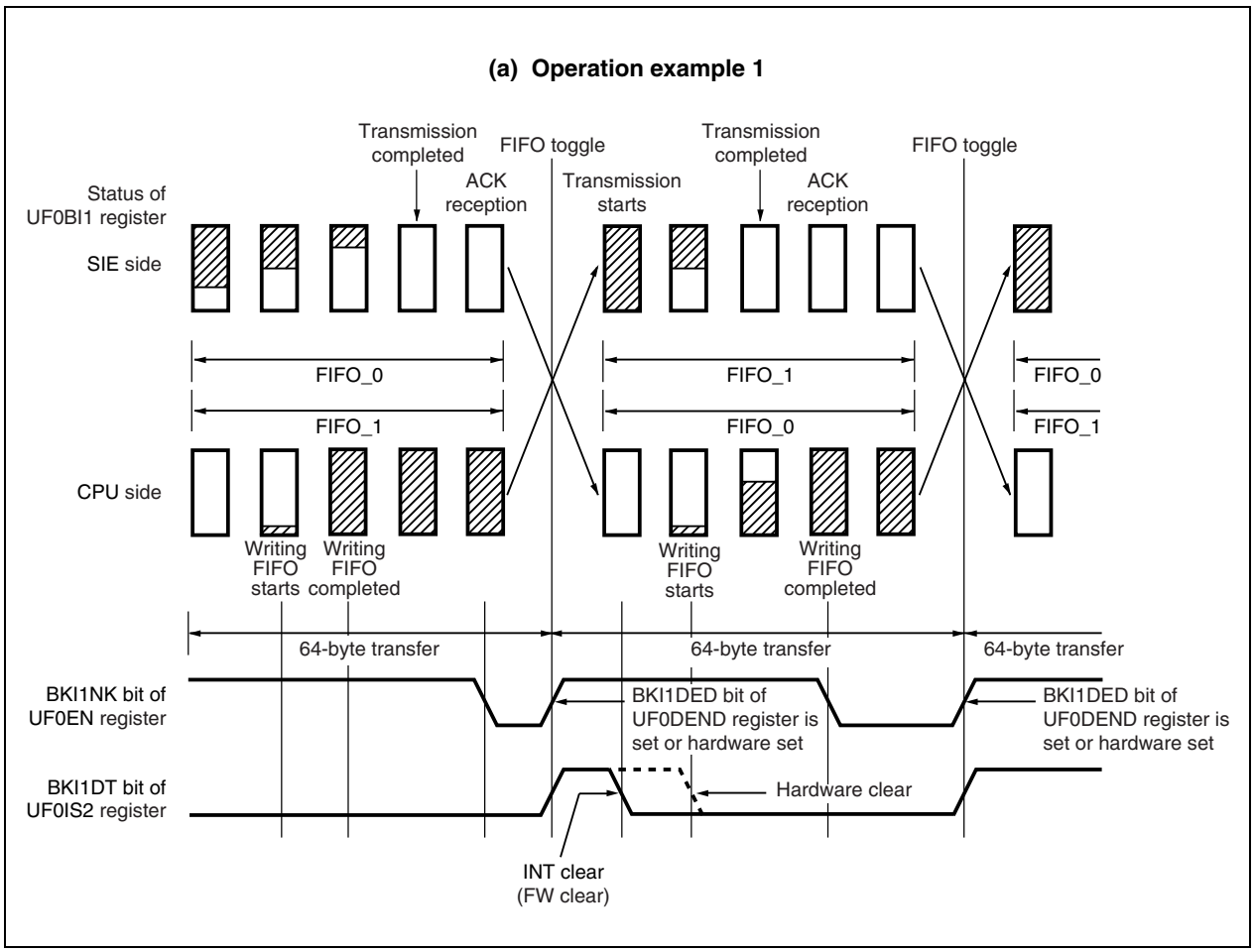


Figure 12-6. Operation of UF0BI1 Register (2/3)

(b) Operation example 2

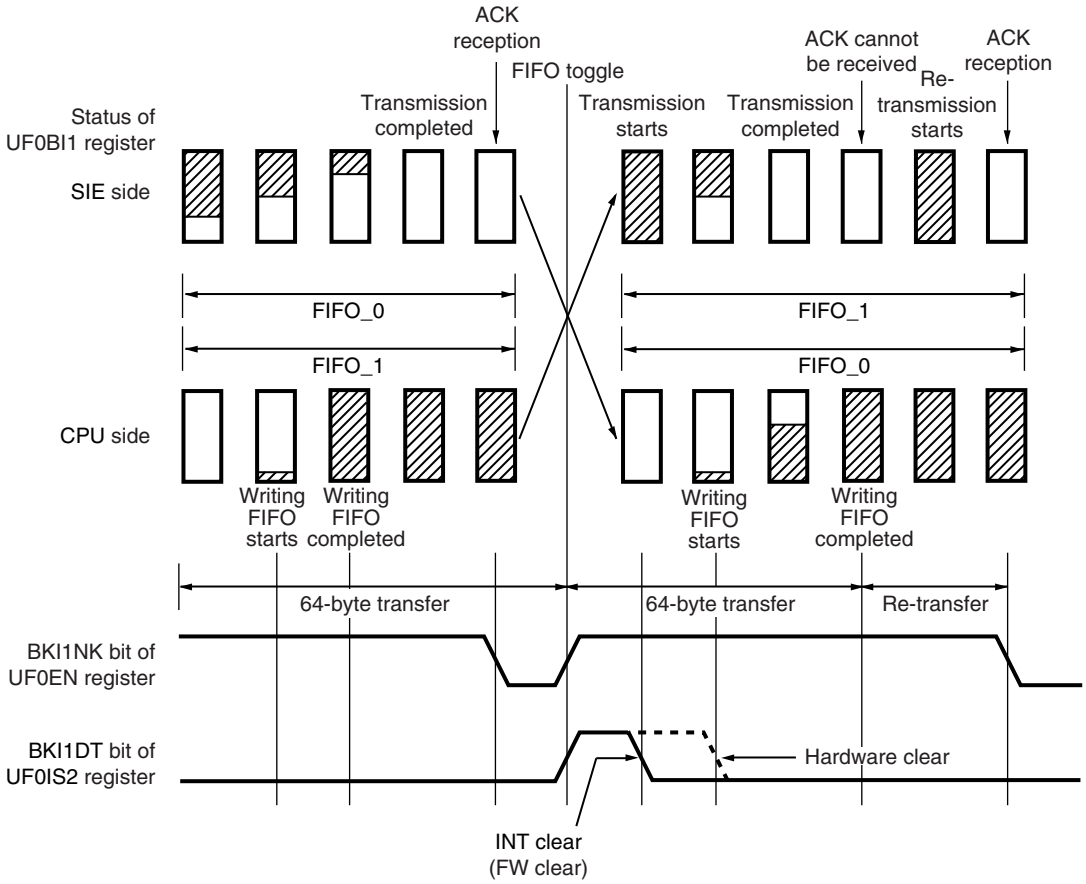
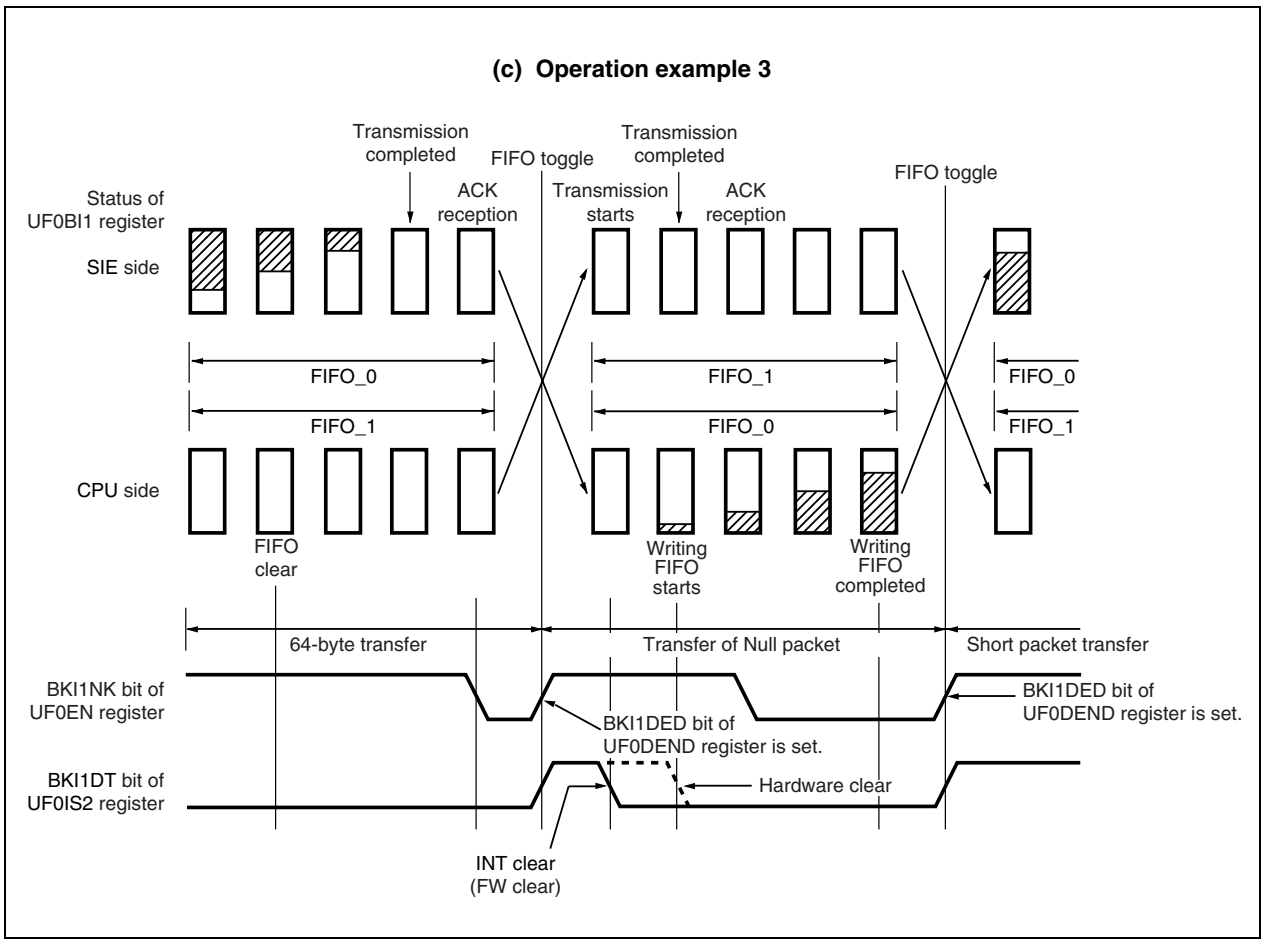


Figure 12-6. Operation of UF0BI1 Register (3/3)



12.4.3 Request data registers

(1) UF0 device status register L (UF0DSTL)

This register stores the value that is to be returned in response to the GET_STATUS Device request.

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

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Device request.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0DSTL	0	0	0	0	0	0	RMWK	SFPW	FF9AH	00H

Bit position	Bit name	Function
1	RMWK	This bit specifies whether the remote wakeup function of the device is used. 1: Enabled 0: Disabled If the device supports a remote wakeup function, this bit is set to 1 by hardware when the SET_FEATURE Device request has been received, and is cleared to 0 by hardware when the CLEAR_FEATURE Device request has been received. If the device does not support a remote wakeup function, make sure that the SET_FEATURE Device request is not issued from the host.
0	SFPW	This bit indicates whether the device is self-powered or bus-powered. 1: Self-powered 0: Bus-powered

(2) UF0 EP0 status register L (UF0E0SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint0 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in USBF, the E0HALT bit is set to 1 by FW. A write access to this register is ignored while a USB-side access to Endpoint0 is being received.

When the E0HALT bit is set to 1 by FW, it is not reflected until the next SETUP token is received if the control transfer immediately before is for the SET_FEATURE Endpoint0, CLEAR_FEATURE Endpoint0, GET_STATUS Endpoint0 request, or an FW-processed request.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint0 request. If Endpoint0 has stalled, the UF0E0W and UF0E0R registers are cleared, and the EP0NKW and EP0NKR bits of the UF0E0N register are cleared to 0.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0SL	0	0	0	0	0	0	0	E0HALT	FF9CH	00H

Bit position	Bit name	Function
0	E0HALT	This bit indicates the status of Endpoint0. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint0 request has been received, and cleared to 0 by hardware when the CLEAR_FEATURE Endpoint0 request has been received. DATA PID is initialized to DATA0.

(3) UF0 EP1 status register L (UF0E1SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint1 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint1, the E1HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint1 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint1 request. If Endpoint1 has stalled, the UF0BI1 register is cleared and the BKI1NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint1, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E1SL	0	0	0	0	0	0	0	E1HALT	FF9DH	00H

Bit position	Bit name	Function
0	E1HALT	This bit indicates the status of Endpoint1. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint1 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint1 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint1 is linked has correctly been received. DATA PID is initialized to DATA0.

(4) UF0 EP2 status register L (UF0E2SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint2 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint2, the E2HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint2 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint2 request. If Endpoint2 has stalled, the UF0BO1 register is cleared and the BKO1NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint2, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E2SL	0	0	0	0	0	0	0	E2HALT	FF9EH	00H

Bit position	Bit name	Function
0	E2HALT	This bit indicates the status of Endpoint2. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint2 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint2 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint2 is linked has correctly been received. DATA PID is initialized to DATA0.

(5) UF0 address register (UF0ADRS)

This register stores the device address.

This register is read-only, in 8-bit units.

The device address sent by the SET_ADDRESS request is analyzed and the resultant value is automatically written to this register. If the SET_ADDRESS request is processed by FW, the value of this register is reflected as the device address when the SUCCESS signal is received in the status stage.

Caution Do not perform write access to this register. Operation is not guaranteed if this register is written.

	7	6	5	4	3	2	1	0	Address	After reset
UF0ADRS	0	ADRS6	ADRS5	ADRS4	ADRS3	ADRS2	ADRS1	ADRS0	FF90H	00H

Bit position	Bit name	Function
6 to 0	ADRS6 to ADRS0	These bits hold the device address of SIE.

(6) UF0 configuration register (UF0CNF)

This register stores the value that is to be returned in response to the GET_CONFIGURATION request.

This register is read-only, in 8-bit units.

When the SET_CONFIGURATION request is received, its wValue is automatically written to this register.

When a change of the value of this register from 00H to other than 00H is detected, the CONF bits of UF0MODS register are set to 1. If the SET_CONFIGURATION request is processed by FW, the status of this register is immediately reflected on the UF0MODS register as soon as data has been written to this register (CONF bits = 1 before completion of the status stage).

Caution Do not perform write access to this register. Operation is not guaranteed if this register is written.

	7	6	5	4	3	2	1	0	Address	After reset
UF0CNF	0	0	0	0	0	0	CONF1	CONF0	FF91H	00H

Bit position	Bit name	Function
1, 0	CONF1, CONF0	These bits hold the data to be returned in response to the GET_CONFIGURATION request.

(7) UF0 interface 0 register (UF0IF0)

This register stores the value that is to be returned in response to the GET_INTERFACE wIndex = 0 request.

This register is read-only, in 8-bit units.

When the SET_INTERFACE request is received, its wValue is automatically written to this register.

If the SET_INTERFACE request is processed by FW, wIndex and wValue are decoded, and the setting of endpoint is automatically changed. At this time, the status bit of the target endpoint and DPID are automatically cleared to 0, depending on the setting. The FIFO is not cleared automatically.

Caution Do not perform write access to this register. Operation is not guaranteed if this register is written.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IF0	0	0	0	0	0	IF02	IF01	IF00	FF92H	00H

Bit position	Bit name	Function
2 to 0	IF02 to IF00	These bits hold the data to be returned in response to GET_INTERFACE wIndex = 0 request.

(8) UF0 interface 1 to 4 registers (UF0IF1 to UF0IF4)

These registers store the value that is to be returned in response to the GET_INTERFACE wIndex = n request (n = 1 to 4).

These registers are read-only, in 8-bit units.

When the SET_INTERFACE request is received, its wValue is automatically written to these registers.

These registers are invalidated according to the setting of the UF0AIFN and UF0AAS registers.

If the SET_INTERFACE request is processed by FW, wIndex and wValue are decoded, and the setting of endpoint is automatically changed. At this time, the status bit of the target endpoint and DPID are automatically cleared to 0, depending on the setting. The FIFO is not cleared automatically.

Caution Do not perform write access to this register. Operation is not guaranteed if this register is written.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IF1	0	0	0	0	0	IF12	IF11	IF10	FF93H	00H
UF0IF2	0	0	0	0	0	IF22	IF21	IF20	FF94H	00H
UF0IF3	0	0	0	0	0	IF32	IF31	IF30	FF95H	00H
UF0IF4	0	0	0	0	0	IF42	IF41	IF40	FF96H	00H

Bit position	Bit name	Function
2 to 0	IFn2 to IFn0	These bits hold the data to be returned in response to GET_INTERFACE wIndex = n request.

Remark n = 1 to 4

(9) UF0 descriptor length register (UF0DSCL)

This register stores the length of the value that is to be returned in response to the GET_DESCRIPTOR Configuration request. The value of this register is the number of bytes of all the descriptors set by the UF0CIEn register minus 1 (n = 0 to 255). The total descriptor length that is to be returned in response to the GET_DESCRIPTOR Configuration request is determined according to the value of this register.

This register can be read or written in 8-bit units. However, data can be written to this register only when the EP0NKA bit is set to 1.

Processing of wLength is automatically controlled. If this register is set to 00H, it means that the descriptor to be returned is 1 byte long. If the register is set to FFH, a descriptor length of 256 bytes is returned. When a descriptor exceeding 256 bytes in length is used, set the CDCGDST bit of the UF0MODC register to 1 and process the GET_DESCRIPTOR request by FW (at this time, the CDCGD bit of the UF0MODS register is also set to 1).

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0DSCL	DPL7	DPL6	DPL5	DPL4	DPL3	DPL2	DPL1	DPL0	FF78H	00H

Bit position	Bit name	Function
7 to 0	DPL7 to DPL0	These bits set the value of the number of bytes of all the descriptors to be returned in response to the GET_DESCRIPTOR Configuration request minus 1.

(10) UF0 device descriptor registers 0 to 17 (UF0DD0 to UF0DD17)

These registers store the value to be returned in response to the GET_DESCRIPTOR Device request.

These registers can be read or written in 8-bit units. However, data can be written to these registers only when the EP0NKA bit is set to 1.

Cautions 1. To rewrite these registers, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

2. Use the value defined by USB Specification Ver. 2.0 and the latest Class Specification as the set value.

	7	6	5	4	3	2	1	0	Address	After reset
UF0DDn (n = 0 to 17)									See **Table 12-5**.	Undefined

Table 12-5. Mapping and Data of UF0 Device Descriptor Registers

Symbol	Address	Field Name	Contents
UF0DD0	F9D1H	bLength	Size of this descriptor
UF0DD1	F9D2H	bDescriptorType	Device descriptor type
UF0DD2	F9D3H	bcdUSB	Value below decimal point of Rev. number of USB specification
UF0DD3	F9D4H		Value above decimal point of Rev. number of USB specification
UF0DD4	F9D5H	bDeviceClass	Class code
UF0DD5	F9D6H	bDeviceSubClass	Subclass code
UF0DD6	F9D7H	bDeviceProtocol	Protocol code
UF0DD7	F9D8H	bMaxPacketSize0	Maximum packet size of Endpoint0
UF0DD8	F9D9H	idVendor	Lower value of vendor ID
UF0DD9	F9DAH		Higher value of vendor ID
UF0DD10	F9DBH	idProduct	Lower value of product ID
UF0DD11	F9DCH		Higher value of product ID
UF0DD12	F9DDH	bcdDevice	Lower value of device release number
UF0DD13	F9DEH		Higher value of device release number
UF0DD14	F9DFH	iManufacturer	Index of string descriptor describing manufacturer
UF0DD15	F9E0H	iProduct	Index of string descriptor describing product
UF0DD16	F9E1H	ISerialNumber	Index of string descriptor describing device serial number
UF0DD17	F9E2H	BNumConfigurations	Number of settable configurations

(11) UF0 configuration/interface/endpoint descriptor registers 0 to 255 (UF0CIE0 to UF0CIE255)

These registers store the value to be returned in response to the GET_DESCRIPTOR Configuration request.

These registers can be read or written in 8-bit units. However, data can be written to these registers only when the EP0NKA bit is set to 1.

Descriptor information of up to 256 bytes can be stored in these registers. Store each descriptor in the order of Configuration, Interface, and Endpoint (see **Table 12-6**). If there are two or more Interfaces, repeatedly store the data following the Interface descriptor.

Table 12-6. Mapping of UF0CIEn Register

Address	Descriptor Stored
F9E3H	Configuration descriptor (9 bytes)
F9ECH	Interface descriptor (9 bytes)
F9F5H	Endpoint1 descriptor (7 bytes)
F9FCH	Endpoint2 descriptor (7 bytes)
FA03H	:
:	:
FAxxH	Interface descriptor (9 bytes)
FAxxH + 9	Endpoint1 descriptor (7 bytes)
FAxxH + 16	Endpoint2 descriptor (7 bytes)
FAxxH + 23	:
:	:

The range of the valid data that can be set to these registers varies according to the setting of the UF0DSCL register. In addition to the descriptors listed in Table 12-7, descriptors peculiar to classes and vendors can also be stored.

If all the values are fixed, they can be stored in ROM.

Cautions 1. To rewrite these registers, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

2. Use the value defined by USB Specification Ver. 2.0 and the latest Class Specification as the set value.



Table 12-7. Data of UF0CIEn Register

(a) Configuration descriptor (9 bytes)

Offset	Field Name	Contents
0	bLength	Size of this descriptor
1	bDescriptorType	Descriptor type
2	wTotalLength	Lower value of the total number of bytes of Configuration, all Interface, and all Endpoint descriptors
3		Higher value of the total number of bytes of Configuration, all Interface, and all Endpoint descriptors
4	bNumInterface	Number of Interfaces
5	bConfigurationValue	Value to select this Configuration
6	iConfiguration	Index of string descriptor describing this Configuration
7	bmAttributes	Features of this Configuration (self-powered, without remote wakeup)
8	MaxPower	Maximum power consumption of this Configuration (unit: mA) Note

Note This value is expressed in 2mA units. (example: 50 = 100 mA)

(b) Interface descriptor (9 bytes)

Offset	Field Name	Contents
0	bLength	Size of this descriptor
1	bDescriptorType	Descriptor type
2	bInterfaceNumber	Value of this Interface
3	bAlternateSetting	Value to select alternative setting of Interface
4	bNumEndpoints	Number of usable Endpoints
5	bInterfaceClass	Class code
6	bInterfaceSubClass	Subclass code
7	bInterfaceProtocol	Protocol code
8	Interface	Index of string descriptor describing this Interface

(c) Endpoint descriptor (7 bytes)

Offset	Field Name	Contents
0	bLength	Size of this descriptor
1	bDescriptorType	Descriptor type
2	bEndpointAddress	Address/transfer direction of this Endpoint
3	bmAttributes	Transfer type
4	wMaxPaketSize	Lower value of maximum number of transfer data
5		Higher value of maximum number of transfer data
6	bInterval	Transfer interval

12.4.4 Peripheral control register

(1) USB function 0 buffer control register (UF0BC)

This register performs enable control and floating control on the input buffer of the USB function.

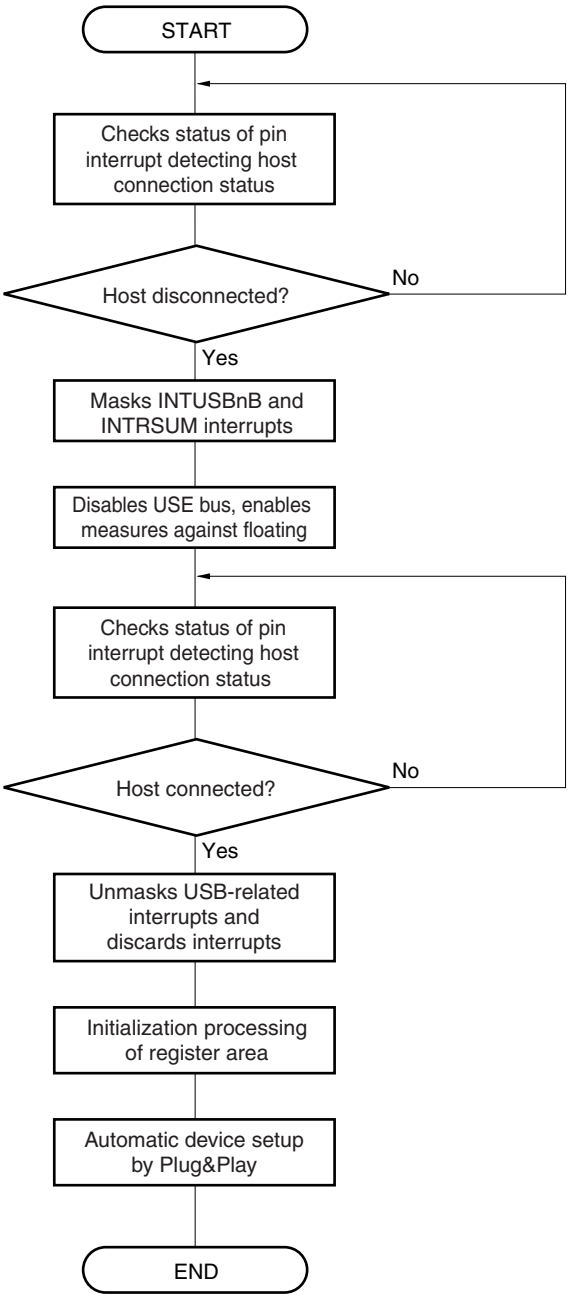
This register can be read or written in 8-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
UF0BC	0	0	0	0	0	0	UBFIEN	UBFIOR	FF8BH	00H

Bit position	Bit name	Function
1	UBFIEN	This bit controls use of the USB buffer. 1: Buffer valid 0: Buffer invalid **Caution Clear this bit to 0 when the USB is not used. If this bit is set to 1, a current of 3 mA (TYP.) constantly flows, regardless of whether the USB is used or not.**
0	UBFIOR	This bit controls use of floating measures of the USB buffer. 1: Disables floating measures 0: Enables floating measures This bit prevents erroneous recognition of Bus Reset, Suspend, and Resume due to an undefined value when a cable is not connected (when data input is floated). When this bit is set to 1, control the processing for floating by the VBUS signal (which recognizes cable connection).

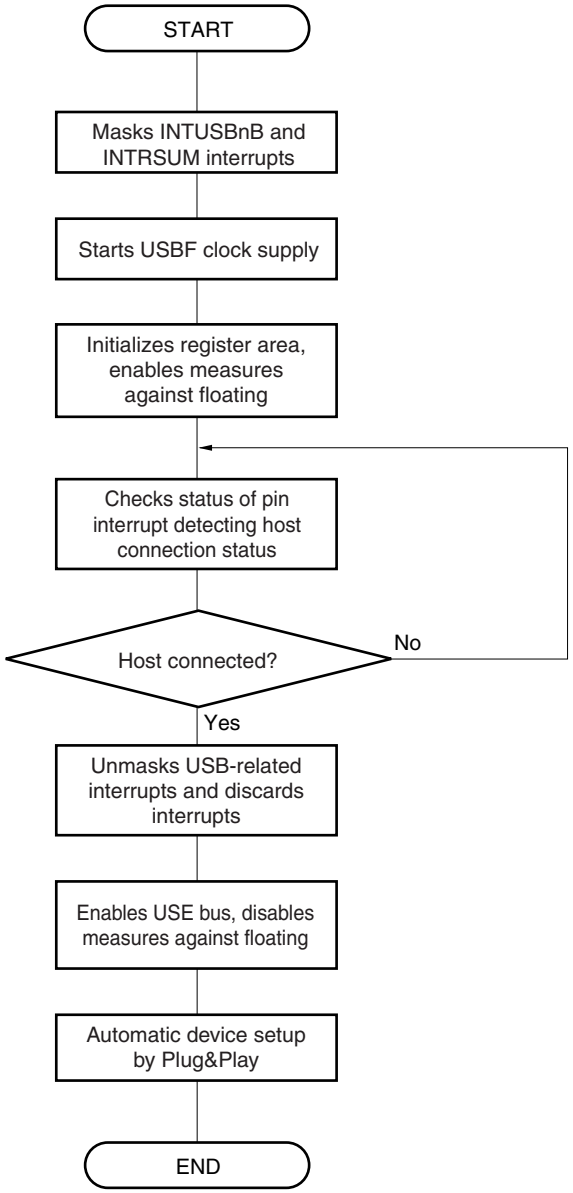
The following flowcharts illustrate the program execution when the host is disconnected and then reconnected, and the program execution when power is supplied.

Figure 12-7. Flowchart of Program When Host Is Disconnected and Then Reconnected



Remark n = 0 to 2

Figure 12-8. Flowchart of Program When Power Is Supplied



Remark n = 0 to 2

12.5 STALL Handshake or No Handshake

Errors of USBF are defined to be handled as follows.

Transfer Type	Transaction	Target Packet	Error Type	Function Response	Processing
Control transfer/ bulk transfer	IN/OUT/SETUP	Token	Endpoint not supported	No response	None
			Endpoint transfer direction mismatch	No response	None
			CRC error	No response	None
			Bit stuffing error	No response	None
	OUT/SETUP	Data	Timeout	No response	None
			PID check error	No response	None
			Unsupported PID (other than Data PID)	No response	None
			CRC error	No response	Discard received data
			Bit stuffing error	No response	Discard received data
	OUT	Data	Data PID mismatch	ACK	Discard received data
Control transfer (SETUP stage)	SETUP	Data	Overrun	No response	Discard received data
Control transfer (data stage)	OUT	Data	Overrun	No response[Note 1]	Set SNDSTL bit of UF0SDS register to 1 and discard received data
Control transfer (status stage)	OUT	Data	Overrun	ACK or no response[Note 2]	Set SNDSTL bit of UF0SDS register to 1 and discard received data
Bulk transfer	OUT	Data	Overrun	No response[Note 1]	Set EnHALT bit of UF0EnSL register (n = 0 to 2) to 1
Control transfer/ bulk transfer	IN	Handshake	PID check error	–	Hold transferred data and re-transfer data[Note 3]
			Unsupported PID (other than ACK PID)	–	Hold transferred data and re-transfer data[Note 3]
			Timeout	–	Hold transferred data and re-transfer data[Note 3]

Notes 1. A STALL response is made to re-transfer by the host.

2. An ACK response is made if the transfer data is of less than MaxPacketSize and the data received in the status stage is discarded. If MaxPacketSize is exceeded, no response is made, the SNDSTL bit of the UF0SDS register is set to 1, and the received data is discarded.

3. If an OUT transaction indicating a change from the data stage to the status stage is received during control transfer, an error is not handled and it is assumed that reception has been correctly completed.

Cautions 1. It is judged by the Alternative Setting number currently set whether the target endpoint is valid or invalid.

2. For the response to the request included in control transfer to/from Endpoint0, see 12.3 Requests.

12.6 Register Values in Specific Status

Table 12-8. Register Values in Specific Status (1/2)

Register Name	After CPU Reset ($\overline{\text{RESET}}$)	After Bus Reset
UF0E0N register	00H	Value is held.
UF0E0NA register	00H	Value is held.
UF0EN register	00H	Value is held.
UF0ENM register	00H	Value is held.
UF0SDS register	00H	Value is held.
UF0CLR register	00H	Value is held.
UF0SET register	00H	Value is held.
UF0EPS0 register	00H	Value is held.
UF0EPS1 register	00H	Value is held.
UF0EPS2 register	00H	Value is held.
UF0IS0 register	00H	Value is held.
UF0IS1 register	00H	Value is held.
UF0IS2 register	00H	Value is held.
UF0IS3 register	00H	Value is held.
UF0IS4 register	00H	Value is held.
UF0IM0 register	00H	Value is held.
UF0IM1 register	00H	Value is held.
UF0IM2 register	00H	Value is held.
UF0IM3 register	00H	Value is held.
UF0IM4 register	00H	Value is held.
UF0IC0 register	FFH	Value is held.
UF0IC1 register	FFH	Value is held.
UF0IC2 register	FFH	Value is held.
UF0IC3 register	FFH	Value is held.
UF0IC4 register	FFH	Value is held.
UF0FIC0 register	00H	Value is held.
UF0FIC1 register	00H	Value is held.
UF0DEND register	00H	Value is held.
UF0GPR register	00H	Value is held.
UF0MODC register	00H	Value is held.
UF0MODS register	00H	Bit 2 (CONF): Cleared (0), Other bits: Value is held.
UF0AIFN register	00H	Value is held.
UF0AAS register	00H	Value is held.
UF0ASS register	00H	00H
UF0E1IM register	00H	Value is held.
UF0E2IM register	00H	Value is held.

Table 12-8. Register Values in Specific Status (2/2)

Register Name	After CPU Reset ($\overline{\text{RESET}}$)	After Bus Reset
UF0E0R register	Undefined[Note 1]	Value is held.
UF0E0L register	00H	Value is held.
UF0E0ST register	00H	00H
UF0E0W register	Undefined[Note 1]	Value is held.
UF0BO1 register	Undefined[Note 1]	Value is held.
UF0BO1L register	00H	Value is held.
UF0BI1 register	Undefined[Note 1]	Value is held.
UF0DSTL register	00H	00H
UF0E0SL register	00H	00H
UF0E1SL register	00H	00H
UF0E2SL register	00H	00H
UF0ADRS register	00H	00H
UF0CNF register	00H	00H
UF0IF0 register	00H	00H
UF0IF1 register	00H	00H
UF0IF2 register	00H	00H
UF0IF3 register	00H	00H
UF0IF4 register	00H	00H
UF0DSCL register	00H	Value is held.
UF0DDn register (n = 0 to 17)	**Note 2**	**Note 2**
UF0CIEn register (n = 0 to 255)	**Note 2**	**Note 2**

Notes 1. This register can be cleared to 0 by the $\overline{\text{RESET}}$ signal because its write pointer, counter, and read pointer are cleared to 0 when the $\overline{\text{RESET}}$ signal becomes active, in the same manner as clearing by the UF0FICn register, as the register is controlled by FIFO.

2. This register cannot be cleared to 0. Because data can be written to it by FW, however, any value can be written to the register (before doing so, however, be sure to set the EP0NKA bit of the UF0E0NA register to 1).

12.7 FW Processing

The following FW processing is performed.

- Setting processing on device side for the SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, and CLEAR_FEATURE requests during enumeration processing
- Analysis and processing of XXXXStandard, XXXXClass, and XXXXVendor requests not subject to automatic processing
- Reading data following bulk-transferred OUT token from receive buffer
- Writing data to be returned in response to bulk-transferred IN token

The following table lists the requests supported by FW.

Table 12-9. FW-Supported Standard Requests

Request	Reception Side	Processing/ Frequency	Explanation
CLEAR_FEATURE	Interface	Automatic STALL response	It is considered that this request does not come to Interface because there is no function selector value, though it is reserved for bmRequestType. When this request is received, the hardware makes an automatic STALL response.
SET_FEATURE	Interface	Automatic STALL response	It is considered that this request does not come to Interface because there is no function selector value, though it is reserved for bmRequestType. When this request is received, the hardware makes an automatic STALL response.
GET_DESCRIPTOR	String	FW	Returns the string descriptor. When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and writes the data to be returned to the host, to the UF0E0W register.
SET_DESCRIPTOR	Device	FW	Rewrites the device descriptor. When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and the writes the data for the next control transfer (OUT) to the UF0DDn register (n = 0 to 17).
SET_DESCRIPTOR	Configuration	FW	Rewrites the configuration descriptor. When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and the writes the data for the next control transfer (OUT) to the UF0CIEn register (n = 0 to 255).
SET_DESCRIPTOR	String	FW	Rewrites the string descriptor. When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and loads the data for the next control transfer (OUT).
Other	NA	FW	When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and performs the necessary processing.

12.7.1 Initialization processing

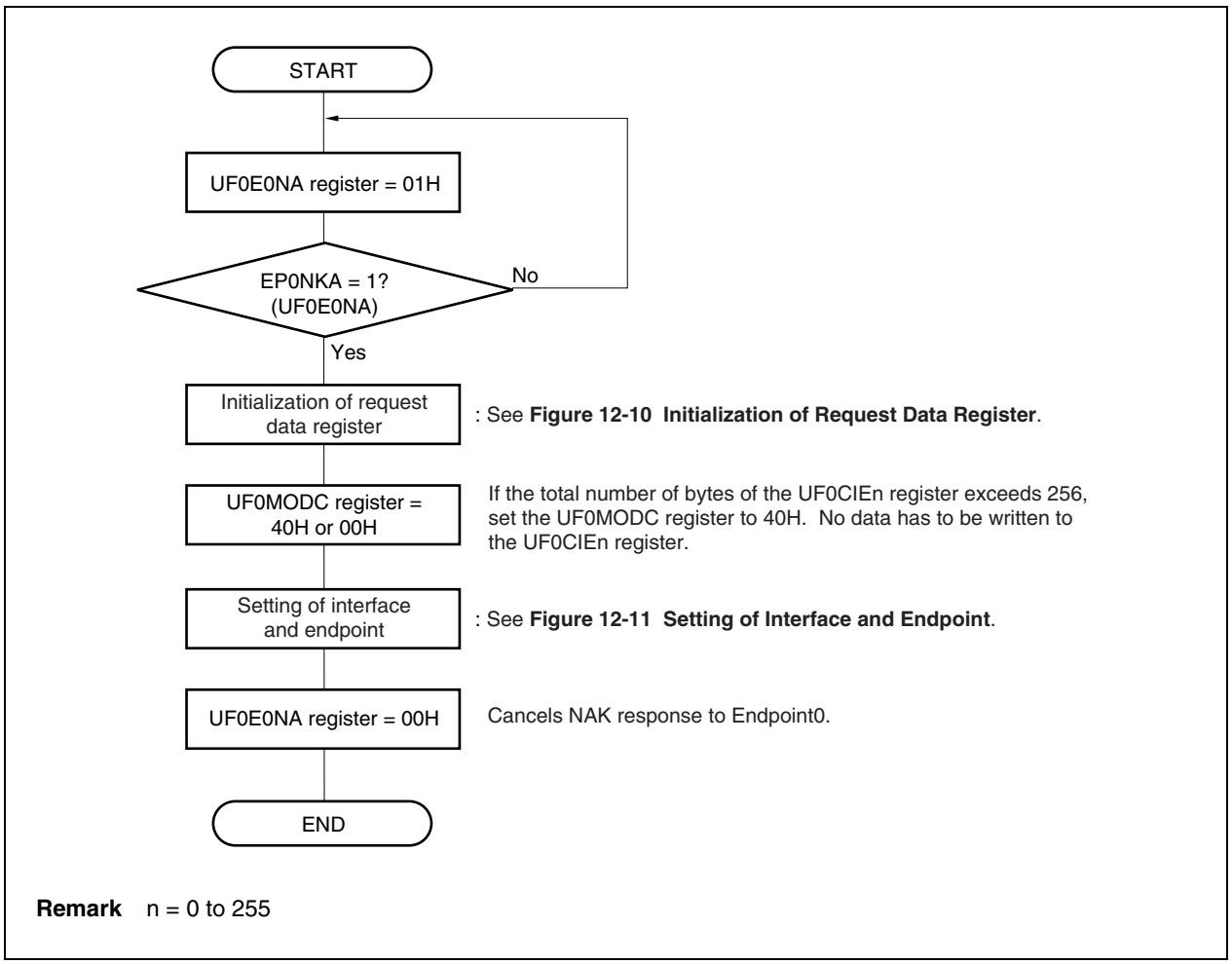
Initialization processing is executed in the following two ways.

- Initialization of request data register
- Setting of interrupt

When the request data register is initialized, data for the GET_XXXX request to which a value is to be automatically returned is written and an endpoint is allocated to an interface. In the interrupt settings, the interrupt sources that do not have to be checked can be masked by using the UF0IMn register (n = 0 to 4).

The following flowcharts illustrate the above processing.

Figure 12-9. Initializing Request Data Register



Remark n = 0 to 255

Figure 12-10. Initialization of Request Data Register Area

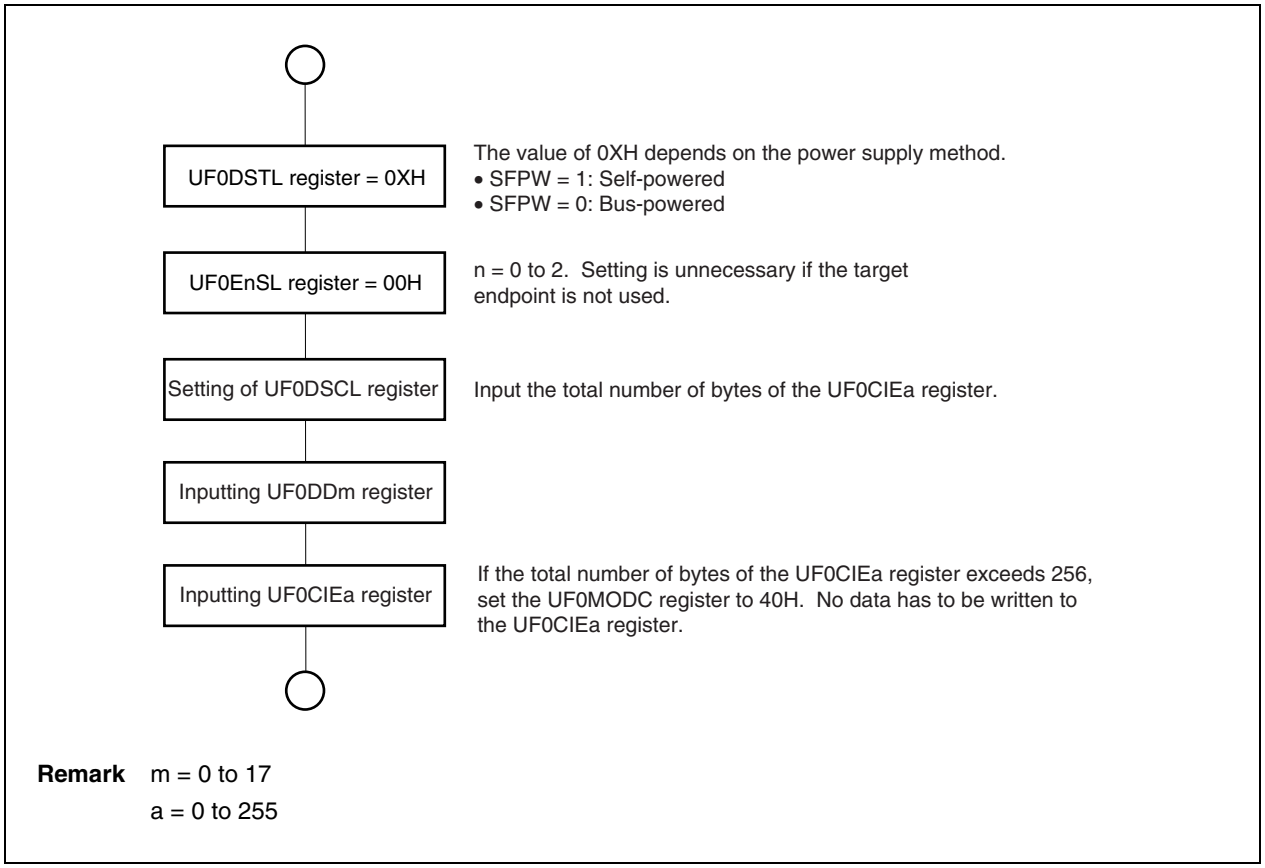


Figure 12-11. Setting of Interface and Endpoint

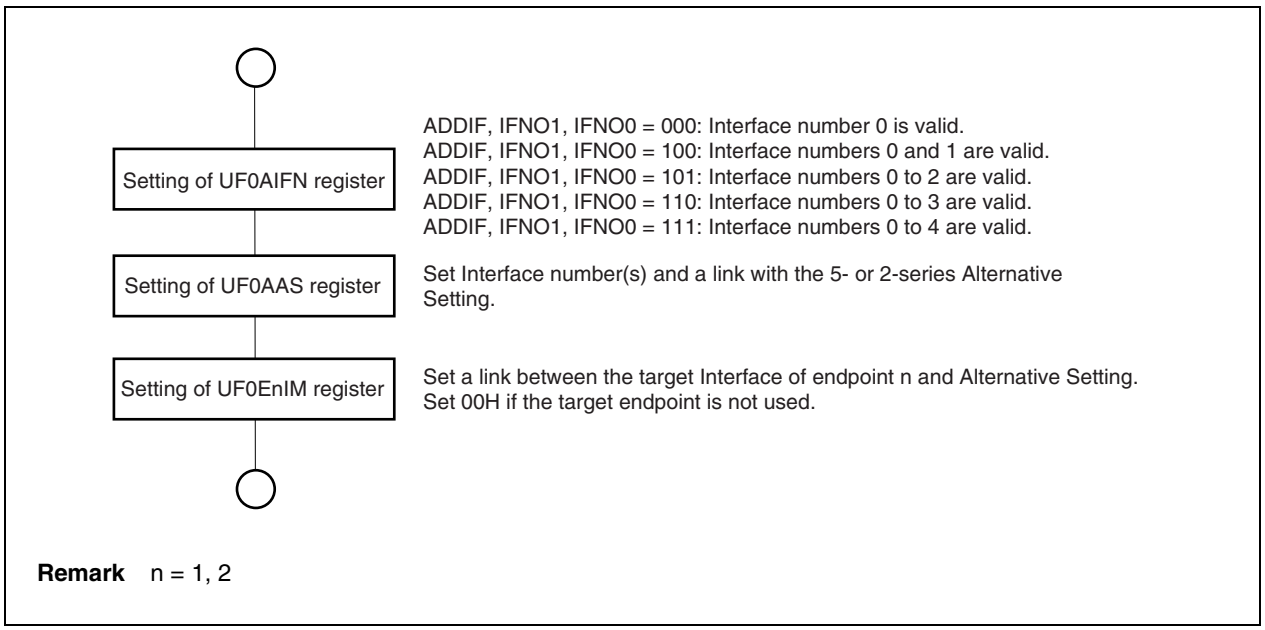
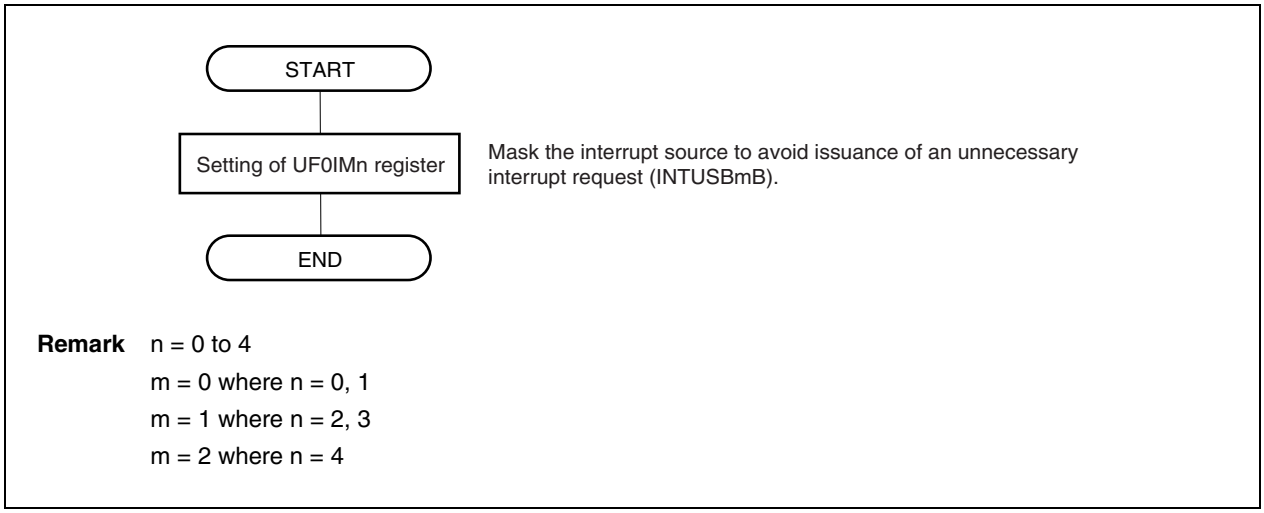


Figure 12-12. Setting of Interrupt

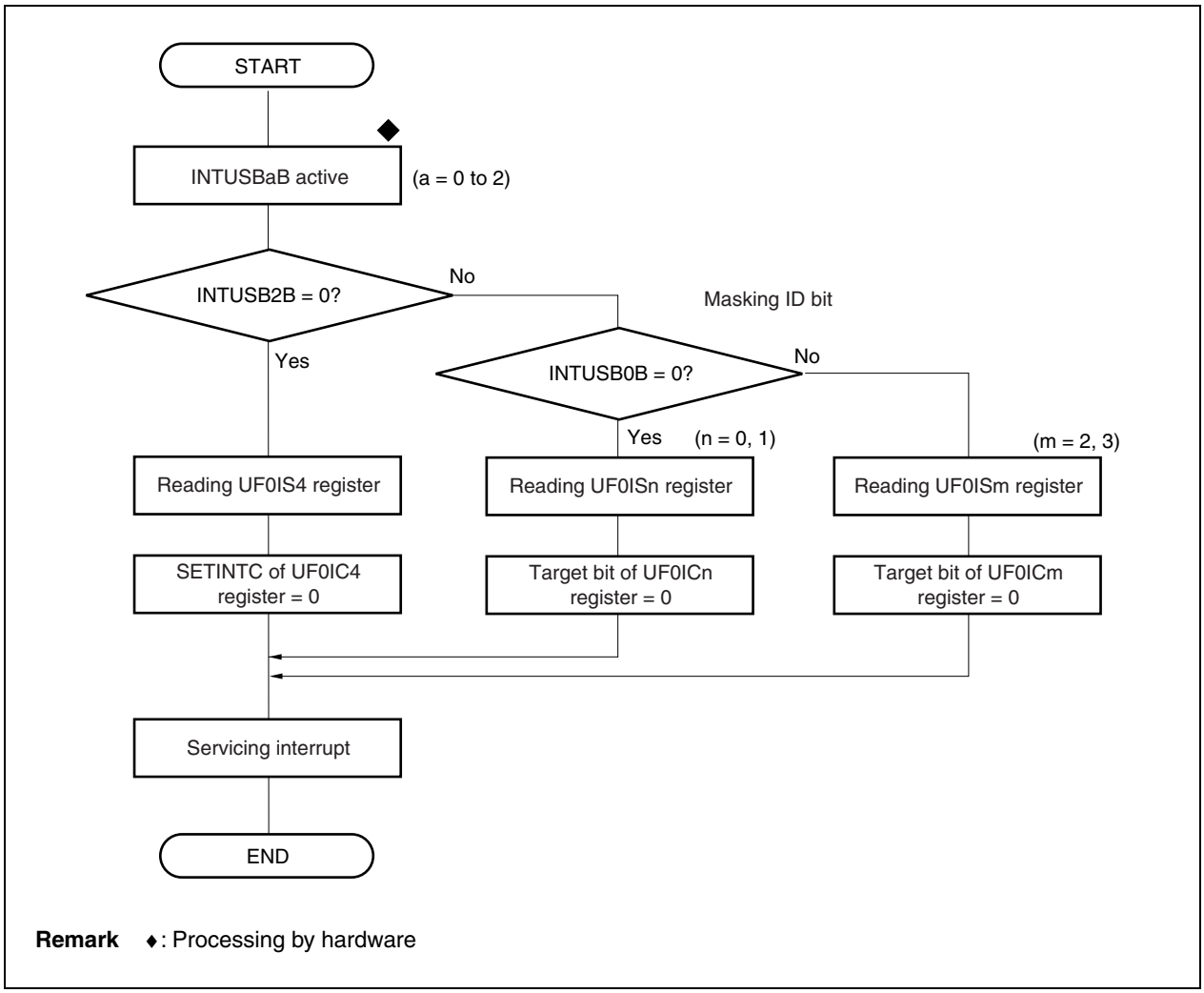


Remark n = 0 to 4
m = 0 where n = 0, 1
m = 1 where n = 2, 3
m = 2 where n = 4

12.7.2 Interrupt servicing

The following flowchart illustrates how an interrupt is serviced.

Figure 12-13. Interrupt Servicing



The following bits of the UF0ISn register are automatically cleared by hardware when a given condition is satisfied (n = 1 to 3).

- E0INDT, E0ODT, SUCES, STG, and CPUDEC bits of UF0IS1 register
- BKI1DT bit of UF0IS2 register
- BKO1FL and BKO1DT bits of UF0IS3 register

Because clearing an interrupt source by the UF0ICn register is given a lower priority than setting an interrupt source by hardware, the interrupt source may not be cleared depending on the timing (n = 0 to 4).

12.7.3 USB main processing

USB main processing involves processing USB transactions. The types of transactions to be processed are as follows.

- Fully automatically processed request for control transfer
- Automatically processed requests for control transfer
  (SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, CLEAR_FEATURE)
- CPUDEC request for control transfer
- Processing for bulk transfer (IN)
- Processing for bulk transfer (OUT)

Processing for endpoint n involves writing or reading for data transfer.

(1) Fully automatically processed request for control transfer

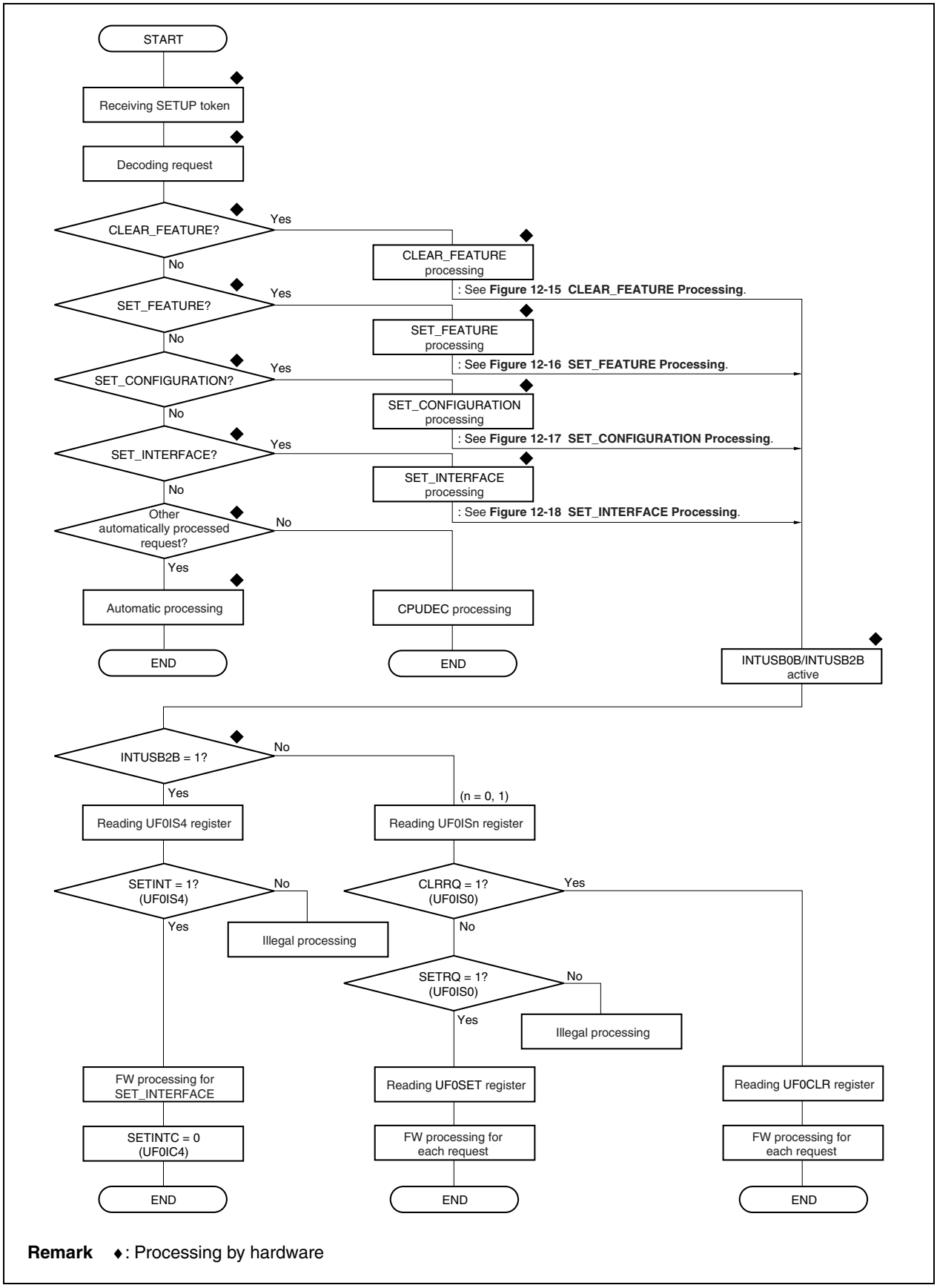
Because the fully automatically processed request for control transfer is executed by hardware, it cannot be referenced by FW. Therefore, FW does not have to perform any special processing for this request.

(2) Automatically processed requests for control transfer
(SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, CLEAR_FEATURE)

Processing to write a register for automatically processed requests for control transfer, such as SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, and CLEAR_FEATURE requests, is automatically executed by hardware, but an interrupt request is issued for recognition on the device side. This processing may be ignored if there is no special processing to be executed.

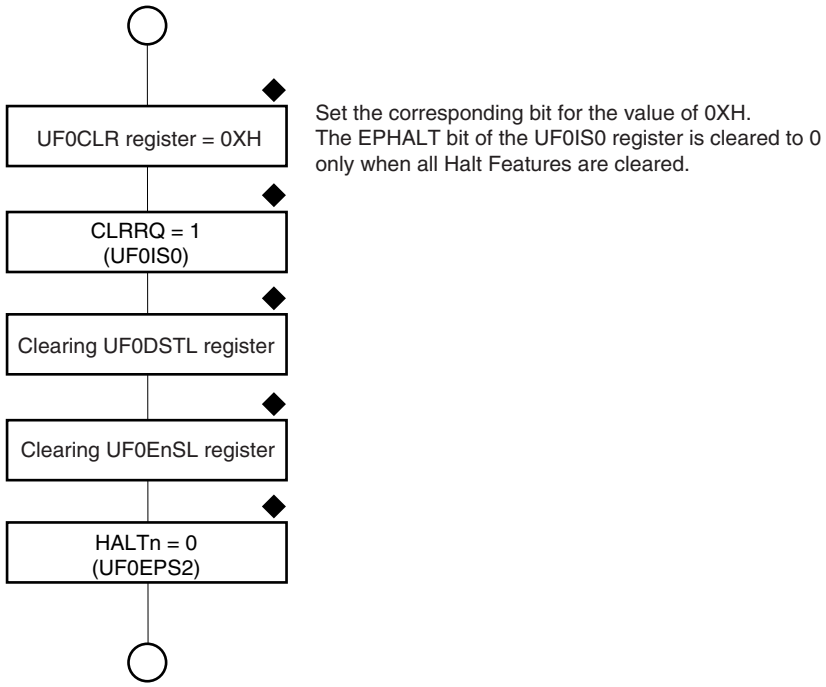
The flowcharts are shown below.

Figure 12-14. Automatically Processed Requests for Control Transfer



Remark ♦: Processing by hardware

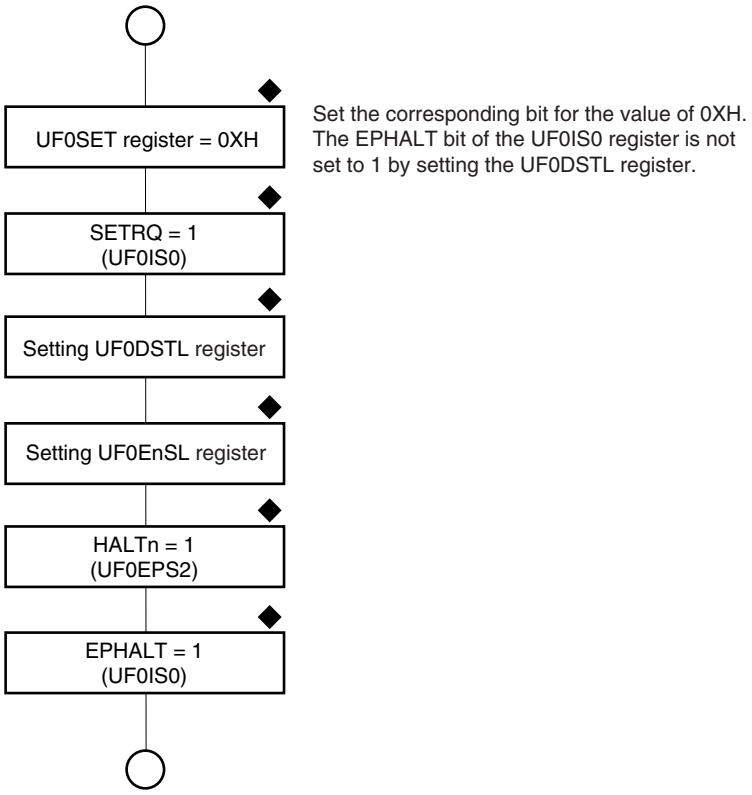
Figure 12-15. CLEAR_FEATURE Processing



Remarks 1. n = 0 to 2

2. ♦: Processing by hardware

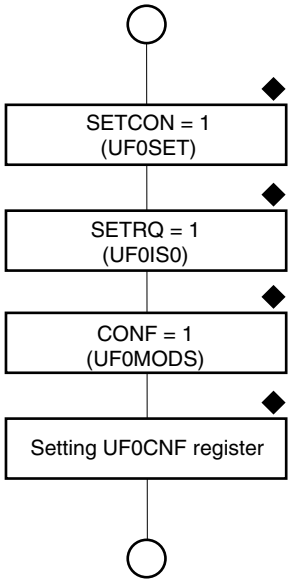
Figure 12-16. SET_FEATURE Processing



Remarks 1. n = 0 to 2

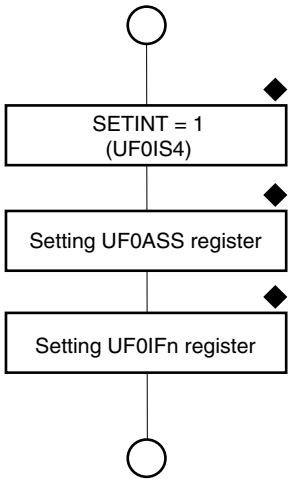
2. ♦: Processing by hardware

Figure 12-17. SET_CONFIGURATION Processing



Remark ♦: Processing by hardware

Figure 12-18. SET_INTERFACE Processing



Remarks 1. n = 0 to 4

2. ♦: Processing by hardware

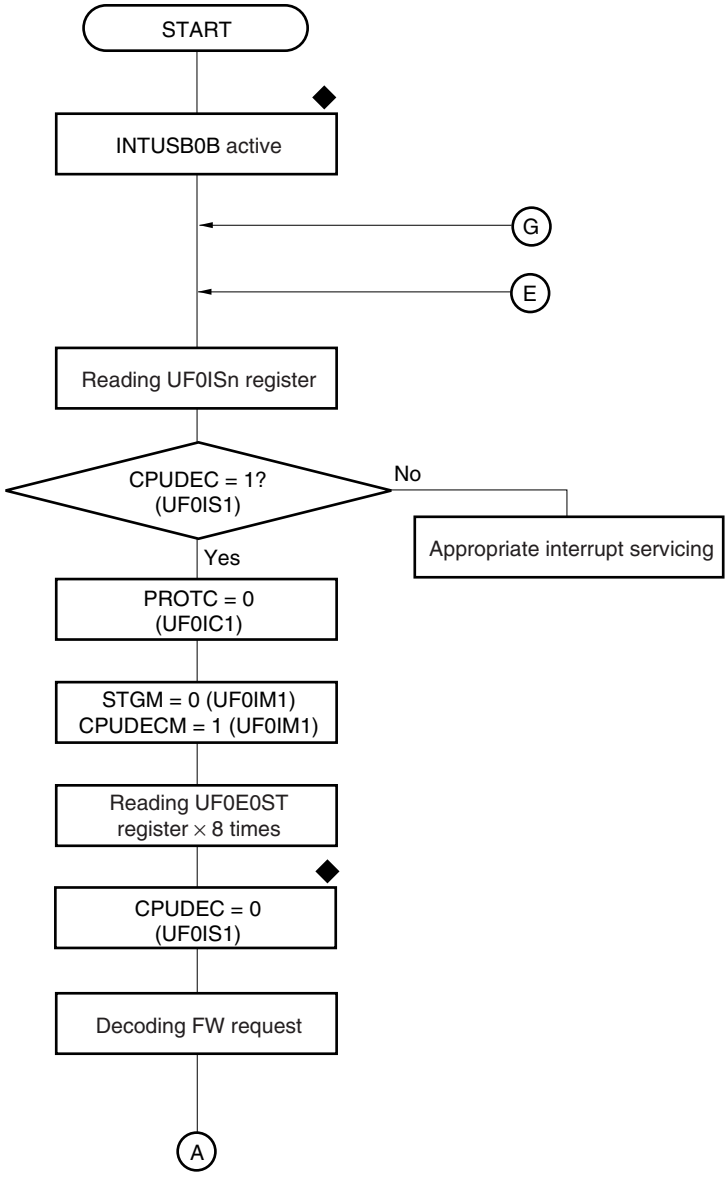
(3) CPUDEC request for control transfer

The CPUDEC request can be classified into three types of processing: control transfer (write), control transfer (read), and control transfer (without data). Control transfer (write) indicates a request that uses the OUT transaction in the data stage (e.g., SET_DESCRIPTOR), and control transfer (read) indicates a request that uses the IN transaction in the data stage (e.g., GET_DESCRIPTOR). Control transfer (without data) indicates a request that has no data stage (e.g., SET_CONFIGURATION).

The flowcharts are shown below.

Figure 12-19. CPUDEC Request for Control Transfer (1/12)

(a) Token phase (1/2)

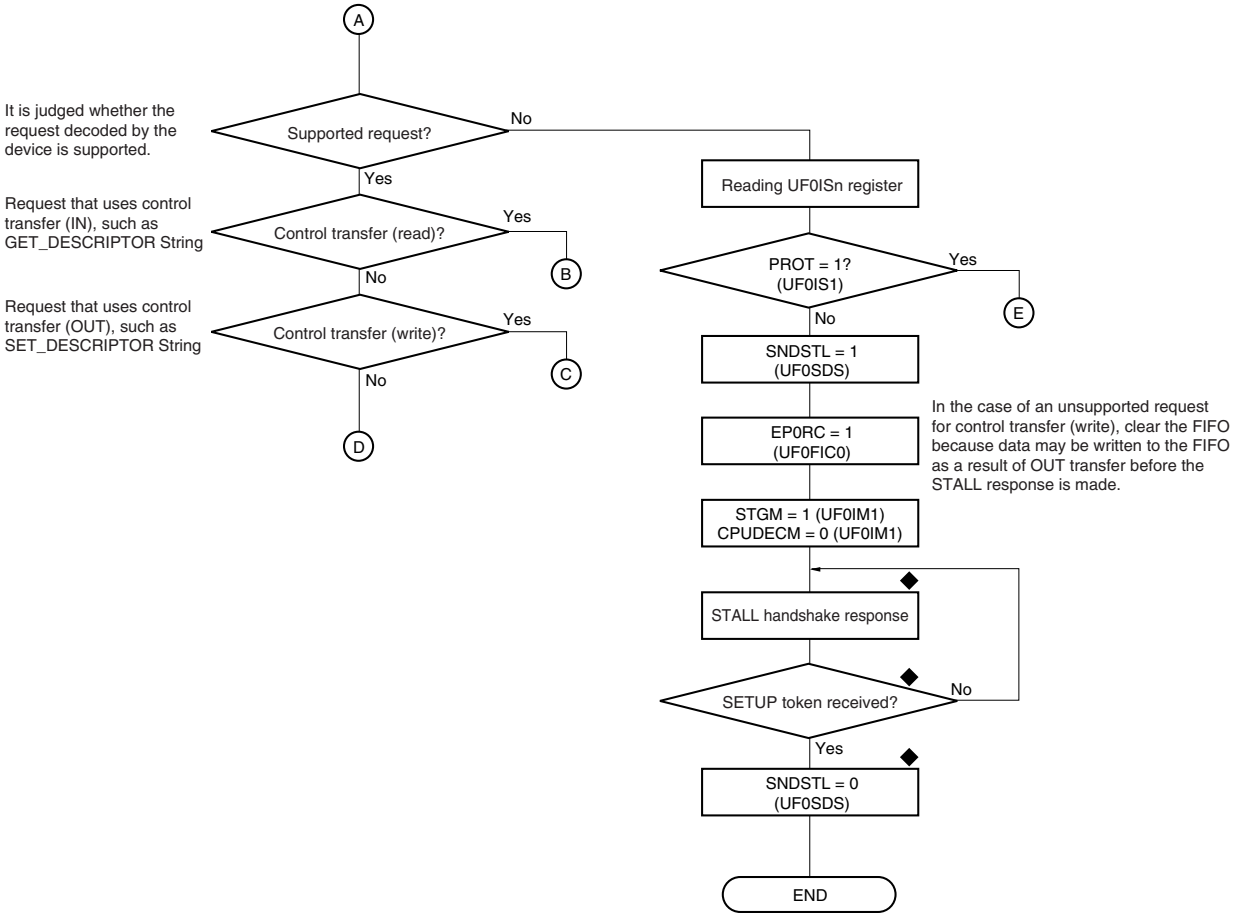


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (2/12)

(a) Token phase (2/2)

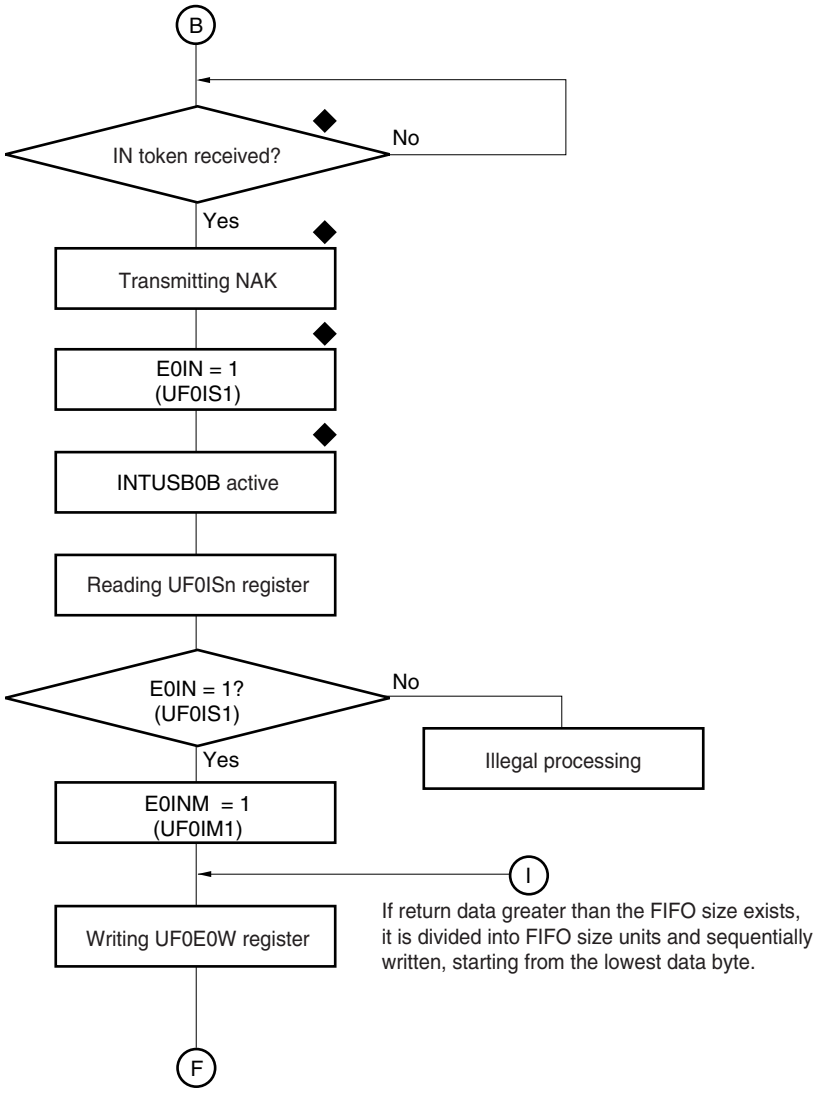


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (3/12)

(b) Control transfer (read) (1/4)

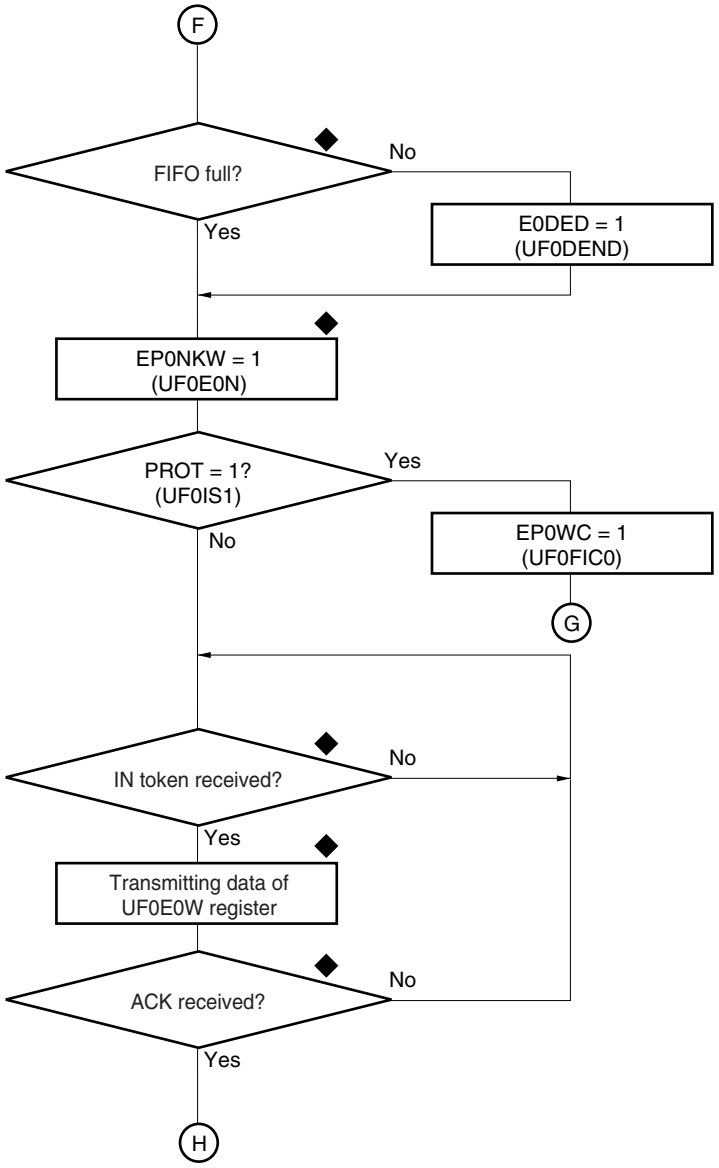


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (4/12)

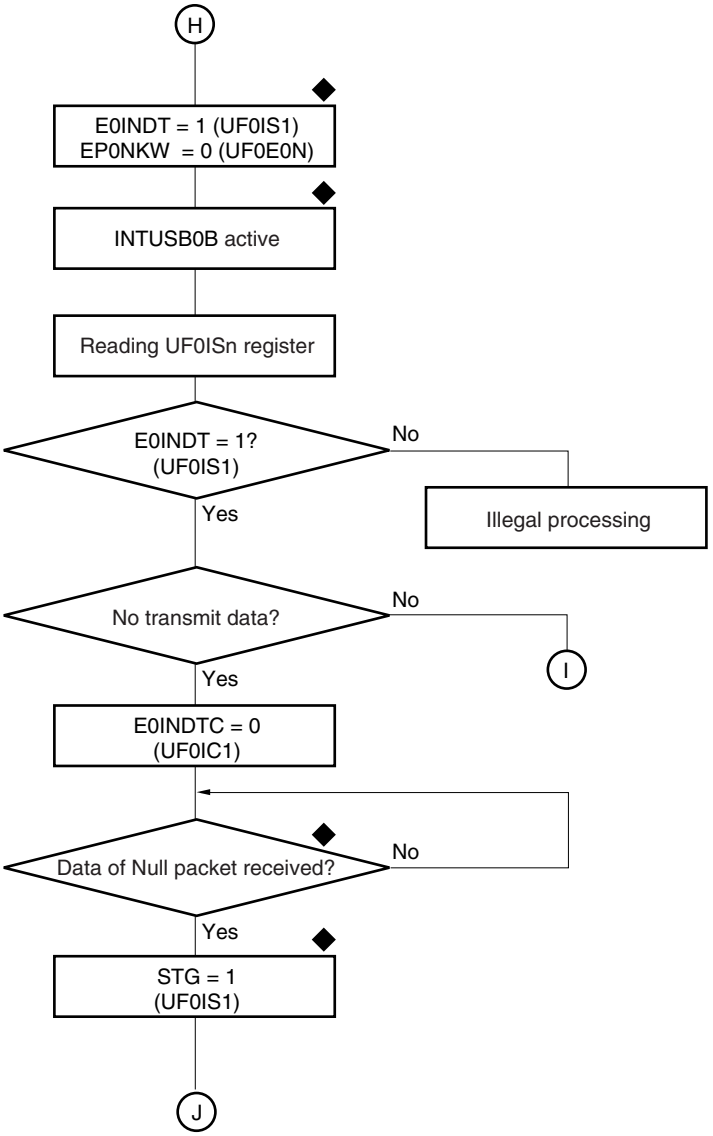
(b) Control transfer (read) (2/4)



Remark ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (5/12)

(b) Control transfer (read) (3/4)

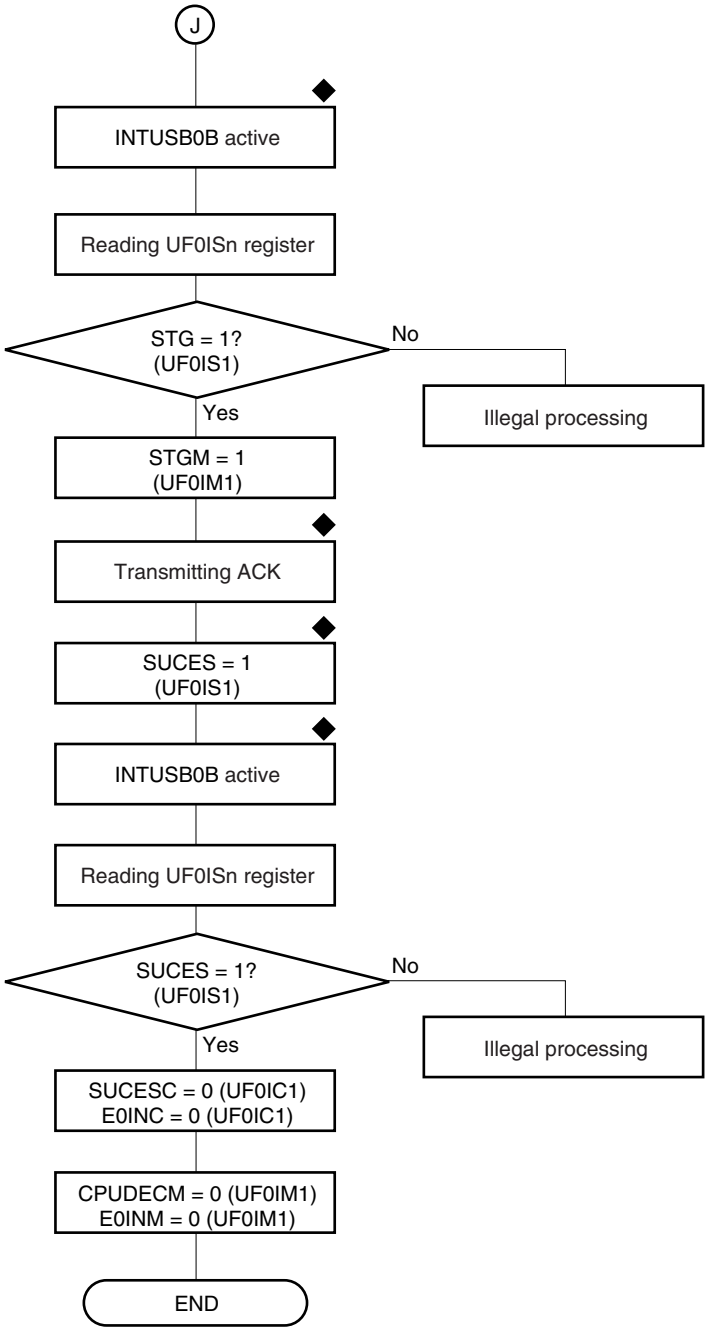


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (6/12)

(b) Control transfer (read) (4/4)

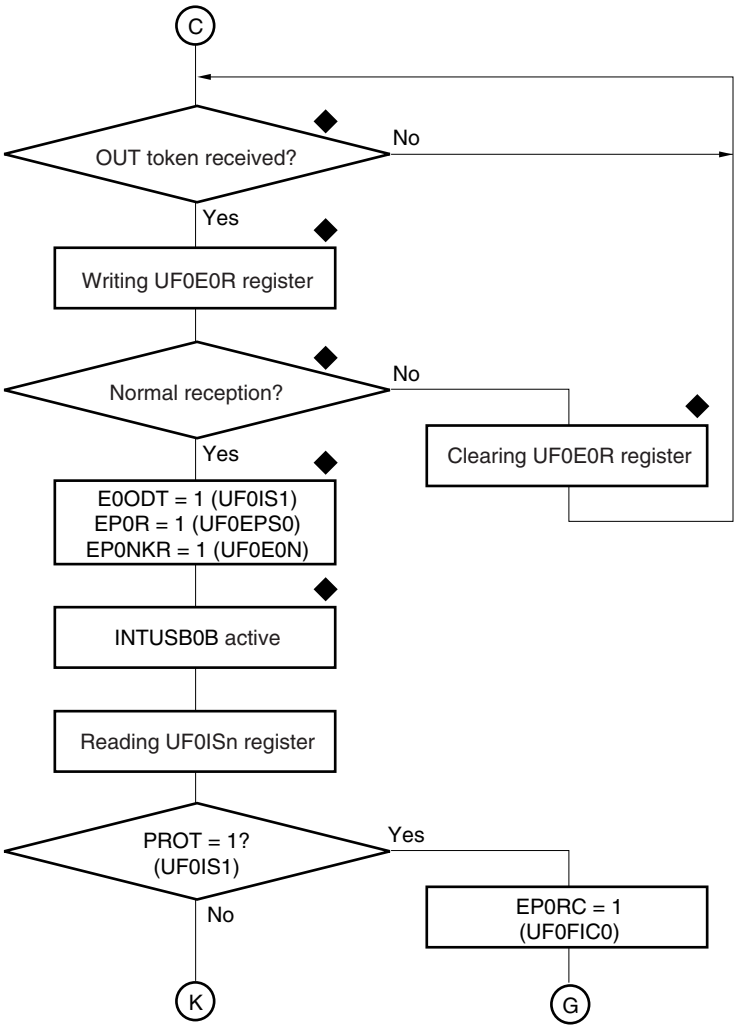


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (7/12)

(c) Control transfer (write) (1/4)

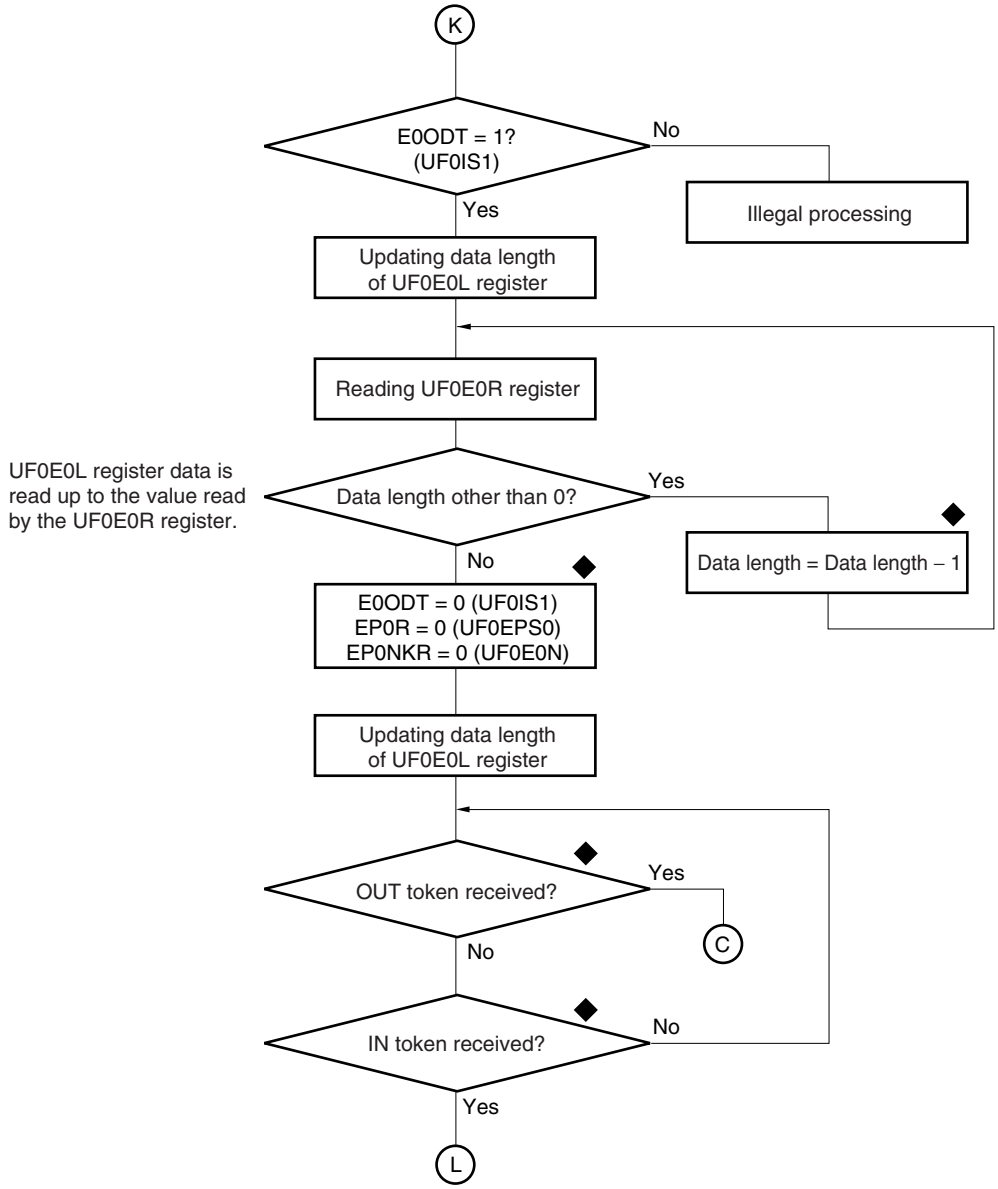


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (8/12)

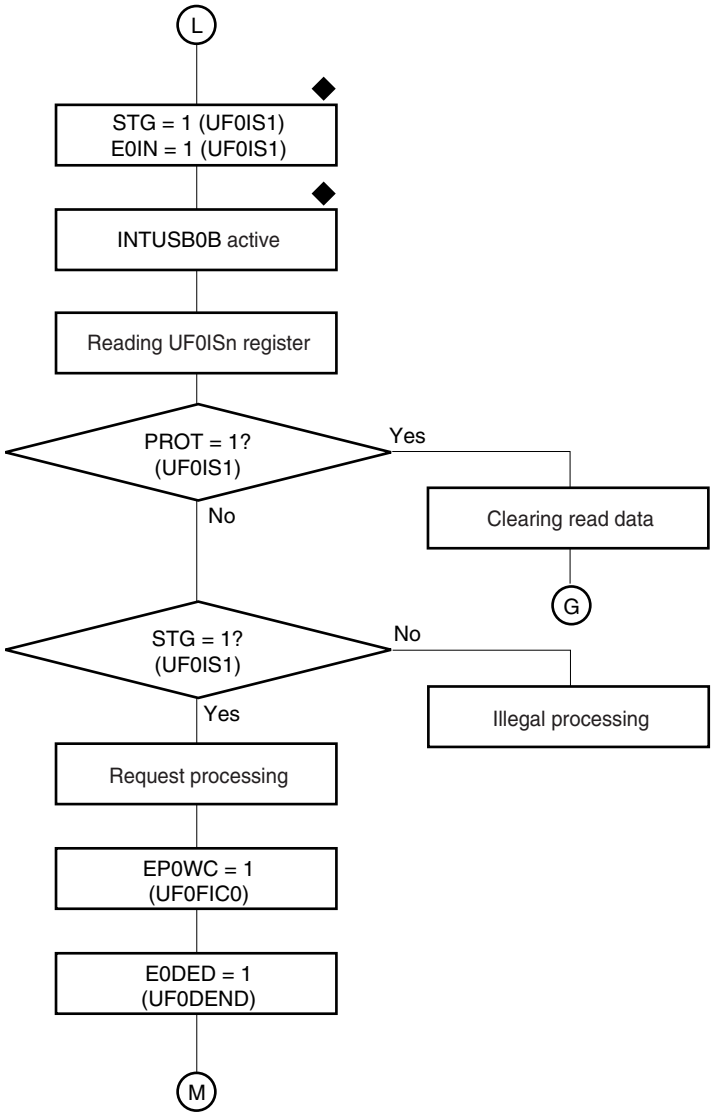
(c) Control transfer (write) (2/4)



Remark ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (9/12)

(c) Control transfer (write) (3/4)

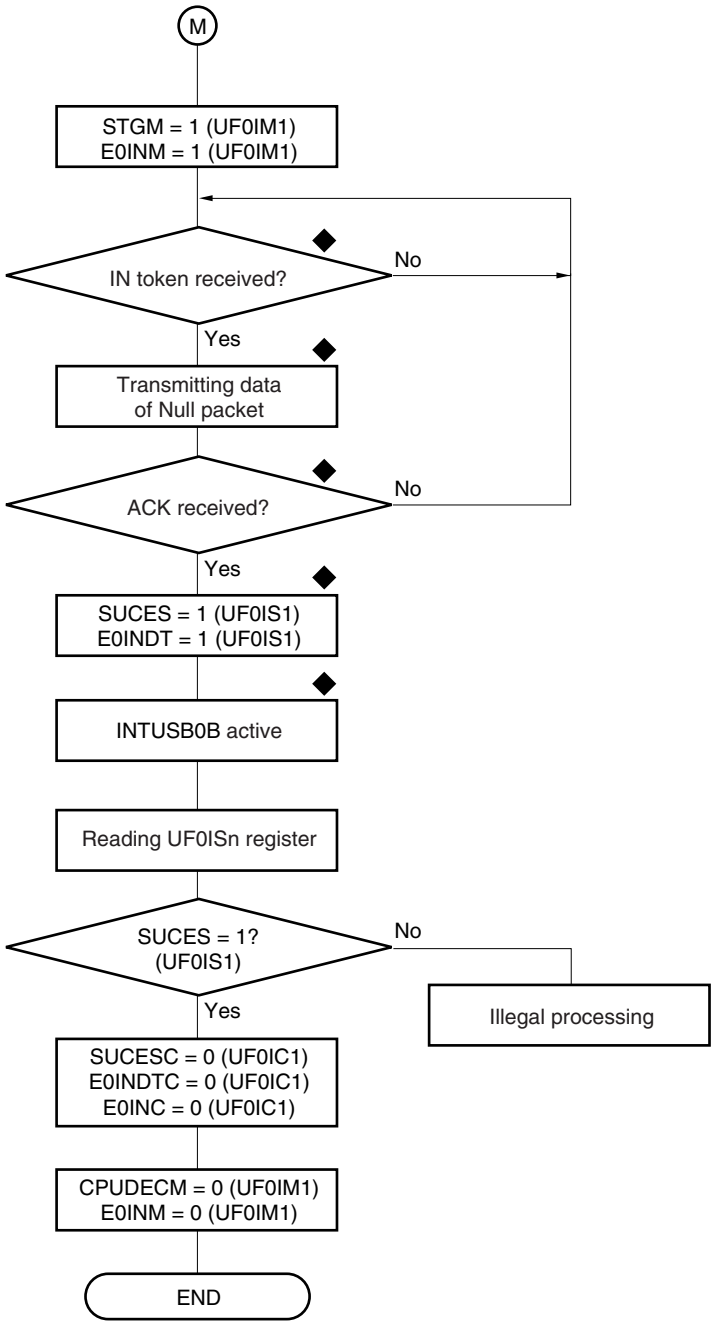


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (10/12)

(c) Control transfer (write) (4/4)

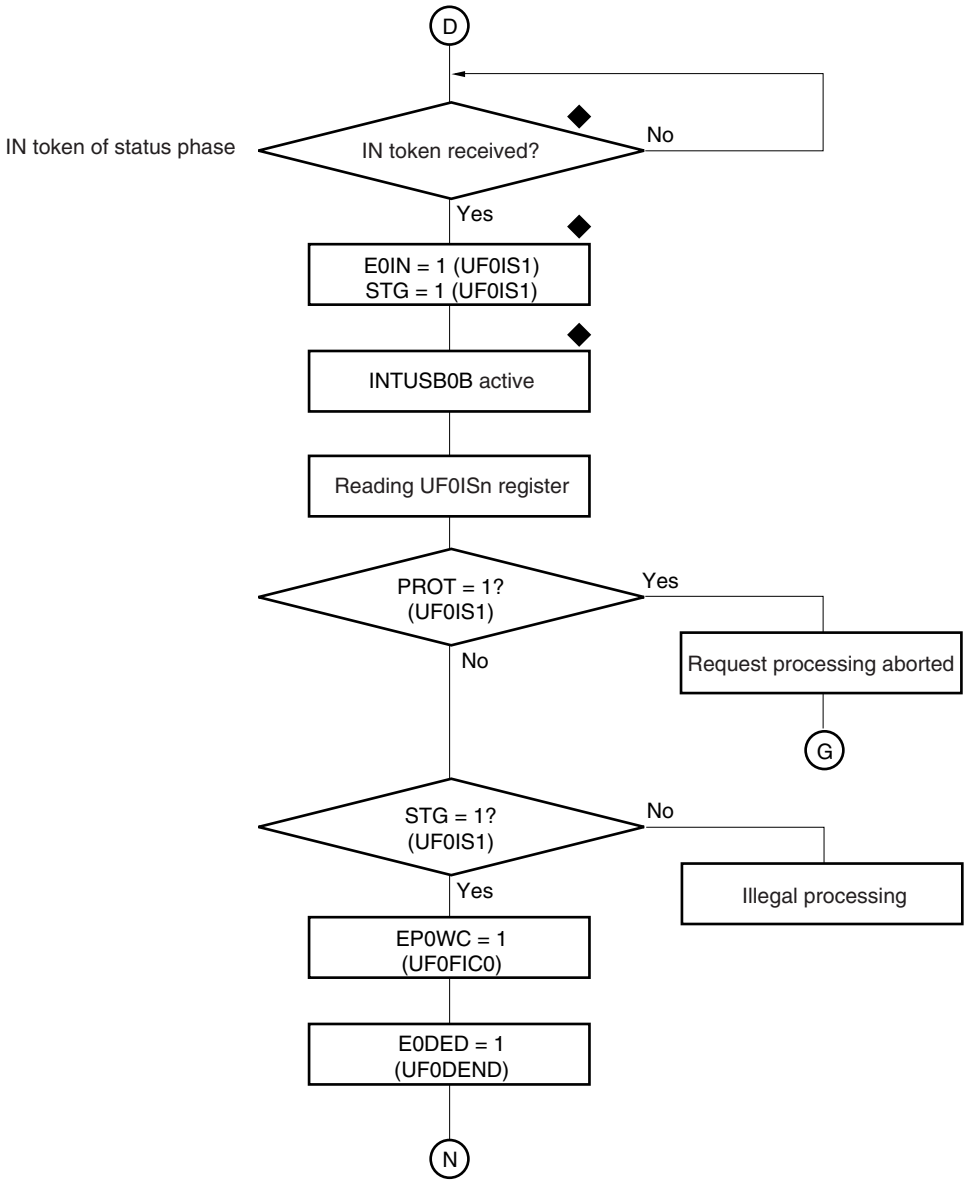


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (11/12)

(d) Control transfer (without data stage) (1/2)

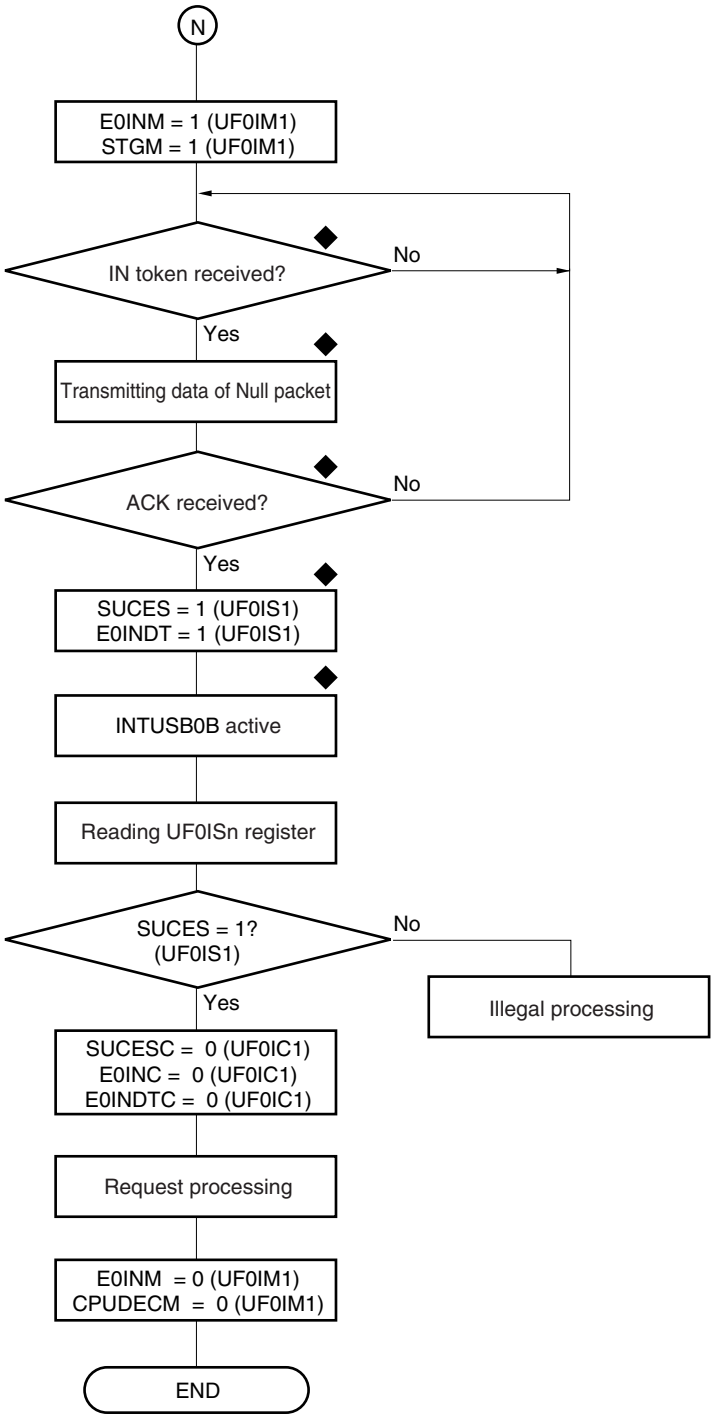


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-19. CPUDEC Request for Control Transfer (12/12)

(d) Control transfer (without data stage) (2/2)



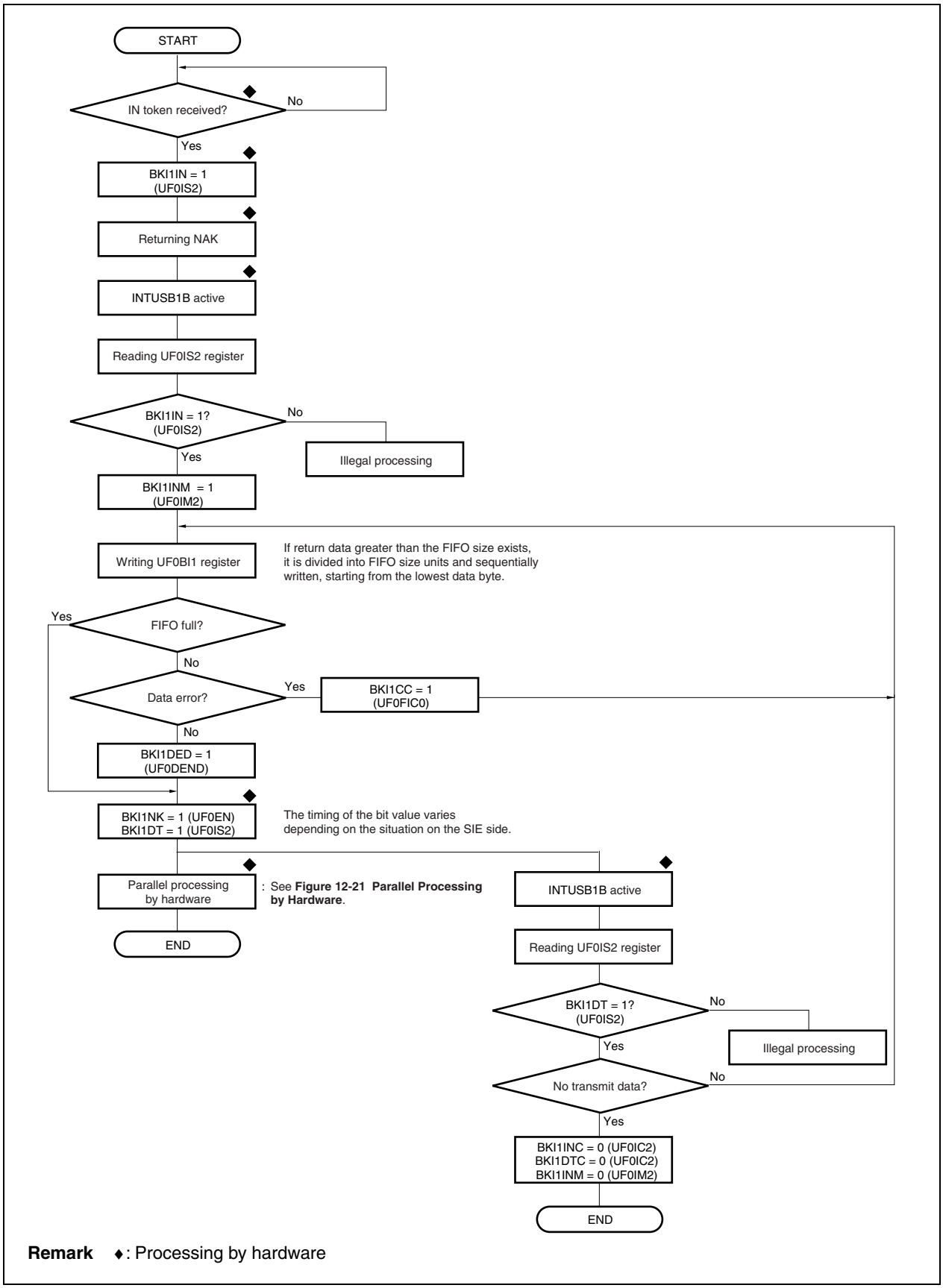
Remarks 1. n = 0, 1

2. ♦: Processing by hardware

(4) Processing for bulk transfer (IN)

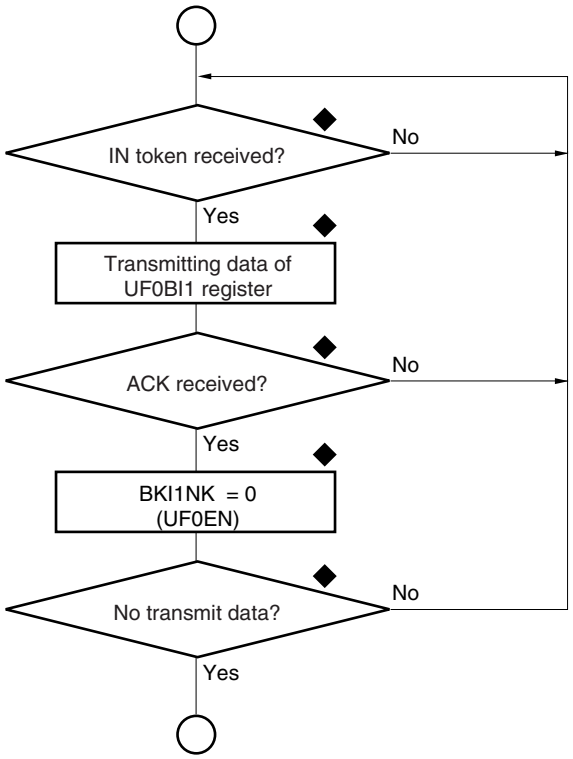
Bulk transfer (IN) is allocated to Endpoint1. The flowchart is shown below.

Figure 12-20. Processing for Bulk Transfer (IN)



Remark ♦: Processing by hardware

Figure 12-21. Parallel Processing by Hardware

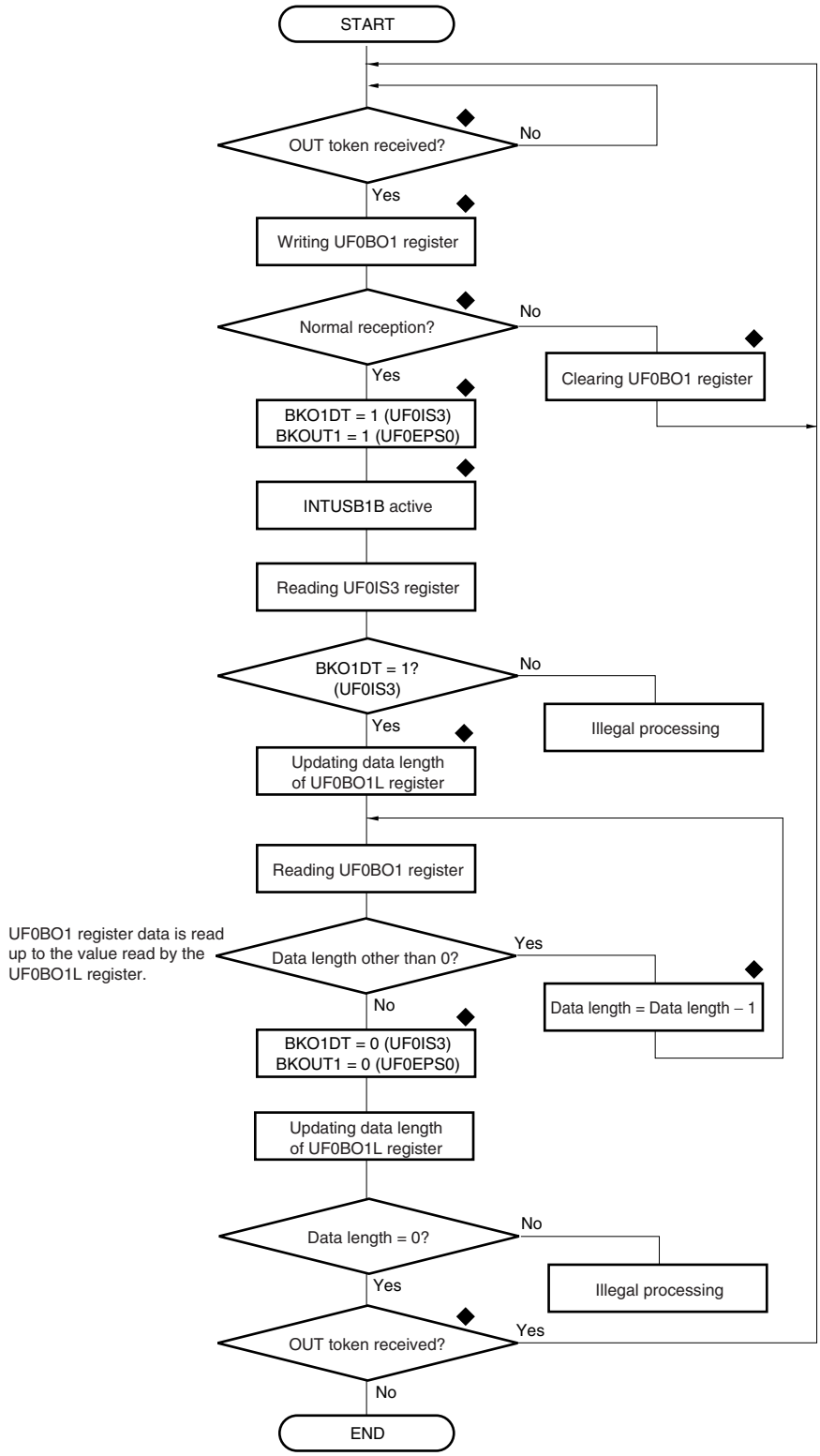


Remark ♦: Processing by hardware

(5) Processing for bulk transfer (OUT)

Bulk transfer (OUT) is allocated to Endpoint2. The flowchart is shown below.

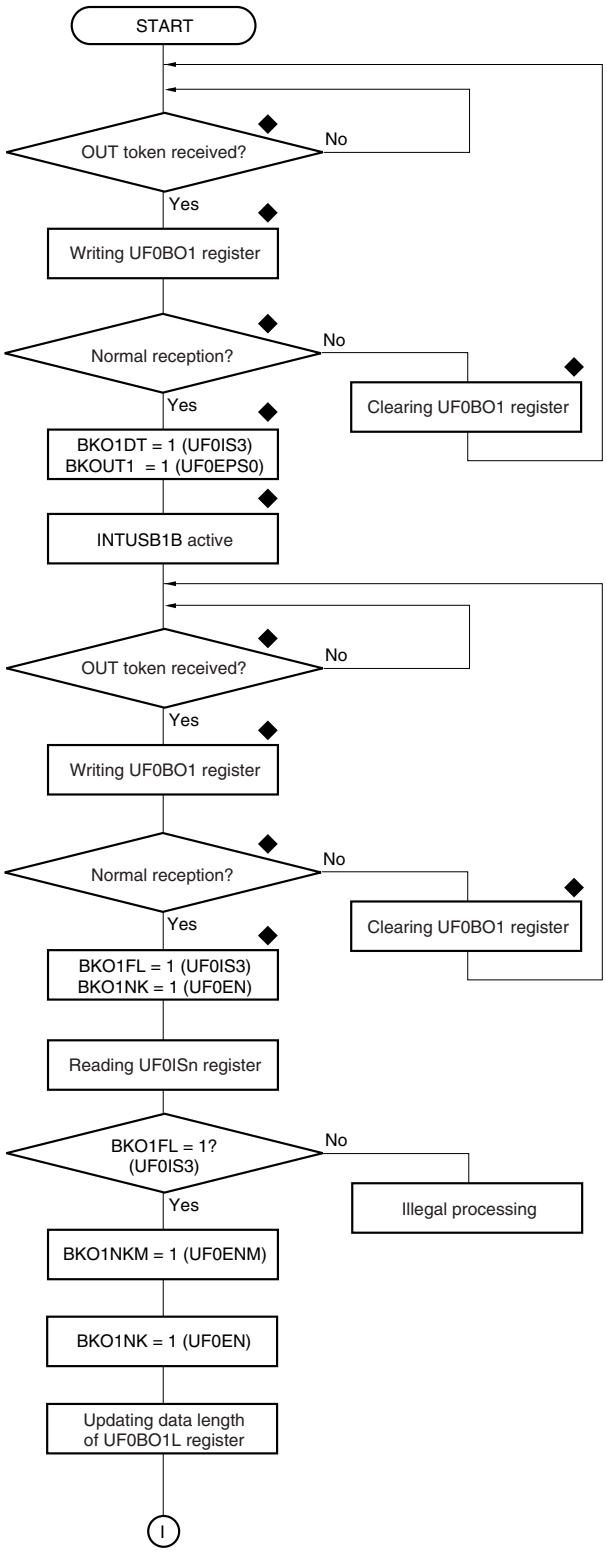
Figure 12-22. Normal Processing for Bulk Transfer (OUT)



Remark ♦: Processing by hardware

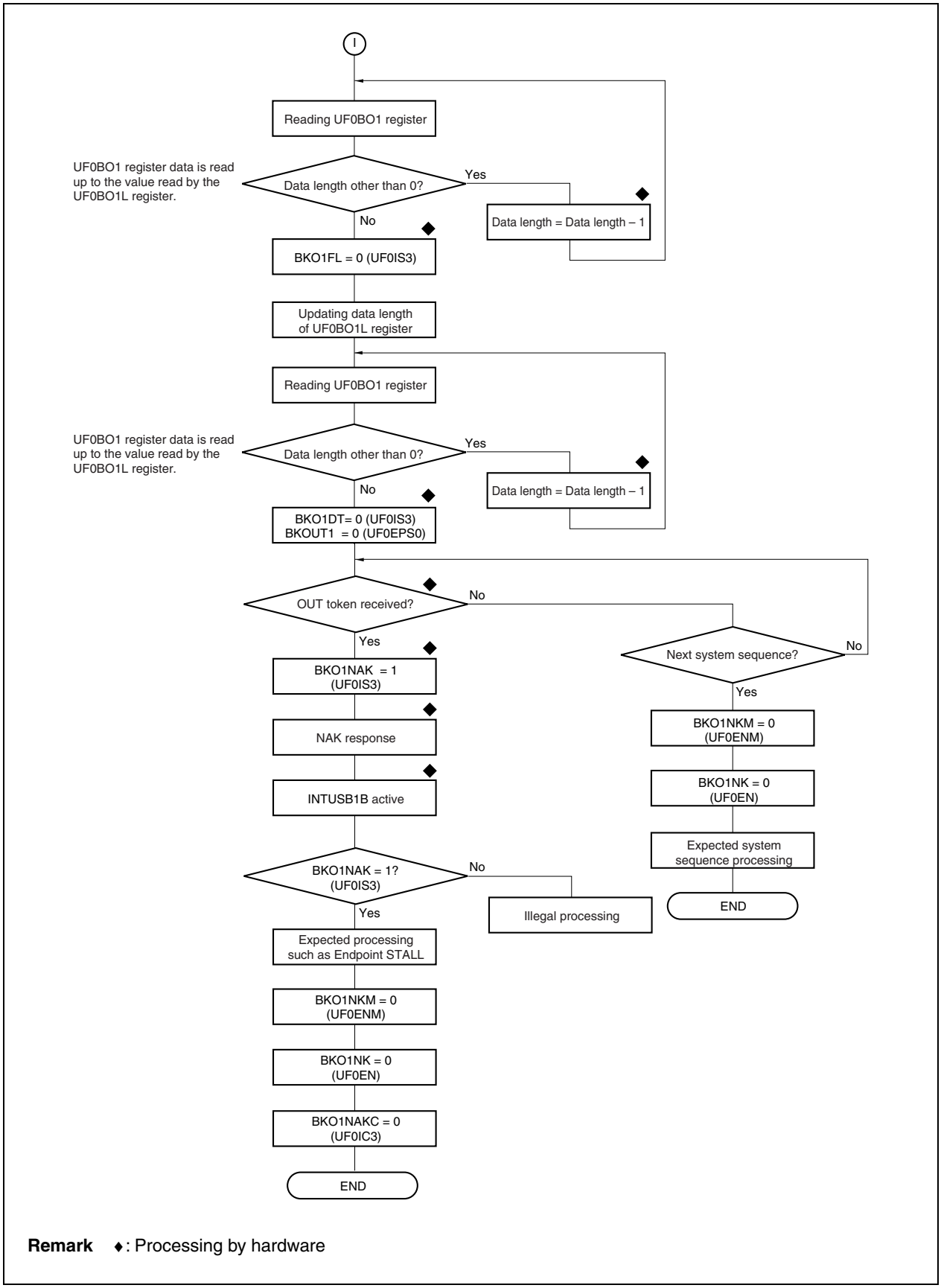
During bulk transfer (OUT), more data may be transmitted from the host than expected by the system. Endpoint2 for bulk transfer (OUT) of the μPD78F0730 consist of two 64-byte buffers so that NAK responses are suppressed as much as possible and data can be read from the CPU side even while the bus side is being accessed as the transfer rate of the USB bus increases. Consequently, if the host sends more data than expected by the system, up to 128 bytes of extra data may be automatically received in the worst case. In this case, change the control flow from that of the normal processing of Endpoint2 to the flow illustrated below when the quantity of data expected by the system has decreased to two packets.

Figure 12-23. Processing If More Data Than Expected by System Is Transmitted (1/2)



Remark ♦: Processing by hardware

Figure 12-23. Processing If More Data Than Expected by System Is Transmitted (2/2)



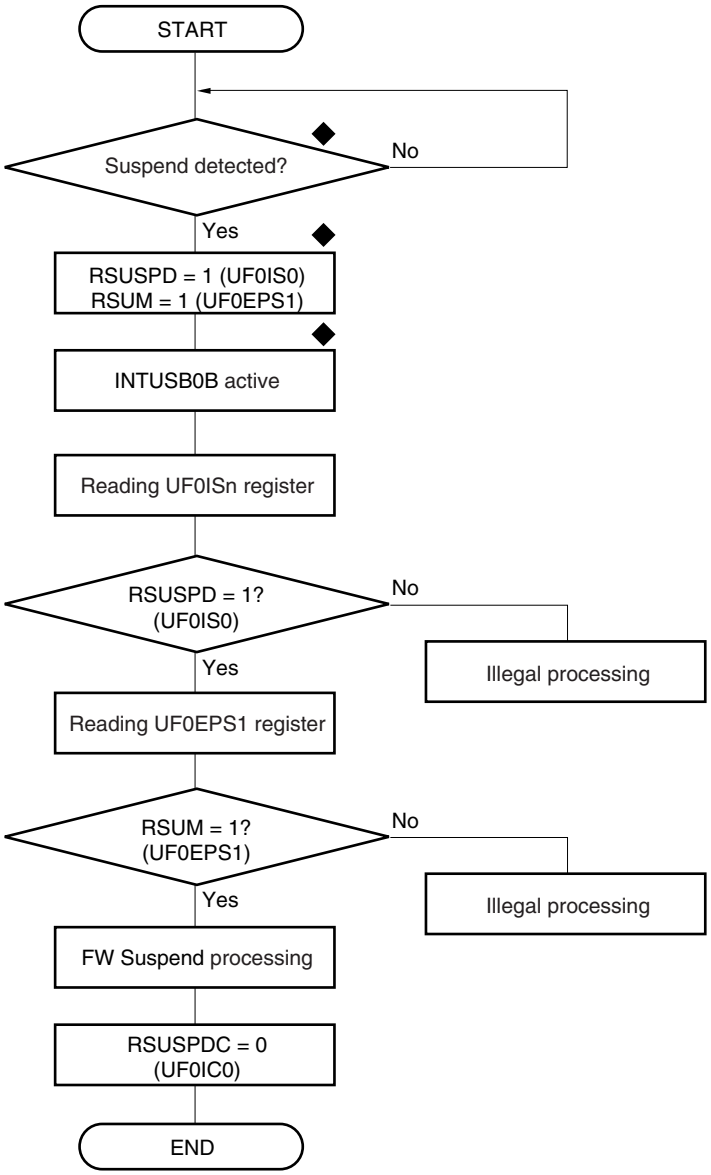
Remark ♦: Processing by hardware

12.7.4 Suspend/Resume processing

How Suspend/Resume processing is performed differs depending on the configuration of the system. One example is given below.

Figure 12-24. Example of Suspend/Resume Processing (1/3)

(a) Example of Suspend processing

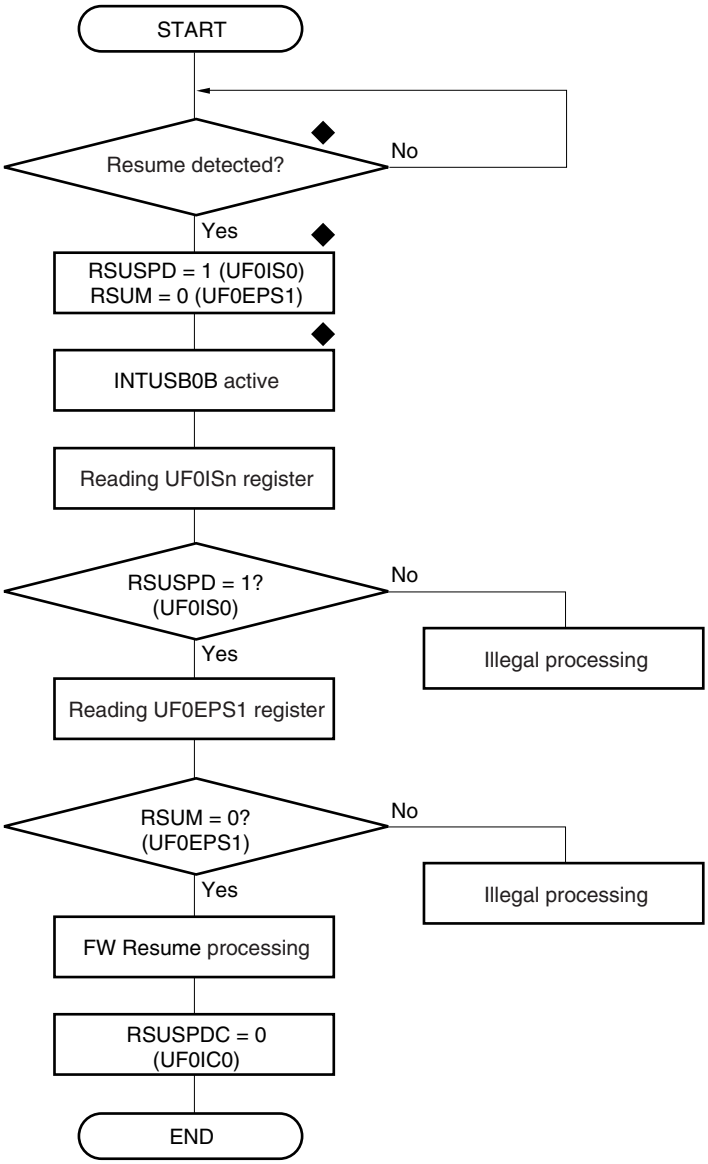


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-24. Example of Suspend/Resume Processing (2/3)

(b) Example of Resume processing

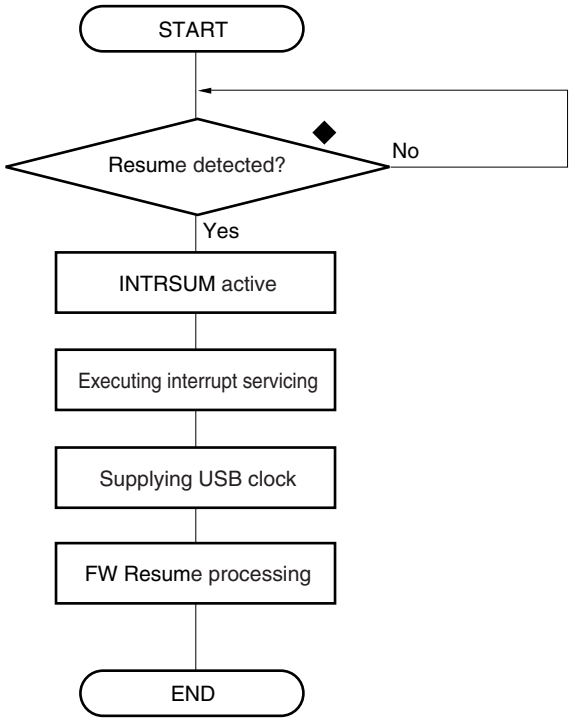


Remarks 1. n = 0, 1

2. ♦: Processing by hardware

Figure 12-24. Example of Suspend/Resume Processing (3/3)

(c) Example of Resume processing (when supply of USB clock to USBF is stopped)



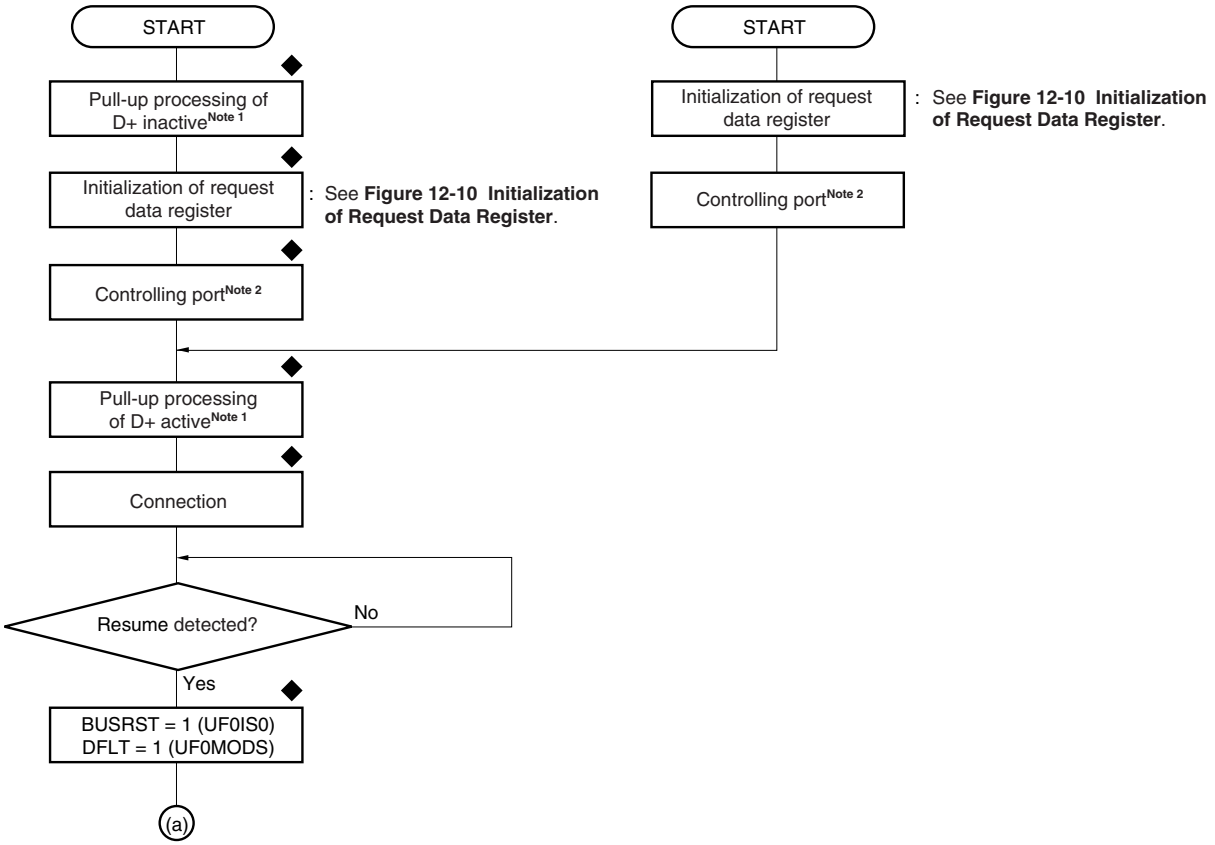
Remark ♦: Processing by hardware

12.7.5 Processing after power application

The processing to be performed after power application differs depending on the configuration of the system. One example is given below.

Figure 12-25. Example of Processing After Power Application/Power Failure (1/3)

(a) Processing after power application (1/2)



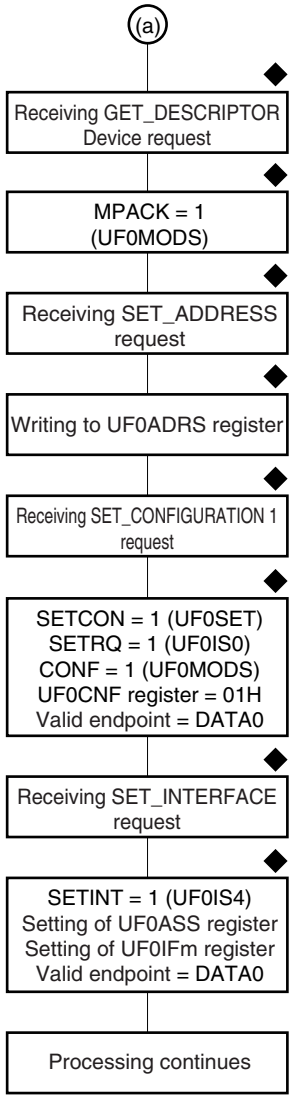
Notes 1. Use one general-purpose port pin for the signal that controls switching of the pull-up resistor of the USB bus.

2. The input mode or control mode of the general-purpose port pin allocated in **Note 1** may be selected as the default value. Note the active level of pull-up processing of D+ on power application.

Remark ♦: Processing by hardware

Figure 12-25. Example of Processing After Power Application/Power Failure (2/3)

(a) Processing after power application (2/2)

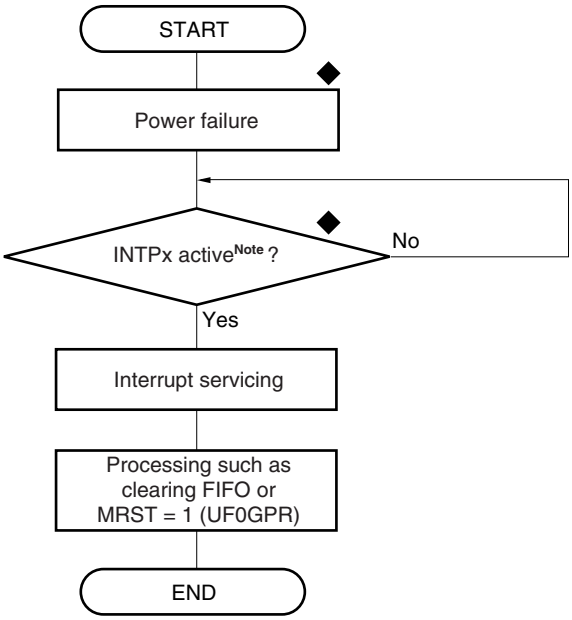


Remarks 1. m = 0 to 4

2. ♦: Processing by hardware

Figure 12-25. Example of Processing After Power Application/Power Failure (3/3)

(b) Processing on power failure



Note INTPx indicates interrupts by the external interrupt pins of the µPD78F0730 (INTP0 to INTP3), and also indicates interrupts input by the external trigger pins (TI000, TI010, TI50, TI51) of the timer.
Allocate one of above external interrupt pin to the following applications.

- Detecting disconnection of the connector in the case of self-powered mode (SFPW bit of UF0DSTL register = 1). In this case, monitor the VDD line of the USB connector, and input the result to the external interrupt pin at the edge. Note that the noise elimination time is necessary (the noise elimination time is MIN. value of the "input high-level width, low-level width (t_{TIH0}, t_{TIL0}, t_{TIH5}, t_{TIL5}, t_{INTH}, t_{INTL})" in **AC Characteristics (1) Basic operation** of **CHAPTER 22 ELECTRICAL SPECIFICATIONS**).
- Detecting turning off power from a HUB chip when the device is mounted on the same board as a HUB.

Remark ♦: Processing by hardware

12.8 External Circuit Configuration

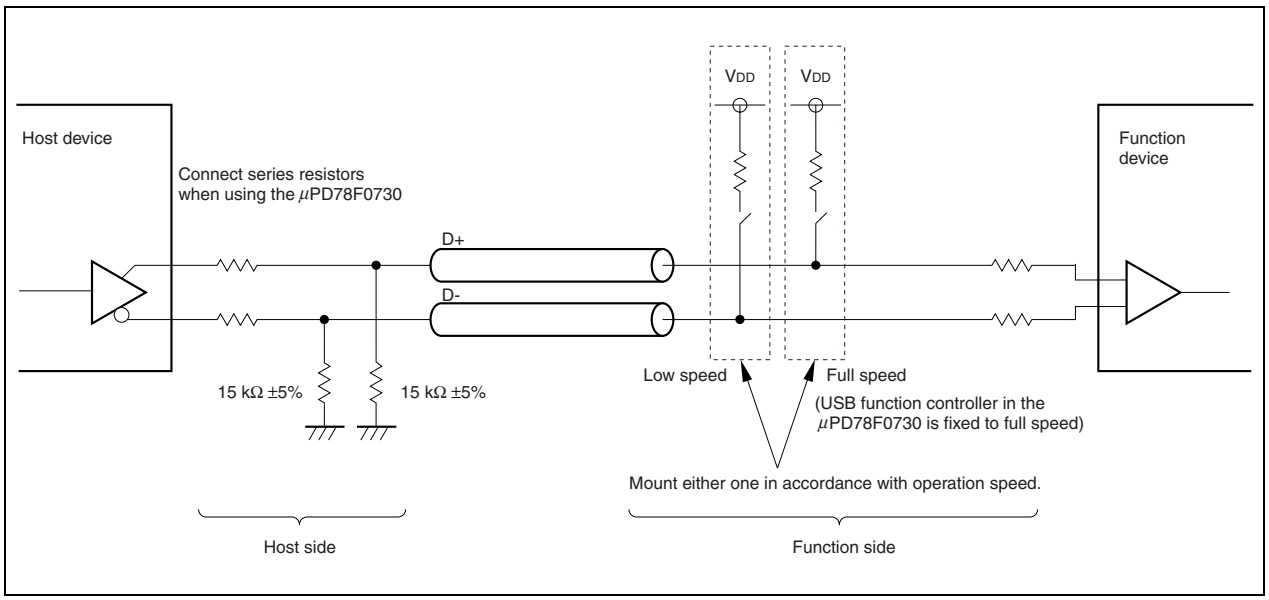
12.8.1 Outline

In USB transmission, when communication is performed with the host controller and function controller facing each other, pull-up/pull-down resistors must be connected to the USB signal (D+/D–) to identify the communication partner. Moreover in the μPD78F0730, series resistors must also be connected.

Because the μPD78F0730 does not include these pull-up/pull-down resistors and series resistors, be sure to connect them externally.

The following shows the outline configuration of the USB transmission line. For details of the external configuration, see the description provided in each section.

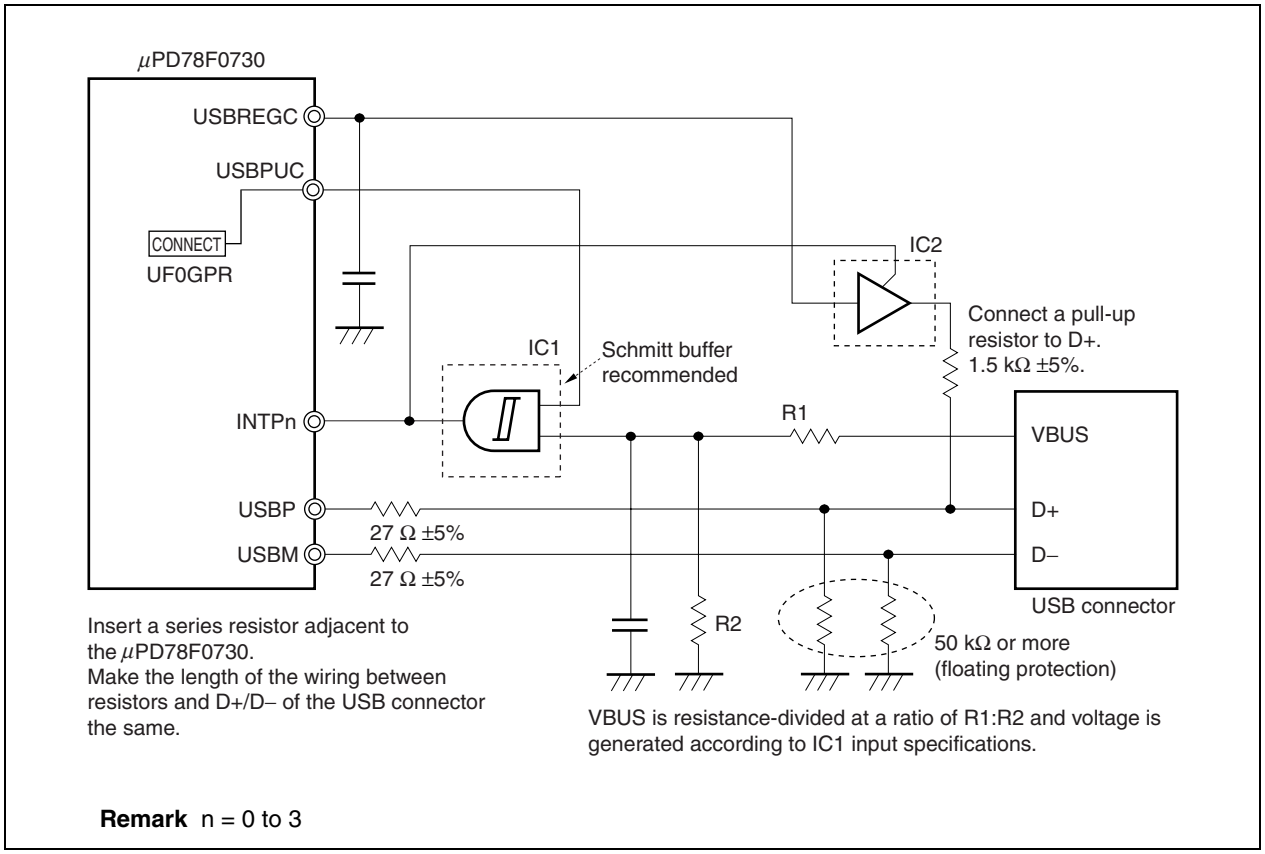
Figure 12-26. Outline Configuration of Pull-up, Pull-down, Series Resistors in USB Transmission Line



12.8.2 USB connection example

Figure 12-27. USB Connection Example

<R>



Remark n = 0 to 3

(1) Series resistor connection to D+/D–

Connect series resistors of 27 Ω ±5% to the D+/D– pins (USBP, USBM) of the USB function controller in the µPD78F0730. If they are not connected, the impedance rating cannot be satisfied and the output waveform may be disturbed.

Allocate the series resistors adjacent to the µPD78F0730, and make the length of the wiring between the series resistors and the USB connectors the same, to make the impedance of D+ and D– equal (a differential with 90 Ω ±5% is recommended).

(2) Pull-up control of D+

Because the USB function controller of the µPD78F0730 is fixed to full speed (FS), be sure to pull up the D+ pin (USBP) by 1.5 kΩ ±5% to USBREGC. To prohibit connection notification (D+ pull-up) to the USB host/HUB (such as while higher priority processing or initialization processing is under execution), the system must control pull-up of D+ via the USBPUC pin.

For a circuit such as the one shown in Figure 12-27, control the pull-up control signal of the D+ pin and the VBUS input signal by using the USBPUC pin and the USB cable VBUS (AND circuit).

In the circuit example in Figure 12-27, pulling up of D+ is prohibited because the USBPUC pin is set to output low level by the initial value after reset.

To be D+ pull-up, be sure to set the CONNECT bit of the UF0GPR register to 1 after reset, and then output the high level from the USBPUC pin.

(3) Detection of USB cable connection/disconnection

The USB function controller (USBF) requires a VBUS input signal to recognize whether the USB cable is connected or disconnected, because the state of the USBF is controlled by hardware. The voltage from the USB host or HUB (5 V) is applied as the VBUS input signal when the USB cable VBUS is connected to the USB host or HUB while the USBF power is off. Therefore, for IC1 in Figure 12-27, use an IC to which voltage can be applied when the system power is off. When disconnecting the USB cable in the circuit in Figure 12-27, the input signal to INTPn may be unstable while the VBUS voltage is dropping. It is therefore recommended to use a Schmitt buffer for IC1 in Figure 12-27.

(4) Floating protection during initialization or when USBF is unused

When the USB function controller is initialized or unused, to avoid a floating status, pull the D+/D− pins down using a resistor of 50 kΩ or higher.

12.9 Cautions for USB Function Controller USBF

(1) Clock accuracy

To operate the USB function controller, the internal clock (16 MHz external clock / 4 × internal clock multiplied by 12 = 48 MHz internal clock or 12 MHz external clock / 2 × internal clock multiplied by 8 = 48 MHz internal clock) must be used as the USB clock. When the USB clock is used, use a resonator with an accuracy of 16 MHz (or 12 MHz) ±500 ppm (max.). If the USB clock accuracy drops, the transmission data cannot satisfy the USB rating.

CHAPTER 13 INTERRUPT FUNCTIONS

13.1 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, PR1H).

Multiple interrupt servicing can be applied to low-priority interrupts when high-priority interrupts are generated. If two or more interrupt requests, each having the same priority, are simultaneously generated, then they are processed according to the priority of vectored interrupt servicing. For the priority order, see **Table 13-1**.

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: 4, internal: 14

(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.

13.2 Interrupt Sources and Configuration

The µPD78F0730 has a total of 19 interrupt sources including maskable interrupts and software interrupts. In addition, they also have up to four reset sources (see **Table 13-1**).

Table 13-1. Interrupt Source List

Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Maskable	0	INTLVI	Low-voltage detection[Note 3]	Internal	0004H	(A)
	1	INTP0	Pin input edge detection	External	0006H	(B)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTUSB0	USB function status 0	Internal	000EH	(A)
	6	INTUSB1	USB function status 1		0010H	
	7	INTSRE6	UART6 reception error generation		0012H	
	8	INTSR6	End of UART6 reception		0014H	
	9	INTST6	End of UART6 transmission		0016H	
	10	INTCSI10	End of CSI10 communication		0018H	
	11	INTTMH1	Match between TMH1 and CMP01 (when compare register is specified)		001AH	
	12	INTUSB2	USB function status 2		001CH	
	13	INTTM50	Match between TM50 and CR50 (when compare register is specified)		001EH	
	14	INTTM000	Match between TM00 and CR000 (when compare register is specified), TI010 pin valid edge detection (when capture register is specified)		0020H	
	15	INTTM010	Match between TM00 and CR010 (when compare register is specified), TI000 pin valid edge detection (when capture register is specified)		0022H	
	16	INTRSUM	USB Resume signal detection		0024H	
	–	–	–	–	0026H	–
	–	–	–	–	0028H	–
	18	INTTM51	Match between TM51 and CR51 (when compare register is specified)	Internal	002AH	(A)
	–	–	–	–	002CH to 003CH	–
Software	–	BRK	BRK instruction execution	–	003EH	(C)
Reset	–	RESET	Reset input	–	0000H	–
		POC	Power-on clear			
		LVI	Low-voltage detection[Note 4]			
		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.

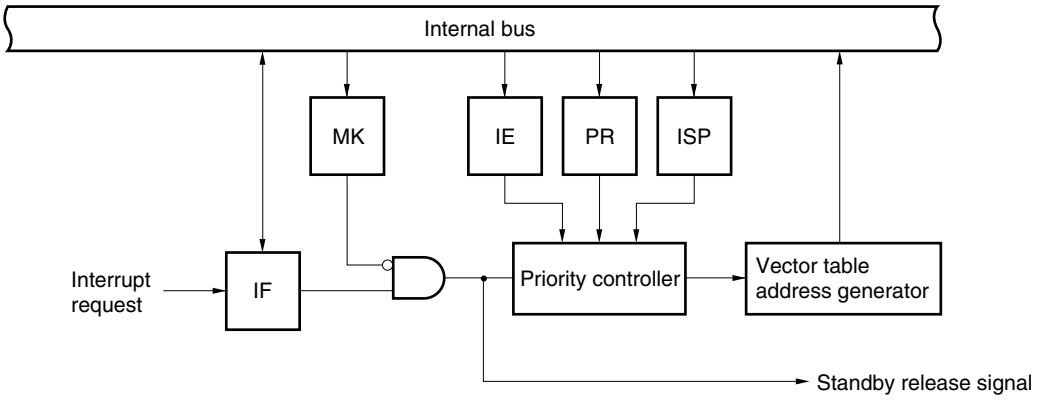
2. Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 13-1.

3. When bit 1 (LVIMD) of the low-voltage detection register (LVIM) is cleared to 0.

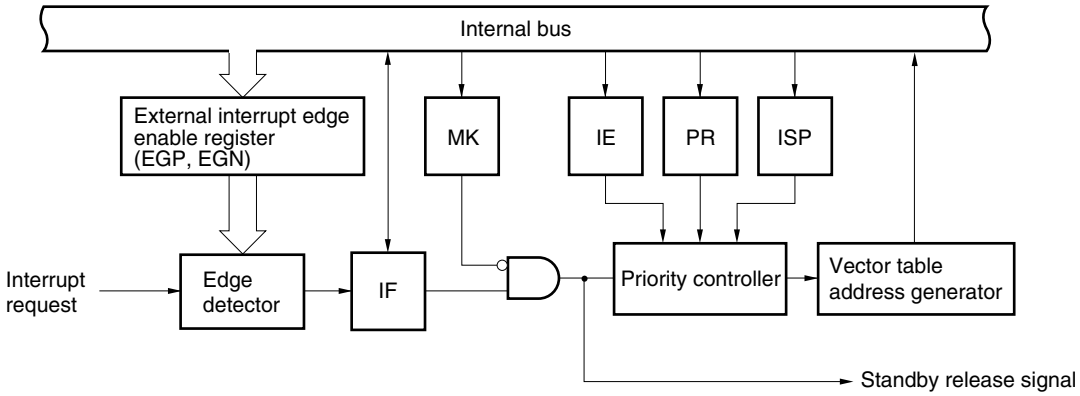
4. When bit 1 (LVIMD) of the low-voltage detection register (LVIM) is set to 1.

Figure 13-1. Basic Configuration of Interrupt Function

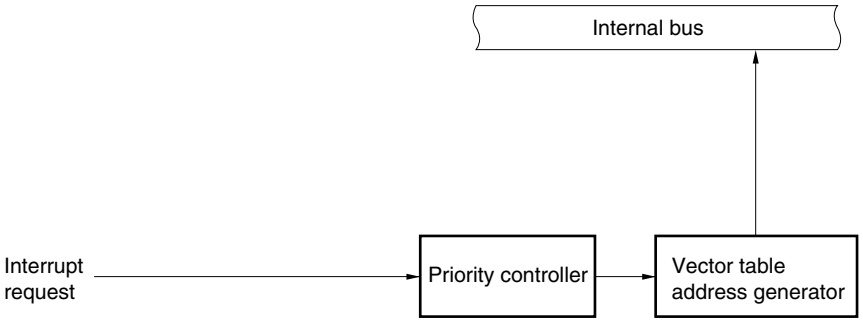
(A) Internal maskable interrupt



(B) External maskable interrupt (INTP0 to INTP3)



(C) Software interrupt



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

13.3 Registers Controlling Interrupt Functions

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

- Interrupt request flag register (IF0L, IF0H, IF1L, IF1H)
- Interrupt mask flag register (MK0L, MK0H, MK1L, MK1H)
- Priority specification flag register (PR0L, PR0H, PR1L, PR1H)
- External interrupt rising edge enable register (EGP)
- External interrupt falling edge enable register (EGN)
- Program status word (PSW)

Table 13-2 shows a list of interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

Table 13-2. Flags Corresponding to Interrupt Request Sources

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specification Flag	
		Register		Register		Register
INTLVI	LVIIF	IF0L	LVIMK	MK0L	LVIPR	PR0L
INTP0	PIF0		PMK0		PPR0	
INTP1	PIF1		PMK1		PPR1	
INTP2	PIF2		PMK2		PPR2	
INTP3	PIF3		PMK3		PPR3	
INTUSB0	USBIF0		USBMK0		USBPR0	
INTUSB1	USBIF1		USBMK1		USBPR1	
INTSRE6	SREIF6		SREMK6		SREPR6	
INTSR6	SRIF6	IF0H	SRMK6	MK0H	SRPR6	PR0H
INTST6	STIF6		STMK6		STPR6	
INTCSI10	CSIIF10		CSIMK10		CSIPR10	
INTTMH1	TMIFH1		TMMKH1		TMPRH1	
INTTM50	TMIF50		TMMK50		TMPR50	
INTTM000	TMIF000		TMMK000		TMPR000	
INTTM010	TMIF010		TMMK010		TMPR010	
INTRSUM	RSUMIF	IF1L	RSUMMK	MK1L	RSUMPR	PR1L
INTTM51	TMIF51		TMMK51		TMPR51	

(1) Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H)

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

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

IF0L, IF0H, IF1L, and IF1H are set by a 1-bit or 8-bit memory manipulation instruction. When IF0L and IF0H, and IF1L and IF1H are combined to form 16-bit registers IF0 and IF1, they are set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to 00H.

Figure 13-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H)

Address: FFE0H After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF0L	SREIF6	USBIF1	USBIF2	PIF3	PIF2	PIF1	PIF0	LVIIF

Address: FFE1H After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF0H	TMIF010	TMIF000	TMIF50	USBIF2	TMIFH1	CSIIF10	STIF6	SRIF6

Address: FFE2H After reset: 00H R/W

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

Address: FFE3H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
IF1H	0	0	0	0	0	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 bits 1, 2, 4 to 7 of IF1L and bits 0 to 7 of IF1H to 0.

2. When operating a timer or serial interface after standby release, operate it once after clearing the interrupt request flag. An interrupt request flag may be set by noise.

Cautions 3. Use the 1-bit memory manipulation instruction (CLR1) for manipulating the flag of the interrupt request flag register. A 1-bit manipulation instruction such as "IF0L.0 = 0;" and "_asm("clr1 IF0L, 0");" should be used when describing in C language, because assembly instructions after compilation must be 1-bit memory manipulation instructions (CLR1).
If an 8-bit memory manipulation instruction "IF0L & = 0xfe;" is described in C language, for example, it is converted to the following three assembly instructions after compilation:

```
mov a, IF0L
and a, #0FEH
mov IF0L, a
```

In this case, at the timing between "mov a, IF0L" and "mov IF0L, a", if the request flag of another bit of the identical interrupt request flag register (IF0L) is set to 1, it is cleared to 0 by "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, MK1H)

The interrupt mask flags are used to enable/disable the corresponding maskable interrupt servicing.

MK0L, MK0H, MK1L, and MK1H are set by a 1-bit or 8-bit memory manipulation instruction. When MK0L and MK0H, and MK1L and MK1H are combined to form 16-bit registers MK0 and MK1, they are set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

Figure 13-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H)

Address: FFE4H After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK0L	SREMK6	USBMK1	USBMK0	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	USBMK2	TMMKH1	CSIMK0	STMK6	SRMK6

Address: FFE6H After reset: FFH R/W

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

Address: FFE7H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
MK1H	1	1	1	1	1	1	1	1

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

Caution Be sure to set bits 1, 2, 4 to 7 of MK1L and bits 0 to 7 of MK1H to 1.

(3) Priority specification flag registers (PR0L, PR0H, PR1L, PR1H)

The priority specification flag registers are used to set the corresponding maskable interrupt priority order.

PR0L, PR0H, PR1L, and PR1H are set by a 1-bit or 8-bit memory manipulation instruction. If PR0L and PR0H, and PR1L and PR1H are combined to form 16-bit registers PR0 and PR1, they are set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

Figure 13-4. Format of Priority Specification Flag Registers (PR0L, PR0H, PR1L, PR1H)

Address: FFE8H After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR0L	SREPR6	USBPR1	USBPR0	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	USBPR2	TMPRH1	CSIPR10	STPR6	SRPR6

Address: FFEAH After reset: FFH R/W

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

Address: FFEBH After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PR1H	1	1	1	1	1	1	1	1

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

Caution Be sure to set bits 1, 2, 4 to 7 of PR1L and bits 0 to 7 of PR1H to 1.

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

These registers specify the valid edge for INTP0 to INTP3.

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

Reset signal generation sets these registers to 00H.

Figure 13-5. 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	0	0	EGP3	EGP2	EGP1	EGP0

Address: FF49H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGN	0	0	0	0	EGN3	EGN2	EGN1	EGN0

EGPn	EGNn	INTPn pin valid edge selection (n = 0 to 3)
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 4 to 7 to 0.

Table 13-3 shows the ports corresponding to EGPn and EGNn.

Table 13-3. Ports Corresponding to EGPn and EGNn

Detection Enable Register		Edge Detection Port	Interrupt Request Signal
EGP0	EGN0	P120	INTP0
EGP1	EGN1	P30	INTP1
EGP2	EGN2	P31	INTP2
EGP3	EGN3	P32	INTP3

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 3

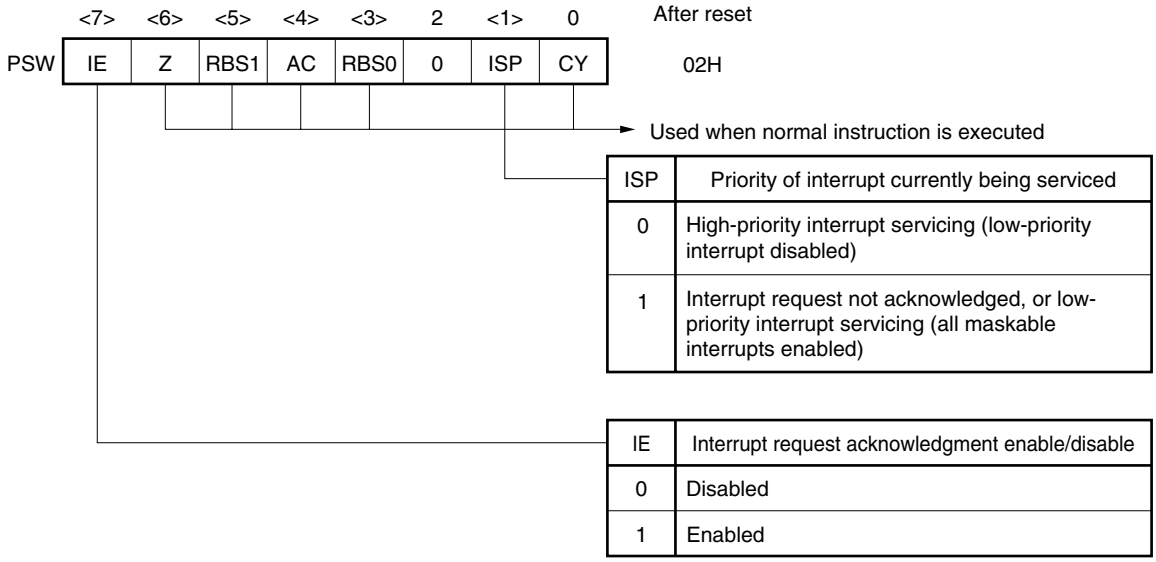
(5) Program status word (PSW)

The program status word is a register used to hold the instruction execution result and the current status for an interrupt request. The IE flag that sets maskable interrupt enable/disable and the 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 13-6. Format of Program Status Word



ISP	Priority of interrupt currently being serviced
0	High-priority interrupt servicing (low-priority interrupt disabled)
1	Interrupt request not acknowledged, or low-priority interrupt servicing (all maskable interrupts enabled)

IE	Interrupt request acknowledgment enable/disable
0	Disabled
1	Enabled

13.4 Interrupt Servicing Operations

13.4.1 Maskable interrupt acknowledgement

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 13-4 below.

For the interrupt request acknowledgement timing, see **Figures 13-8** and **13-9**.

Table 13-4. Time from Generation of Maskable Interrupt Until Servicing

	Minimum Time	Maximum Time[Note]
When ××PR = 0	7 clocks	32 clocks
When ××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.

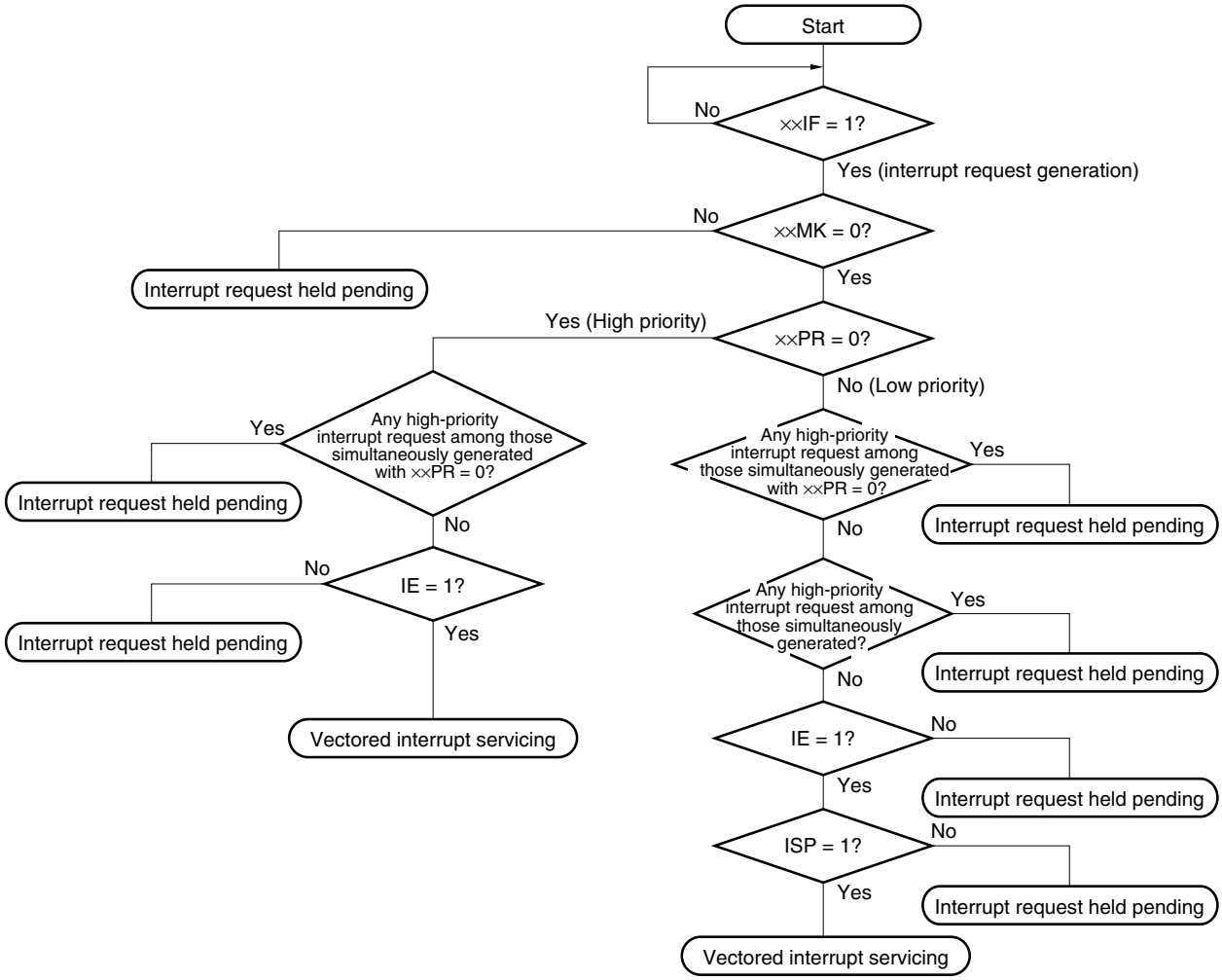
An interrupt request that is held pending is acknowledged when it becomes acknowledgeable.

Figure 13-7 shows the interrupt request acknowledgement 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.

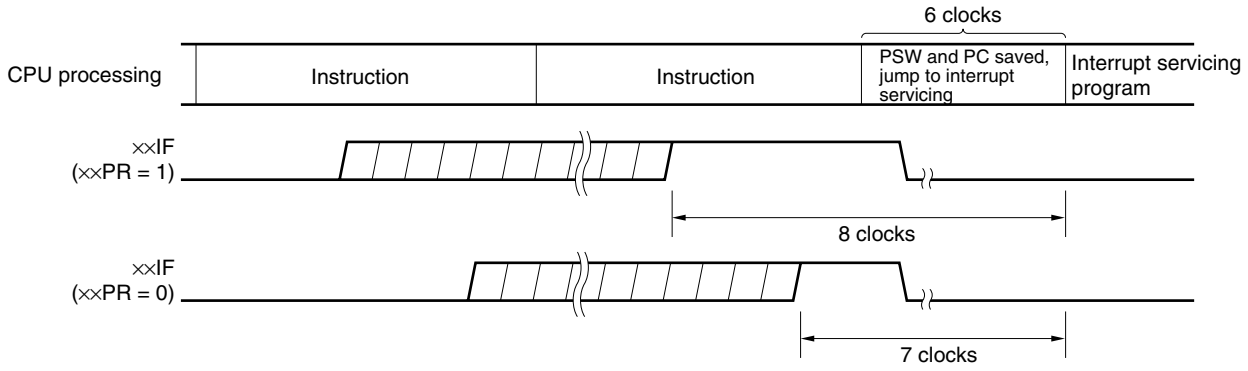
Restoring from an interrupt is possible by using the RETI instruction.

Figure 13-7. Interrupt Request Acknowledgement Processing Algorithm



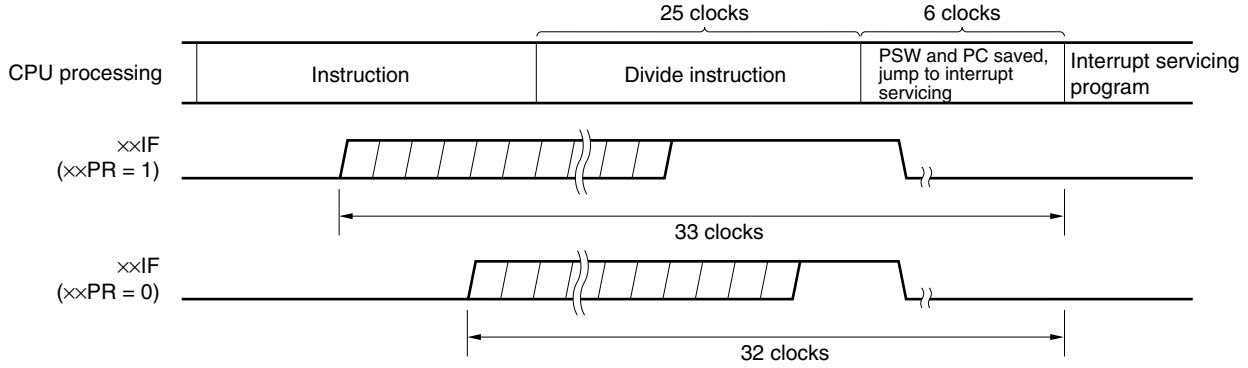
××IF: Interrupt request flag
××MK: Interrupt mask flag
××PR: Priority specification flag
IE: Flag that controls acknowledgement of maskable interrupt request (1 = Enable, 0 = Disable)
ISP: Flag that indicates the priority level of the interrupt currently being serviced (0 = high-priority interrupt servicing, 1 = No interrupt request acknowledged, or low-priority interrupt servicing)

Figure 13-8. Interrupt Request Acknowledgement Timing (Minimum Time)



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

Figure 13-9. Interrupt Request Acknowledgement Timing (Maximum Time)



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

13.4.2 Software interrupt request acknowledgement

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

If a software interrupt request is acknowledged, the contents are saved into the stacks in the order of the program status word (PSW), then program counter (PC), the IE flag is reset (0), and the contents of the vector table (003EH, 003FH) are loaded into the PC and branched.

Restoring from a software interrupt is possible by using the RETB instruction.

Caution Do not use the RETI instruction for restoring from the software interrupt.

13.4.3 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 acknowledgement enabled state is selected (IE = 1). When an interrupt request is acknowledged, interrupt request acknowledgement 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 acknowledgement.

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 13-5 shows relationship between interrupt requests enabled for multiple interrupt servicing and Figure 13-10 shows multiple interrupt servicing examples.

Table 13-5. 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 acknowledgement is disabled.

IE = 1: Interrupt request acknowledgement is enabled.

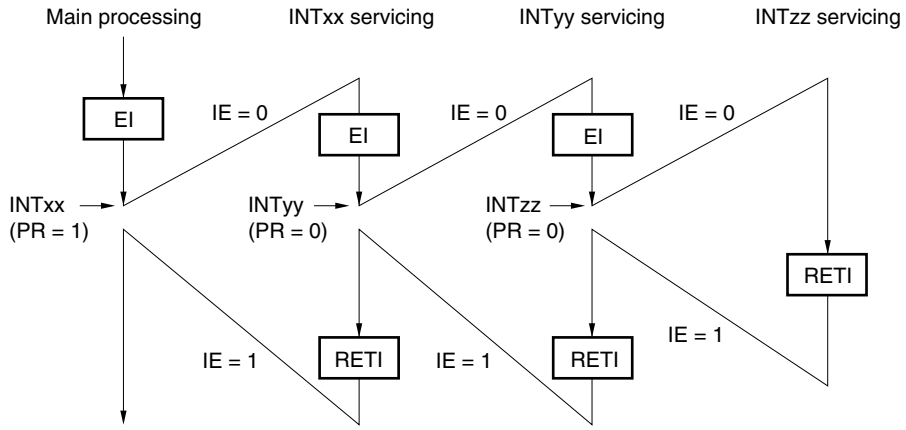
4. PR is a flag contained in PR0L, PR0H, PR1L, and PR1H.

PR = 0: Higher priority level

PR = 1: Lower priority level

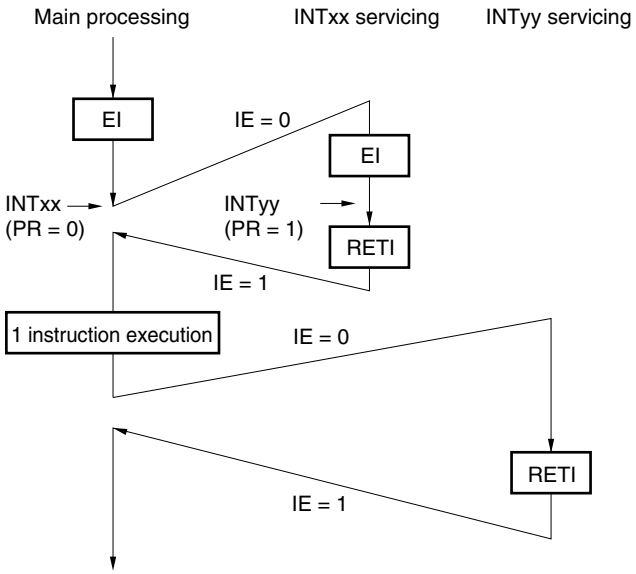
Figure 13-10. Examples of Multiple Interrupt Servicing (1/2)

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

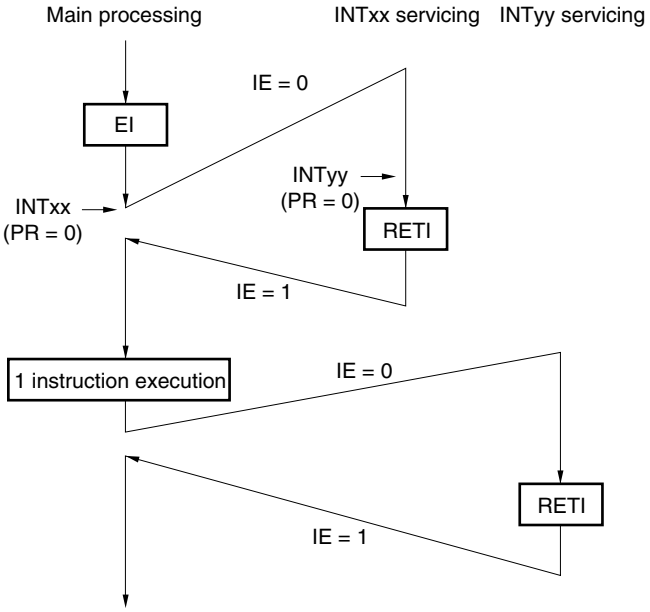


Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 0: Higher priority level
PR = 1: Lower priority level
IE = 0: Interrupt request acknowledgment disabled

Figure 13-10. 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 acknowledgement disabled

13.4.4 Interrupt request hold

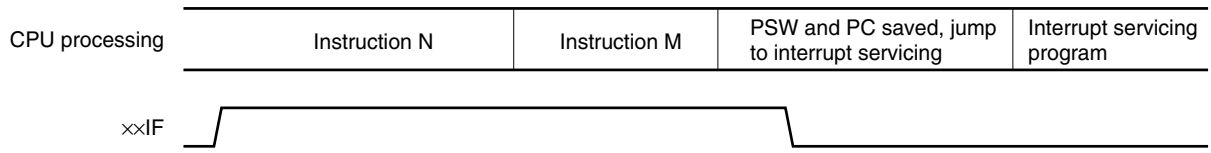
There are instructions where, even if an interrupt request is issued for them while another instruction is being executed, request acknowledgement is held pending until the end of execution of the next instruction. These instructions (interrupt request hold instructions) are listed below.

- MOV PSW, #byte
- MOV A, PSW
- MOV PSW, A
- MOV1 PSW. bit, CY
- MOV1 CY, PSW. bit
- AND1 CY, PSW. bit
- OR1 CY, PSW. bit
- XOR1 CY, PSW. bit
- SET1 PSW. bit
- CLR1 PSW. bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT PSW. bit, $addr16
- BF PSW. bit, $addr16
- BTCLR PSW. bit, $addr16
- EI
- DI
- Manipulation instructions for the IF0L, IF0H, IF1L, IF1H, MK0L, MK0H, MK1L, MK1H, PR0L, PR0H, PR1L, and PR1H registers.

Caution The BRK instruction is not one of the above-listed interrupt request hold instructions. However, the software interrupt activated by executing the BRK instruction causes the IE flag to be cleared. Therefore, even if a maskable interrupt request is generated during execution of the BRK instruction, the interrupt request is not acknowledged.

Figure 13-11 shows the timing at which interrupt requests are held pending.

Figure 13-11. Interrupt Request Hold



Remarks 1. Instruction N: Interrupt request hold instruction

2. Instruction M: Instruction other than interrupt request hold instruction

3. The ××PR (priority level) values do not affect the operation of ××IF (interrupt request).

CHAPTER 14 STANDBY FUNCTION

14.1 Standby Function and Configuration

14.1.1 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.

Caution When shifting to the STOP mode, be sure to stop the peripheral hardware operation operating with main system clock before executing STOP instruction.

14.1.2 Registers controlling standby function

The standby function is controlled by the following two registers.

- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)

Remark For the registers that start, stop, or select the clock, see **CHAPTER 5 CLOCK GENERATOR**.

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

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

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

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

Figure 14-1. 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		
						fx = 12 MHz	fx = 16 MHz
1	0	0	0	0	2^{11}/fx min.	170.7 µs min.	128 µs min.
1	1	0	0	0	2^{13}/fx min.	682.7 µs min.	512 µs min.
1	1	1	0	0	2^{14}/fx min.	1.37 ms min.	1.024 ms min.
1	1	1	1	0	2^{15}/fx min.	2.73 ms min.	2.048 ms min.
1	1	1	1	1	2^{16}/fx min.	5.46 ms min.	4.096 ms min.

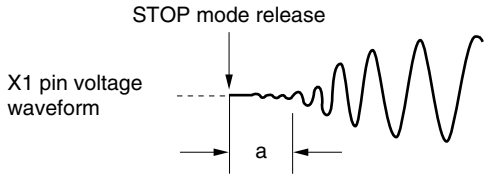
Cautions 1. After the above time has elapsed, the bits are set to 1 in order from MOST11 and remain 1.

2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTS. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.

- **Desired OSTC oscillation stabilization time ≤ Oscillation stabilization time set by OSTS**

Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC after STOP mode is released.

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



Remark fx: X1 clock oscillation frequency

(2) Oscillation stabilization time select register (OSTS)

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

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

When the internal high-speed oscillation clock is selected as the CPU clock, confirm with OSTC that the desired oscillation stabilization time has elapsed after the STOP mode is released. The oscillation stabilization time can be checked up to the time set using OSTC.

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

Reset signal generation sets OSTS to 05H.

Figure 14-2. 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		
				fx = 12 MHz	fx = 16 MHz
0	0	1	2^{11}/fx	170.7 µs	128 µs
0	1	0	2^{13}/fx	682.7 µs	512 µs
0	1	1	2^{14}/fx	1.37 ms	1.024 ms
1	0	0	2^{15}/fx	2.73 ms	2.048 ms
1	0	1	2^{16}/fx	5.46 ms	4.096 ms
Other than above			Setting prohibited		

Cautions 1. To set the STOP mode when the X1 clock is used as the CPU clock, set OSTS before executing the STOP instruction.

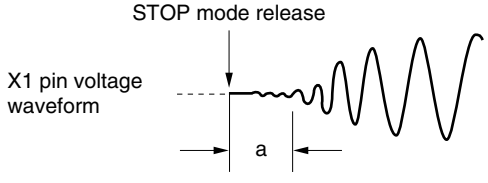
2. Do not change the value of the OSTS register during the X1 clock oscillation stabilization time.

3. The oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTS. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.

- **Desired OSTC oscillation stabilization time ≤ Oscillation stabilization time set by OSTS**

Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC after STOP mode is released.

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



Remark fx: X1 clock oscillation frequency

14.2 Standby Function Operation

14.2.1 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 14-1. Operating Statuses in HALT Mode

HALT Mode Setting / Item			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})
System clock			Clock supply to the CPU is stopped		
	Main system clock	f_{RH}	Operation continues (cannot be stopped)	Status before HALT mode was set is retained	
		f_X	Status before HALT mode was set is retained	Operation continues (cannot be stopped)	Status before HALT mode was set is retained
		f_{EXCLK}	Operates or stops by external clock input		Operation continues (cannot be stopped)
	f_{RL}		Status before HALT mode was set is retained		
PLL			Operable		
CPU			Operation stopped		
Flash memory			Operation stopped		
RAM			Status before HALT mode was set is retained		
Regulator		For chip	Operable in normal operation mode.		
		For USB			
Port (latch)			Status before HALT mode was set is retained		
16-bit timer/event counter 00			Operable		
8-bit timer/event counter		50			
		51			
8-bit timer 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		
		CSI10			
		USB			
Power-on-clear function					
Low-voltage detection function			Detectable in setting before HALT mode transition		
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

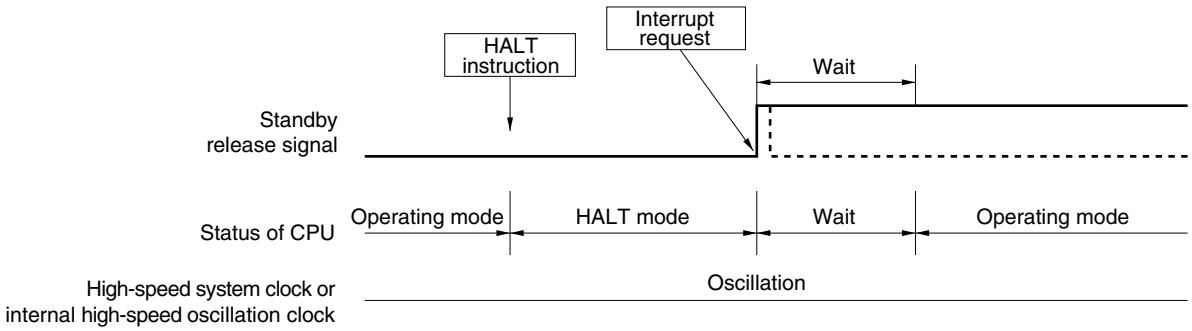
(2) HALT mode release

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 acknowledgement is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgement is disabled, the next address instruction is executed.

Figure 14-3. HALT Mode Release by Interrupt Request Generation



Remarks 1. The broken lines indicate the case when the interrupt request which has released the standby mode is acknowledged.

2. The wait time is as follows:

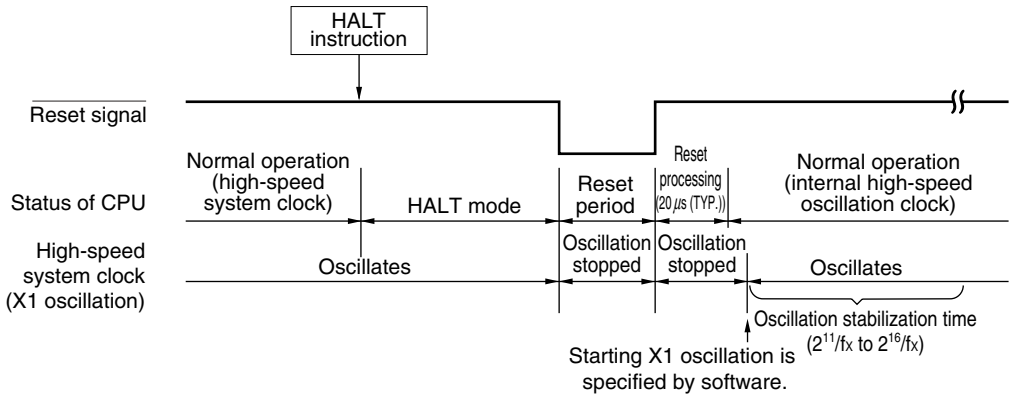
- When vectored interrupt servicing is carried out: 8 or 9 clocks
- When vectored interrupt servicing is not carried out: 2 or 3 clocks

(b) Release by reset signal generation

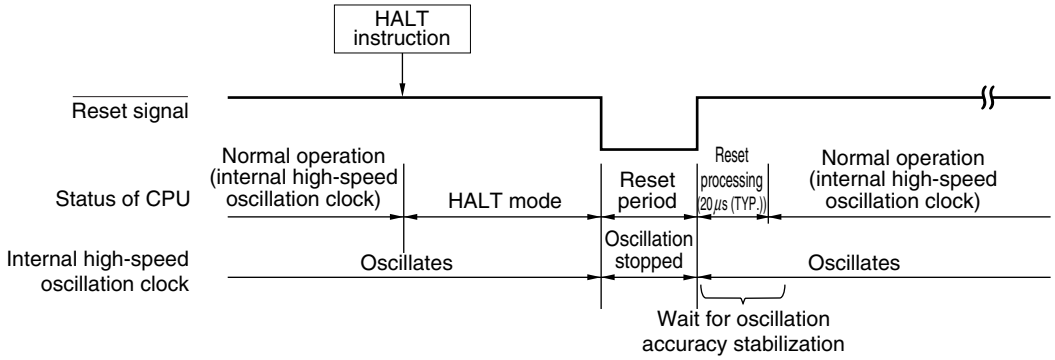
When the reset signal is generated, HALT mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

Figure 14-4. HALT Mode Release by Reset

(1) When high-speed system clock is used as CPU clock



(2) When internal high-speed oscillation clock is used as CPU clock



Remark fx: X1 clock oscillation frequency

Table 14-2. Operation in Response to Interrupt Request in HALT Mode

Release Source	MK××	PR××	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

14.2.2 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 14-3. Operating Statuses in STOP Mode

STOP Mode Setting / Item			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})
System clock			Clock supply to the CPU is stopped		
	Main system clock	f_{RH}	Stopped		
		f_X			
		f_{EXCLK}	Input invalid		
	f_{RL}		Status before STOP mode was set is retained		
PLL			Operation stopped		
CPU			Operation stopped		
Flash memory			Operation stopped		
RAM			Status before STOP mode was set is retained		
Regulator	For chip		Operable in low operating current mode		
	For USB				
Port (latch)			Status before STOP mode was set is retained		
16-bit timer/event counter 00			Operation stopped		
8-bit timer/event counter	50		Operable only when TI50 is selected as the count clock		
	51		Operable only when TI51 is selected as the count clock		
8-bit timer H1			Operable only when f_{RL}, $f_{RL}/2^7$, or $f_{RL}/2^9$ 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		
	CSI10		Operable only when external clock is selected as the serial clock		
	USB		Operation stopped		
Power-on-clear function			Operable		
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

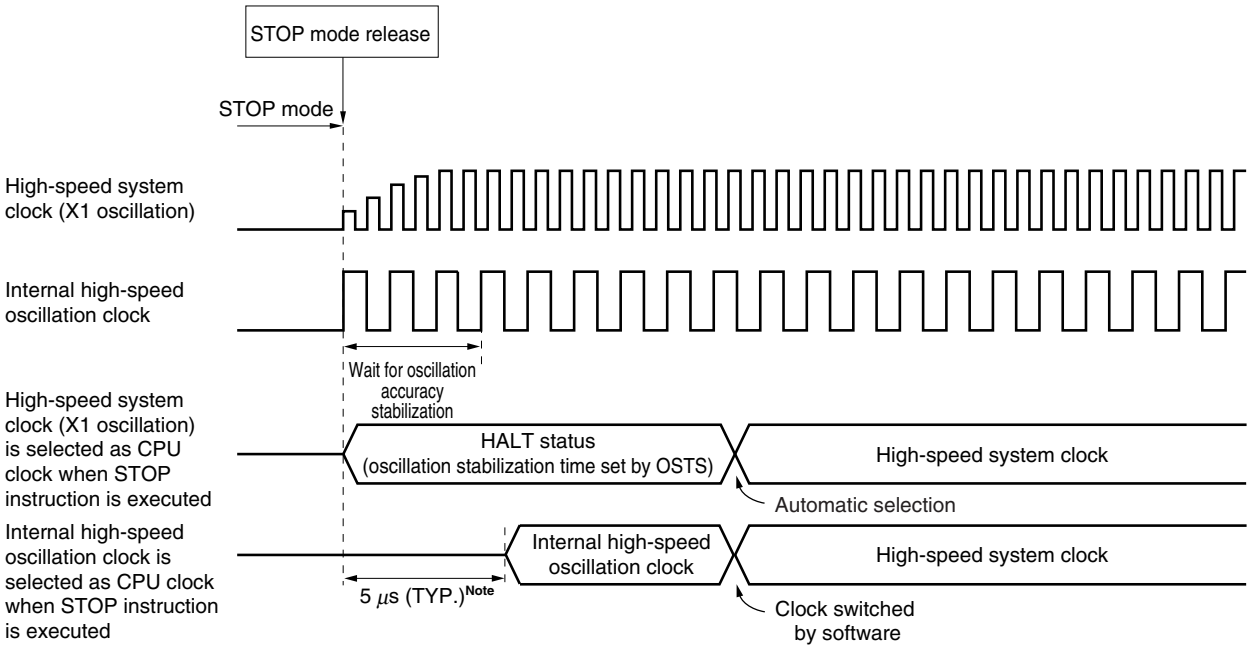
Cautions 1. To use the peripheral hardware that stops operation in the STOP mode, and the peripheral hardware for which the clock that stops oscillating in the STOP mode after the STOP mode is released, restart the peripheral hardware.

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. If the STOP instruction is executed with AMPH set to 1 when the internal high-speed oscillation clock or external main system clock is used as the CPU clock, the internal high-speed oscillation clock or external main system clock is supplied to the CPU 5 μs (MIN.) after the STOP mode has been released.

(2) STOP mode release

Figure 14-5. Operation Timing When STOP Mode Is Released



Note When AMPH = 1

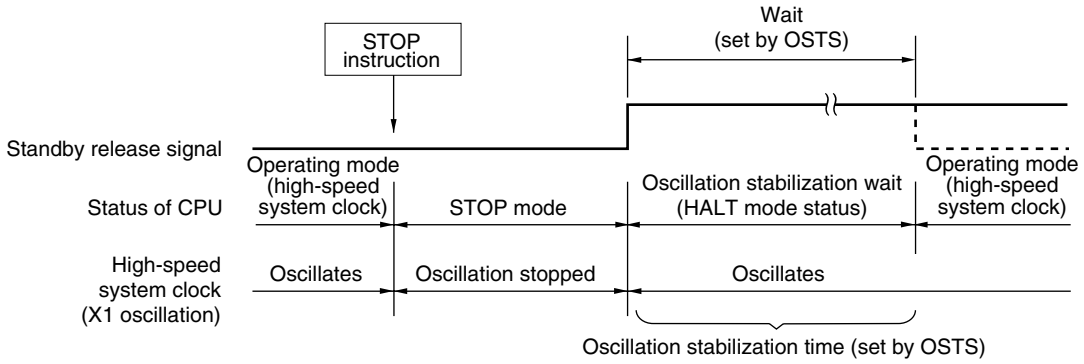
The STOP mode can be released by the following two sources.

(a) Release by unmasked interrupt request

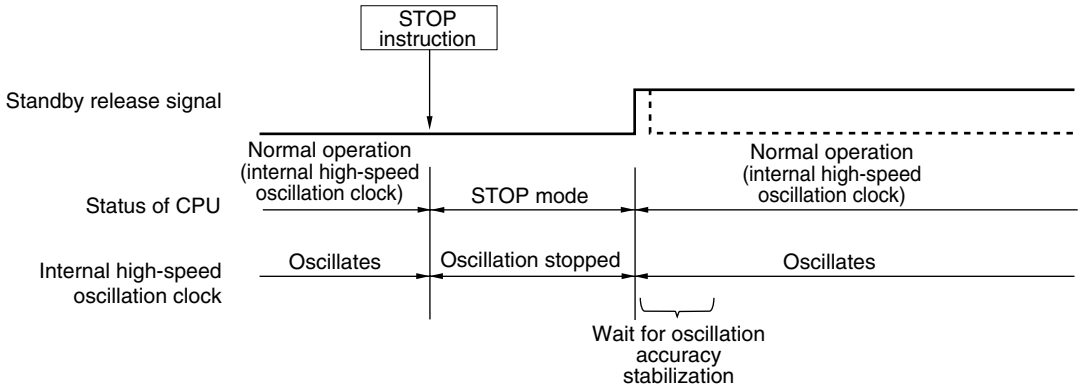
When an unmasked interrupt request is generated, the STOP mode is released. After the oscillation stabilization time has elapsed, if interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

Figure 14-6. STOP Mode Release by Interrupt Request Generation

(1) When high-speed system clock is used as CPU clock



(2) When internal high-speed oscillation clock is used as CPU clock



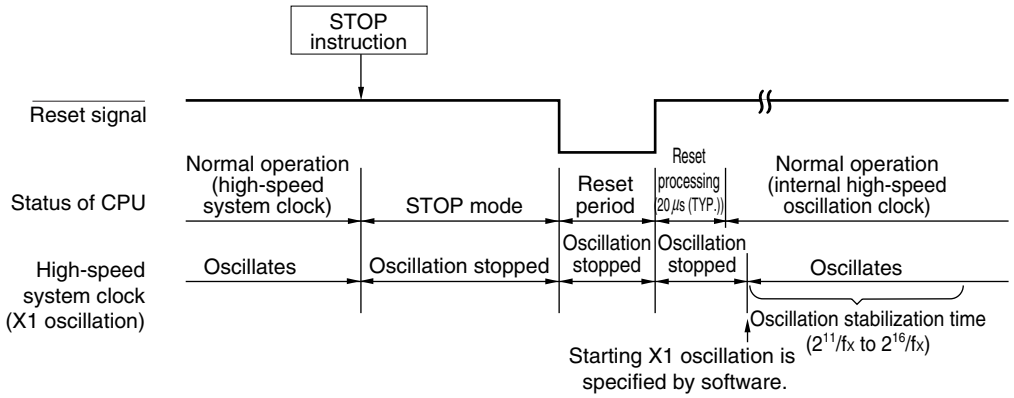
Remark The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

(b) Release by reset signal generation

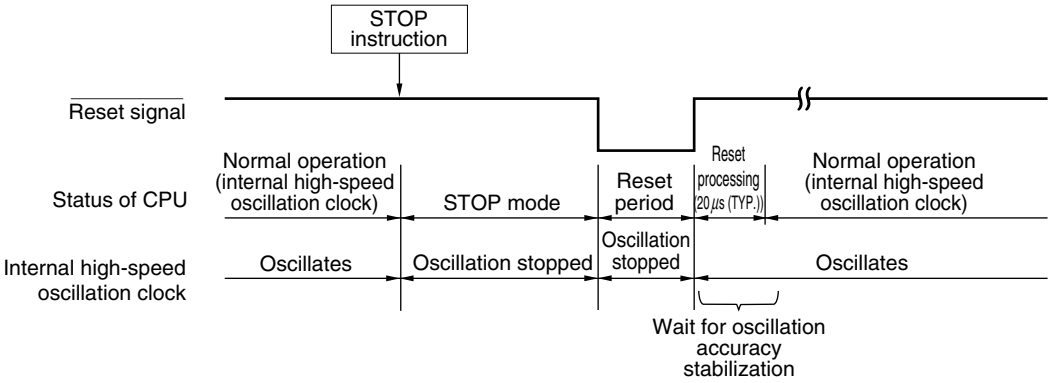
When the reset signal is generated, STOP mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

Figure 14-7. 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 14-4. Operation in Response to Interrupt Request in STOP Mode

Release Source	MK××	PR××	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 15 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)

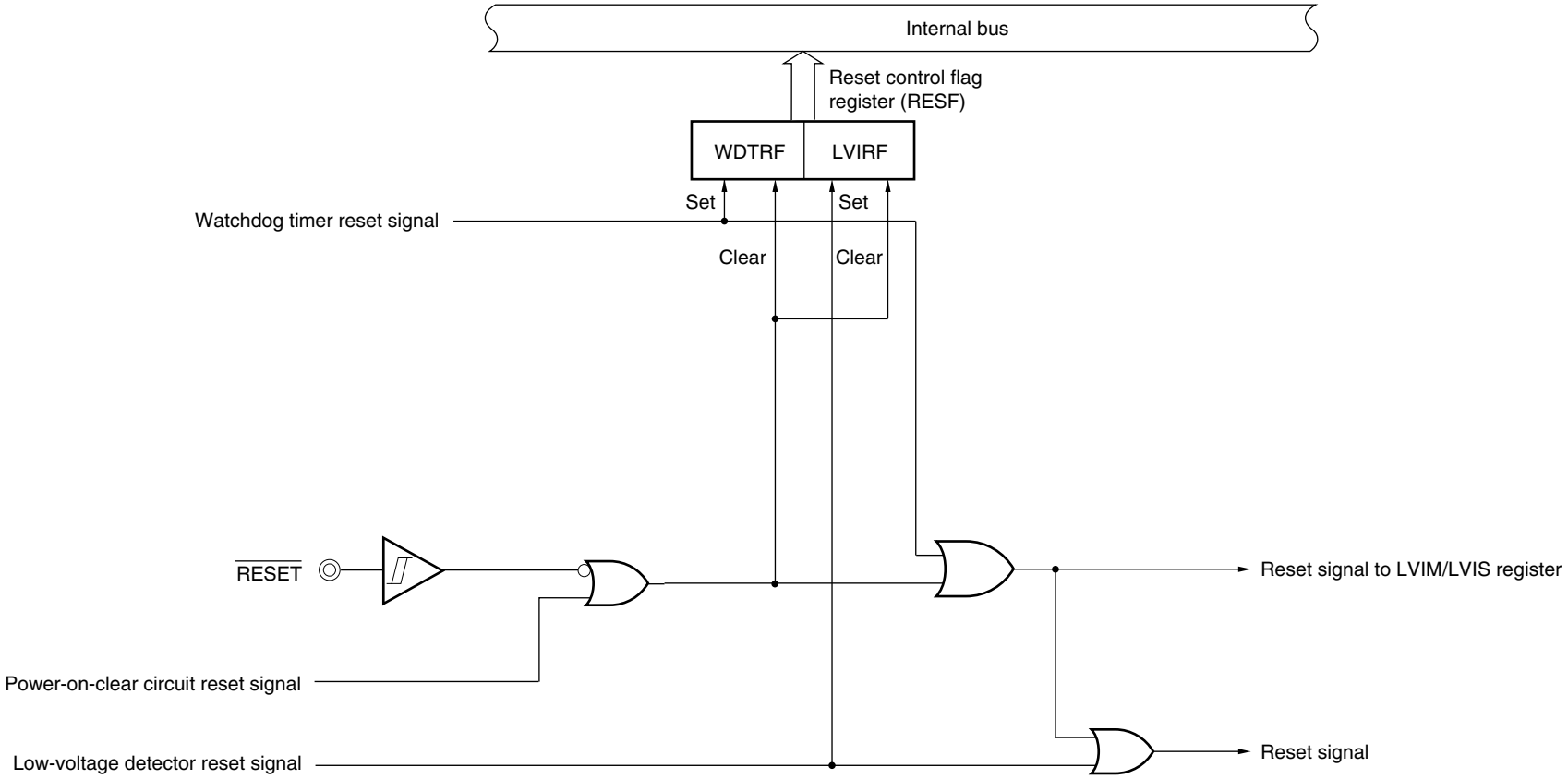
External and internal resets have no functional differences. In both cases, program execution starts at the address at 0000H and 0001H when the reset signal is generated.

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 15-1 and 15-2. 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 15-2** to **15-4**) 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 16 POWER-ON-CLEAR CIRCUIT** and **CHAPTER 17 LOW-VOLTAGE DETECTOR**) after reset processing.

Cautions 1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.
2. During reset signal generation, the X1 clock, internal high-speed oscillation clock, and internal low-speed oscillation clock stop oscillating. External main system clock input becomes invalid.
3. When the STOP mode is released by a reset, the STOP mode contents are held during reset input. However, the port pins become high-impedance.

Figure 15-1. 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 15-2. Timing of Reset by $\overline{\text{RESET}}$ Input

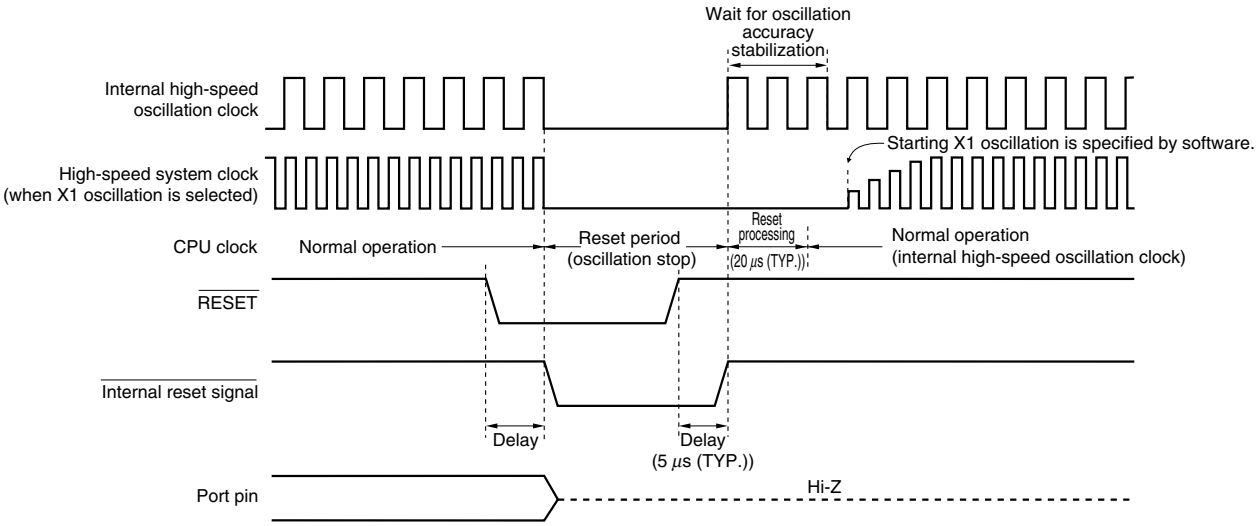
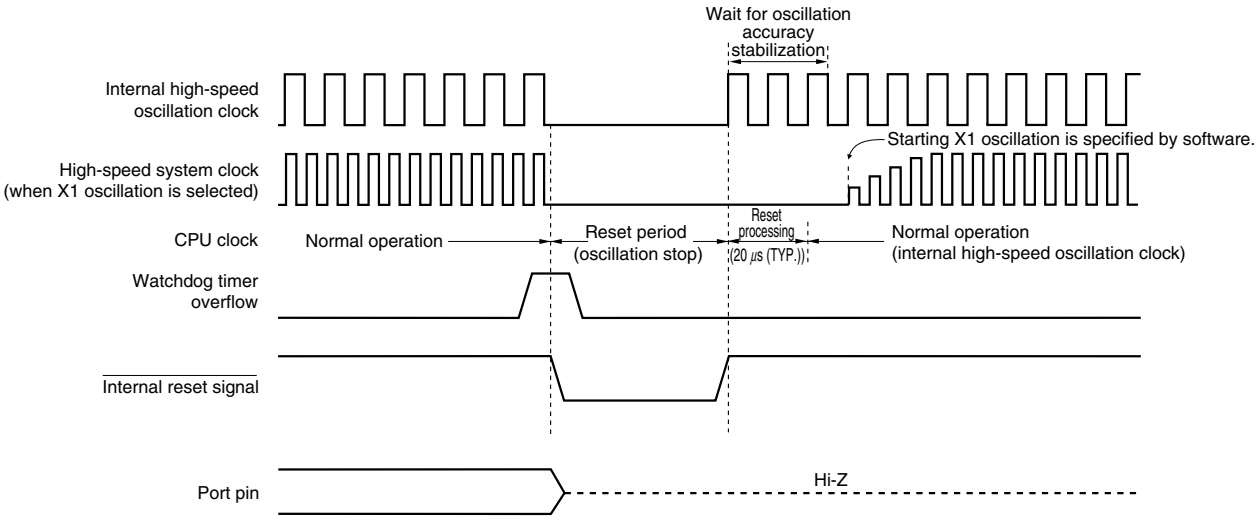
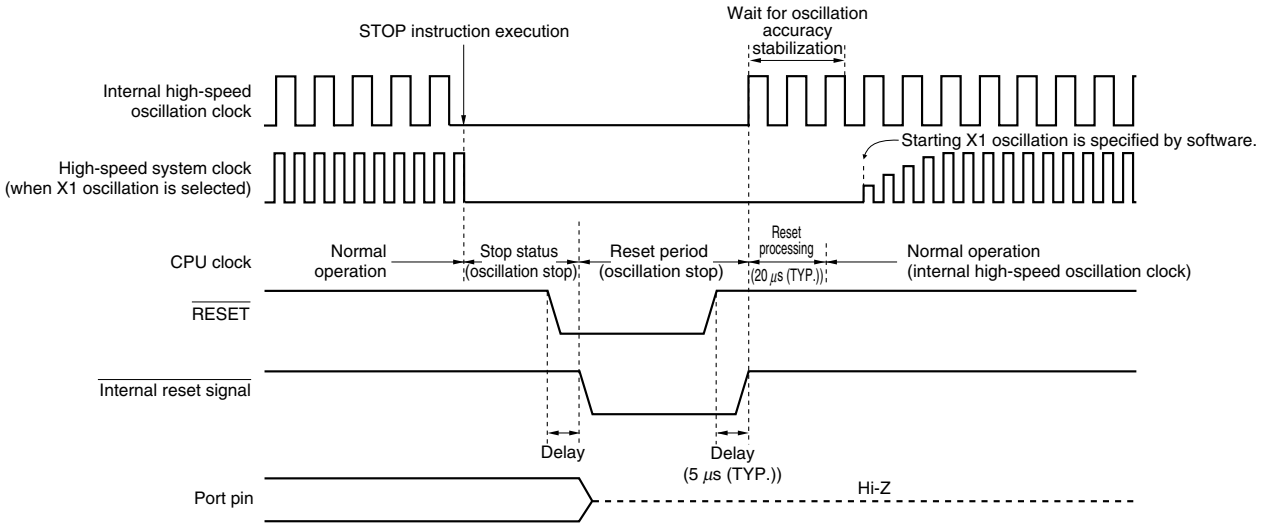


Figure 15-3. Timing of Reset Due to Watchdog Timer Overflow



Caution A watchdog timer internal reset resets the watchdog timer.

Figure 15-4. Timing of Reset in STOP Mode by $\overline{\text{RESET}}$ Input



Remark For the reset timing of the power-on-clear circuit and low-voltage detector, see **CHAPTER 16 POWER-ON-CLEAR CIRCUIT** and **CHAPTER 17 LOW-VOLTAGE DETECTOR**.

Table 15-1. Operation Statuses During Reset Period

Item			During Reset Period
System clock			Clock supply to the CPU is stopped.
	Main system clock	f_{RH}	Operation stopped
		f_X	Operation stopped (pin is I/O port mode)
		f_{EXCLK}	Clock input invalid (pin is I/O port mode)
	f_{RL}		Operation stopped
PLL			
CPU			
Flash memory			
RAM			
Regulator		For chip	Operable
		For USB	
Port (latch)			Operation stopped
16-bit timer/event counter 00			
8-bit timer/event counter		50	
		51	
8-bit timer H1			
Watchdog timer			
Serial interface		UART6	
		CSI10	
		USB	
Power-on-clear function			Operable
Low-voltage detection function			Operation stopped
External interrupt			

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

Table 15-2. Hardware Statuses After Reset Acknowledgment (1/3)

Hardware		After Reset Acknowledgment[Note 1]
Program counter (PC)		The contents of the reset vector table (0000H, 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined[Note 2]
	General-purpose registers	Undefined[Note 2]
Port registers (P0, P1, P3, P6, P12) (output latches)		00H
Port mode registers (PM0, PM1, PM3, PM6, PM12)		FFH
Pull-up resistor option registers (PU0, PU1, PU3, PU12)		00H
Internal expansion RAM size switching register (IXS)		0CH[Note 3]
Internal memory size switching register (IMS)		CFH[Note 3]
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
PLL control register (PLLC)		00H
USB clock control register (UCKC)		00H
16-bit timer/event counter 00	Timer counter 00 (TM00)	0000H
	Capture/compare registers 000, 010 (CR000, CR010)	0000H
	Mode control register 00 (TMC00)	00H
	Prescaler mode register 00 (PRM00)	00H
	Capture/compare control register 00 (CRC00)	00H
	Timer output control register 00 (TOC00)	00H
8-bit timer/event counters 50, 51	Timer counters 50, 51 (TM50, TM51)	00H
	Compare registers 50, 51 (CR50, CR51)	00H
	Timer clock selection registers 50, 51 (TCL50, TCL51)	00H
	Mode control registers 50, 51 (TMC50, TMC51)	00H

Notes 1. During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

2. When a reset is executed in the standby mode, the pre-reset status is held even after reset.

3. The initial values of the internal memory size switching register (IMS) and internal expansion RAM size switching register (IXS) after a reset release are fixed (IMS = CFH, IXS = 0CH), 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 (μPD78F0730)	IMS	IXS
μPD78F0730	C4H	08H

Table 15-2. Hardware Statuses After Reset Acknowledgment (2/3)

Hardware		Status After Reset Acknowledgment[Note 1]
8-bit timer H1	Compare registers 01, 11 (CMP01, CMP11)	00H
	Mode register (TMHMD1)	00H
	Carrier control register 1 (TMCYC1)	00H
Watchdog timer	Enable register (WDTE)	1AH/9AH[Note 2]
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
Serial interfaces CSI10	Transmit buffer register 10 (SOTB10)	00H
	Serial I/O shift register 10 (SIO10)	00H
	Serial operation mode register 10 (CSIM10)	00H
	Serial clock selection register 10 (CSIC10)	00H
USB function controller USBF	UF0 EP0NAK register (UF0E0N)	00H
	UF0 EP0NAKALL register (UF0E0NA)	00H
	UF0 EPNAK register (UF0EN)	00H
	UF0 EPNAK mask register (UF0ENM)	00H
	UF0 SNDSIE register (UF0SDS)	00H
	UF0 CLR request register (UF0CLR)	00H
	UF0 SET request register (UF0SET)	00H
	UF0 EP status n register (UF0EPSn) (n = 0 to 2)	00H
	UF0 INT status n register (UF0ISn) (n = 0 to 4)	00H
	UF0 INT mask n register (UF0IMn) (n = 0 to 4)	00H
	UF0 INT clear n register (UF0ICn) (n = 0 to 4)	FFH
	UF0 FIFO clear n register (UF0FICn) (n = 0, 1)	00H
	UF0 data end register (UF0DEND)	00H
	UF0 GPR register (UF0GPR)	00H
	UF0 mode control register (UF0MODC)	00H
	UF0 mode status register (UF0MODS)	00H
	UF0 active interface number register (UF0AIFN)	00H
	UF0 active alternative setting register (UF0AAS)	00H
	UF0 alternative setting status register (UF0ASS)	00H
	UF0 endpoint n interface mapping register (UF0EnIM) (n = 1, 2)	00H
	UF0 EP0 read register (UF0E0R)	Undefined
	UF0 EP0 length register (UF0E0L)	00H
	UF0 EP0 setup register (UF0E0ST)	00H

Notes 1. During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

2. The reset value of WDTE is determined by the option byte setting.

Table 15-2. Hardware Statuses After Reset Acknowledgment (3/3)

Hardware		Status After Reset Acknowledgment[Note 1]
USB function controller USBF	UF0 EP0 write register (UF0E0W)	00H
	UF0 bulk-out 1 register (UF0BO1)	Undefined
	UF0 bulk-out 1 length register (UF0BO1L)	00H
	UF0 bulk-in 1 register (UF0BI1)	00H
	UF0 device status register (UF0DSTL)	00H
	UF0 EPn status register L (UF0EPnSL) (n = 0 to 2)	00H
	UF0 address register (UF0ADRS)	00H
	UF0 configuration register (UF0CNF)	00H
	UF0 interface n register (UF0IFn) (n = 0 to 4)	00H
	UF0 descriptor length register (UF0DSCL)	00H
	UF0 device descriptor register n (UF0DDn) (n = 0 to 17)	Undefined
	UF0 configuration/interface/endpoint descriptor register n (UF0CIEn) (n = 0 to 255)	Undefined
	USB function 0 buffer control register (UF0BC)	00H
Reset function	Reset control flag register (RESF)	00H[Note 2]
Low-voltage detector	Low-voltage detection register (LVIM)	00H[Note 2]
	Low-voltage detection level selection register (LVIS)	00H[Note 2]
Interrupt	Request flag registers 0L, 0H, 1L, 1H (IF0L, IF0H, IF1L, IF1H)	00H
	Mask flag registers 0L, 0H, 1L, 1H (MK0L, MK0H, MK1L, MK1H)	FFH
	Priority specification flag registers 0L, 0H, 1L, 1H (PR0L, PR0H, PR1L, PR1H)	FFH
	External interrupt rising edge enable register (EGP)	00H
	External interrupt falling edge enable register (EGN)	00H

Notes 1. During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

2. These values vary depending on the reset source.

Register \ Reset Source		$\overline{\text{RESET}}$ 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					

15.1 Register for Confirming Reset Source

Many internal reset generation sources exist in the µPD78F0730. 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 15-5. Format of Reset Control Flag Register (RESF)

Address: FFACH After reset: 00H[Note] R

Symbol	7	6	5	4	3	2	1	0
RESF	0	0	0	WDTRF	0	0	0	LVIRF

WDTRF	Internal reset request by watchdog timer (WDT)
0	Internal reset request is not generated, or RESF is cleared.
1	Internal reset request is generated.

LVIRF	Internal reset request by low-voltage detector (LVI)
0	Internal reset request is not generated, or RESF is cleared.
1	Internal reset request is generated.

Note The value after reset varies depending on the reset source.

Caution Do not read data by a 1-bit memory manipulation instruction.

The status of RESF when a reset request is generated is shown in Table 15-3.

Table 15-3. 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 16 POWER-ON-CLEAR CIRCUIT

16.1 Functions of Power-on-Clear Circuit

The power-on-clear circuit (POC) has the following functions.

- Generates internal reset signal at power on.
  In the 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)[Note], the reset signal is released when the supply voltage (V_{DD}) exceeds 2.7 V ±0.2 V.

- Compares supply voltage (V_{DD}) and detection voltage (V_{POC} = 1.59 V ±0.15 V), generates internal reset signal when $V_{DD} < V_{POC}$, and releases reset when $V_{DD} \geq V_{DDPOC}$.

<R> **Note** For the µPD78F0730, be sure to set the 2.7 V/1.59 V POC mode by using the option byte (POCMODE = 1). Also, design the circuit so the supply voltage (V_{DD}) rises sufficiently fast and reaches at 4.0 V within 1.94 ms after reset is released by the POC.

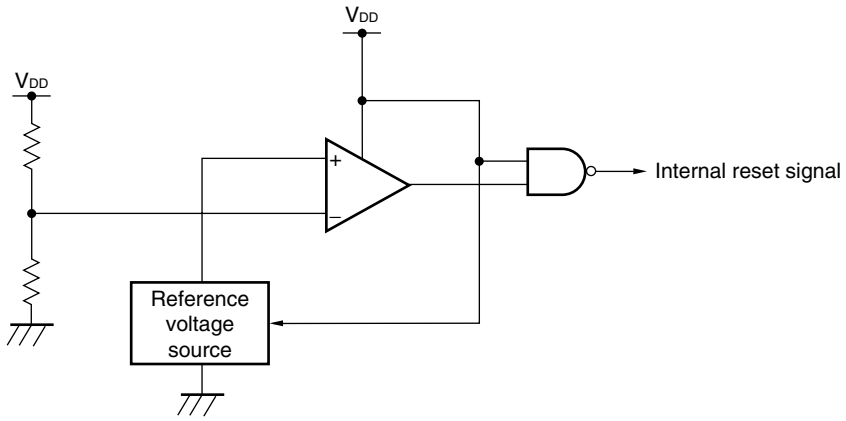
Caution If an internal reset signal is generated in the POC circuit, the reset control flag register (RESF) is cleared to 00H.

Remark This product incorporates multiple hardware functions that generate an internal reset signal. A flag that indicates the reset source is located in the reset control flag register (RESF) for when an internal reset signal is generated by the watchdog timer (WDT) 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 15 RESET FUNCTION**.

16.2 Configuration of Power-on-Clear Circuit

The block diagram of the power-on-clear circuit is shown in Figure 16-1.

Figure 16-1. Block Diagram of Power-on-Clear Circuit



16.3 Operation of Power-on-Clear Circuit

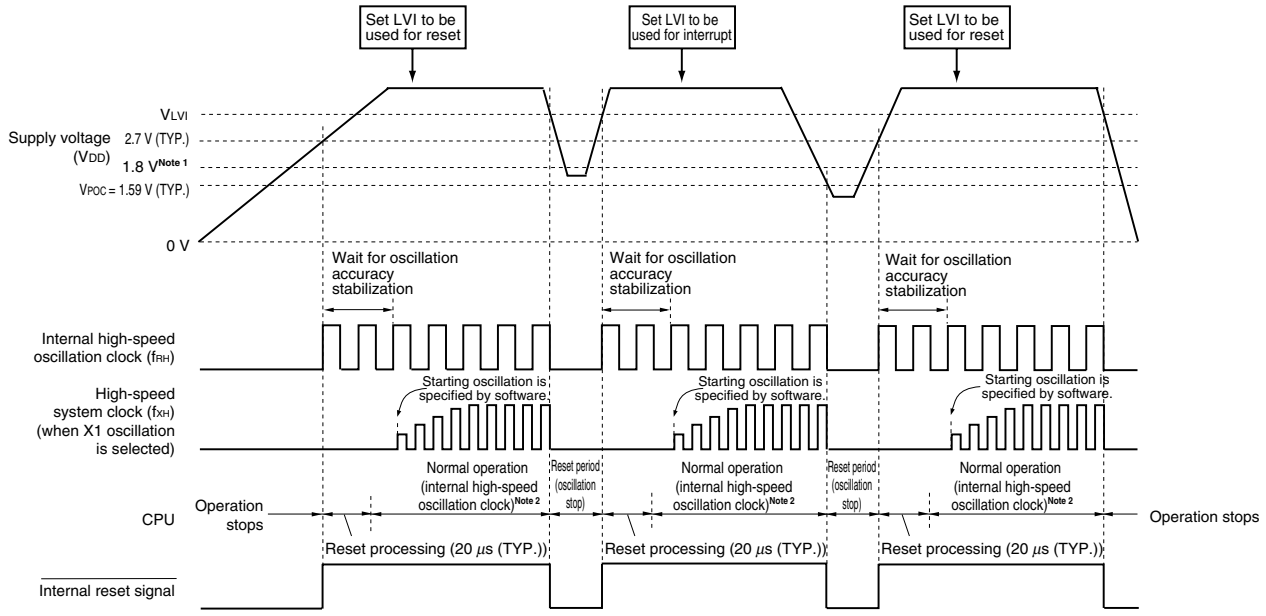
In 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)

- An internal reset signal is generated on power application. When the supply voltage (V_{DD}) exceeds the detection voltage (V_{DDPOC} = 2.7 V ±0.2 V), the reset status is released.
- The supply voltage (V_{DD}) and detection voltage (V_{POC} = 1.59 V ±0.15 V) are compared. When $V_{DD} < V_{POC}$, the internal reset signal is generated. It is released when $V_{DD} \geq V_{DDPOC}$.

The timing of generation of the internal reset signal by the power-on-clear circuit and low-voltage detector is shown below.

Figure 16-2. Timing of Generation of Internal Reset Signal by Power-on-Clear Circuit and Low-Voltage Detector

In 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)



Notes 1. The operation guaranteed range is 1.8 V ≤ V_{DD} ≤ 5.5 V. To make the state at lower than 1.8 V reset state when the supply voltage falls, use the reset function of the low-voltage detector, or input the low level to the $\overline{\text{RESET}}$ pin.

2. The internal high-speed oscillation clock and a high-speed system clock can be selected as the CPU 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 17 LOW-VOLTAGE DETECTOR).

Remarks 1. V_{LVI}: LVI detection voltage

2. V_{POC}: POC detection voltage

3. For the µPD78F0730, be sure to set the 2.7 V/1.59 V POC mode by using the option byte (POCMODE = 1).

16.4 Cautions for Power-on-Clear Circuit

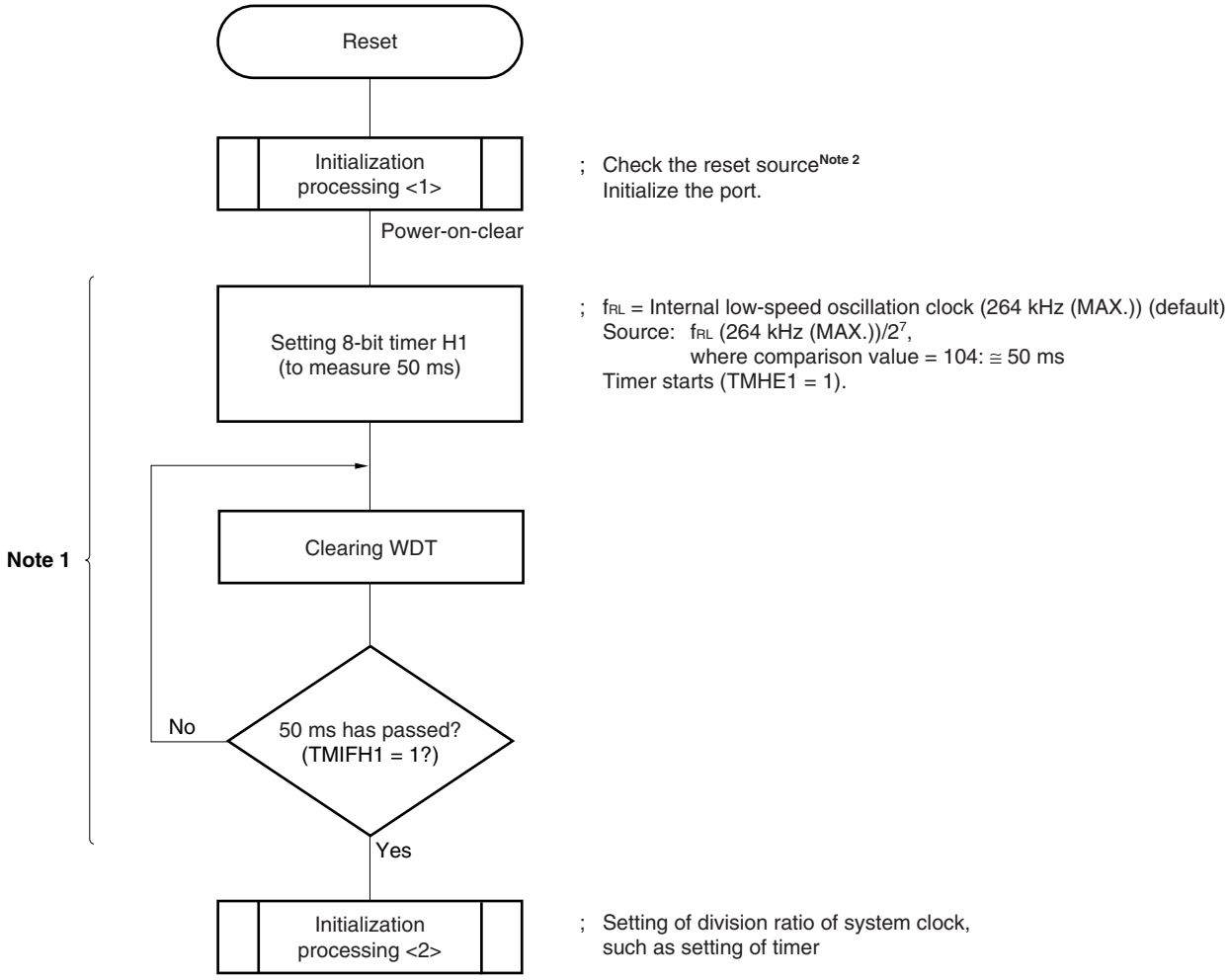
In a system where the supply voltage (V_{DD}) fluctuates for a certain period in the vicinity of the POC detection voltage (V_{POC}), the system may be repeatedly reset and released from the reset status. In this case, the time from release of reset to the start of the operation of the microcontroller 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 16-3. Example of Software Processing After Reset Release (1/2)

• If supply voltage fluctuation is 50 ms or less in vicinity of POC detection voltage

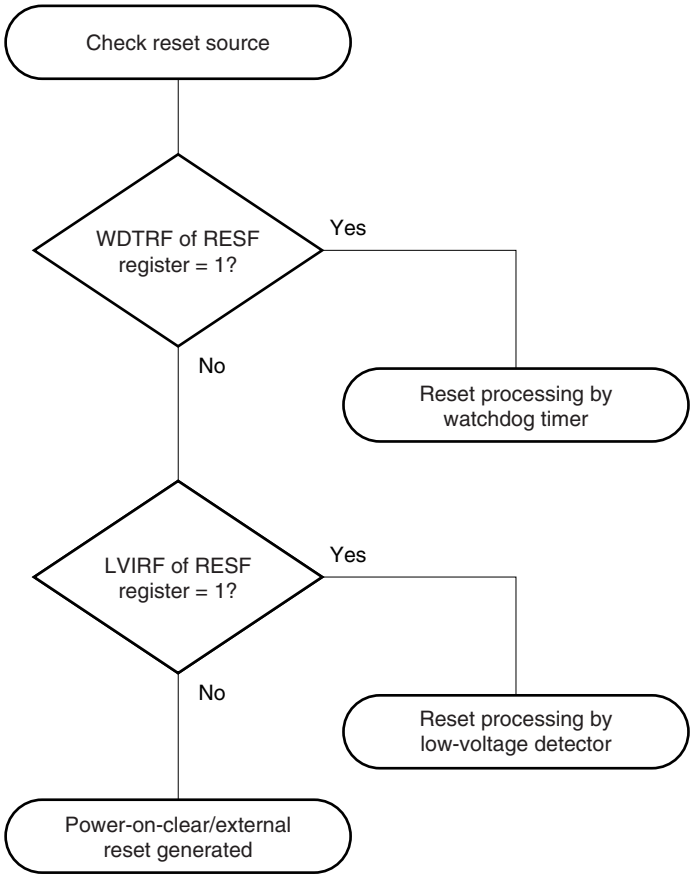


Notes 1. If reset is generated again during this period, initialization processing <2> is not started.

2. A flowchart is shown on the next page.

Figure 16-3. Example of Software Processing After Reset Release (2/2)

• Checking reset source



CHAPTER 17 LOW-VOLTAGE DETECTOR

17.1 Functions of Low-Voltage Detector

The low-voltage detector (LVI) has the following functions.

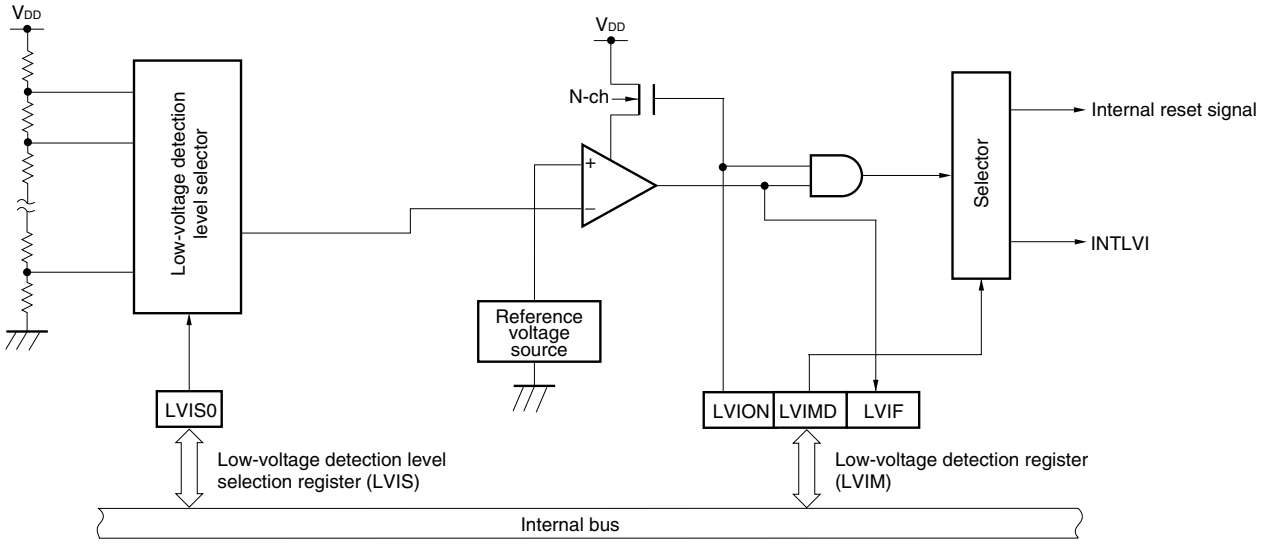
- Compares supply voltage (V_{DD}) and detection voltage (V_{LVI}), and generates an internal interrupt signal or internal reset signal when $V_{DD} < V_{LVI}$. Detection levels (2 levels) of supply voltage can be changed by software.

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 15 RESET FUNCTION**.

17.2 Configuration of Low-Voltage Detector

The block diagram of the low-voltage detector is shown in Figure 17-1.

Figure 17-1. Block Diagram of Low-Voltage Detector



17.3 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.

Reset signal generation sets LVIM to 00H.

Figure 17-2. Format of Low-Voltage Detection Register (LVIM)

Address: FFBEH After reset: 00H R/W[Note 1]

Symbol	<7>	6	5	4	3	2	<1>	<0>
LVIM	LVION	0	0	0	0	0	LVIMD	LVIF

LVION[Notes 2, 3]	Enables low-voltage detection operation
0	Disables operation
1	Enables operation

LVIMD[Note 2]	Low-voltage detection operation mode selection
0	Generates interrupt signal when supply voltage (V_{DD}) < detection voltage (V_{LVI})
1	Generates internal reset signal when supply voltage (V_{DD}) < detection voltage (V_{LVI})

LVIF[Note 4]	Low-voltage detection flag
0	Supply voltage (V_{DD}) ≥ detection voltage (V_{LVI}), or when operation is disabled
1	Supply voltage (V_{DD}) < detection voltage (V_{LVI})

Notes 1. Bit 0 is read-only.

2. LVION and LVIMD 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.

3. 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 µs (MAX.)) when LVION is set to 1 until the voltage is confirmed at LVIF.

4. The value of LVIF is output as the interrupt request signal INTLVI when LVION = 1 and LVIMD = 0.

Caution. 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 LVIMD to 0 and then clear LVION to 0.**

(2) Low-voltage detection level selection register (LVIS)

This register selects the low-voltage detection level.

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

Reset signal generation input sets LVIS to 00H.

Figure 17-3. Format of Low-Voltage Detection Level Selection Register (LVIS)

Address: FFBFH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
LVIS	0	0	0	0	0	0	0	LVIS0

LVIS0	Detection level
0	V_{LVI0} (4.24 V ±0.1 V)
1	V_{LVI1} (4.09 V ±0.1 V)

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

2. Do not change the value of LVIS during LVI operation.

(3) Port mode register 12 (PM12)

When using the P120/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 17-4. 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)

17.4 Operation of Low-Voltage Detector

The low-voltage detector can be used in the following two modes.

(1) Used as reset

Compare the supply voltage (V_{DD}) and detection voltage (V_{LVI}), generate an internal reset signal when $V_{DD} < V_{LVI}$, and releases internal reset when $V_{DD} \geq V_{LVI}$.

(2) Used as interrupt

Compare the supply voltage (V_{DD}) and detection voltage (V_{LVI}), and generate an interrupt signal (INTLVI) when $V_{DD} < V_{LVI}$.

17.4.1 When used as reset

(1) When detecting level of supply voltage (V_{DD})

- When starting operation
  - <1> Mask the LVI interrupt (LVIMK = 1).
  - <2> Set the detection voltage using bit 0 (LVIS0) of the low-voltage detection level selection register (LVIS).
  - <3> Set bit 7 (LVION) of LVIM to 1 (enables LVI operation).
  - <4> Use software to wait for an operation stabilization time (10 μs (MAX.)).
  - <5> Wait until it is checked that (supply voltage (V_{DD}) ≥ detection voltage (V_{LVI})) by bit 0 (LVIF) of LVIM.
  - <6> Set bit 1 (LVIMD) of LVIM to 1 (generates internal reset signal when supply voltage (V_{DD}) < detection voltage (V_{LVI})).

  Figure 17-5 shows the timing of the internal reset signal generated by the low-voltage detector. The numbers in this timing chart correspond to <1> to <6> above.

  Cautions 1. <1> must always be executed. When LVIMK = 0, an interrupt may occur immediately after the processing in <3>.

  2. If supply voltage (V_{DD}) ≥ detection voltage (V_{LVI}) when LVIMD is set to 1, an internal reset signal is not generated.

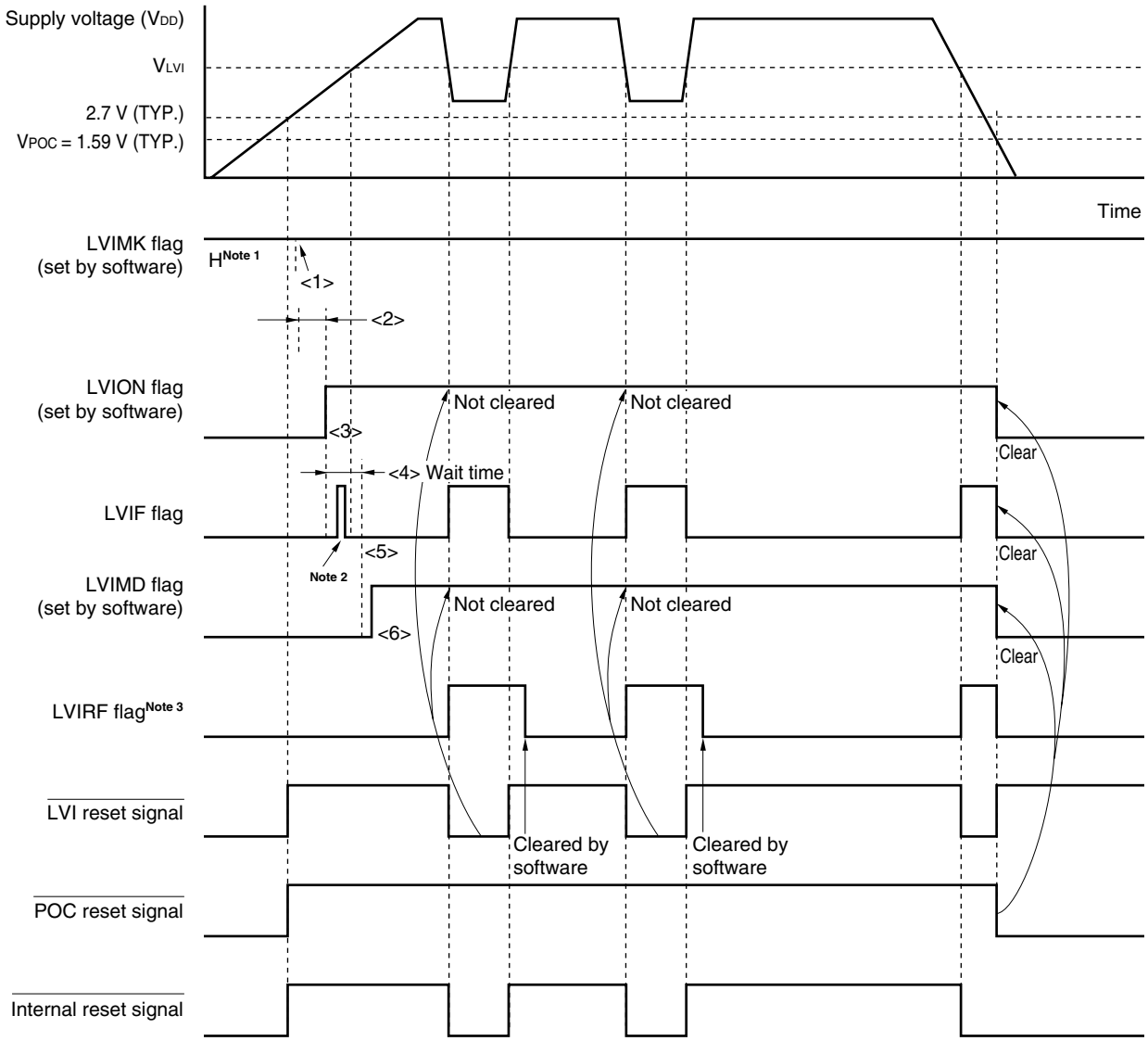
- When stopping operation
  Either of the following procedures must be executed.

  - 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 17-5. Timing of Low-Voltage Detector Internal Reset Signal Generation (Detects Level of Supply Voltage (V_{DD}))

In 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)



Notes 1. The LVIMK flag is set to "1" by reset signal generation.

2. The LVIF flag may be set (1).

3. LVIRF is bit 0 of the reset control flag register (RESF). For details of RESF, see **CHAPTER 15 RESET FUNCTION**.

Remarks 1. <1> to <6> in Figure 17-5 above correspond to <1> to <6> in the description of "When starting operation" in **17.4.1 (1) When detecting level of supply voltage (V_{DD})**.

2. For the µPD78F0730, be sure to set the 2.7 V/1.59 V POC mode by using the option byte (POCMODE = 1).

17.4.2 When used as interrupt

(1) When detecting level of supply voltage (V_{DD})

- When starting operation
  - <1> Mask the LVI interrupt (LVIMK = 1).
  - <2> Set the detection voltage using bit 0 (LVIS0) of the low-voltage detection level selection register (LVIS).
  - <3> Set bit 7 (LVION) of LVIM to 1 (enables LVI operation).
  - <4> Use software to wait for an operation stabilization time (10 μs (MAX.)).
  - <5> Confirm that "supply voltage (V_{DD}) ≥ detection voltage (V_{LVI})" at bit 0 (LVIF) of LVIM.
  - <6> Clear the interrupt request flag of LVI (LVIIF) to 0.
  - <7> Release the interrupt mask flag of LVI (LVIMK).
  - <8> Clear bit 1 (LVIMD) of LVIM to 0 (generates interrupt signal when supply voltage (V_{DD}) < detection voltage (V_{LVI})) (default value).
  - <9> Execute the EI instruction (when vector interrupts are used).

  Figure 17-6 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.

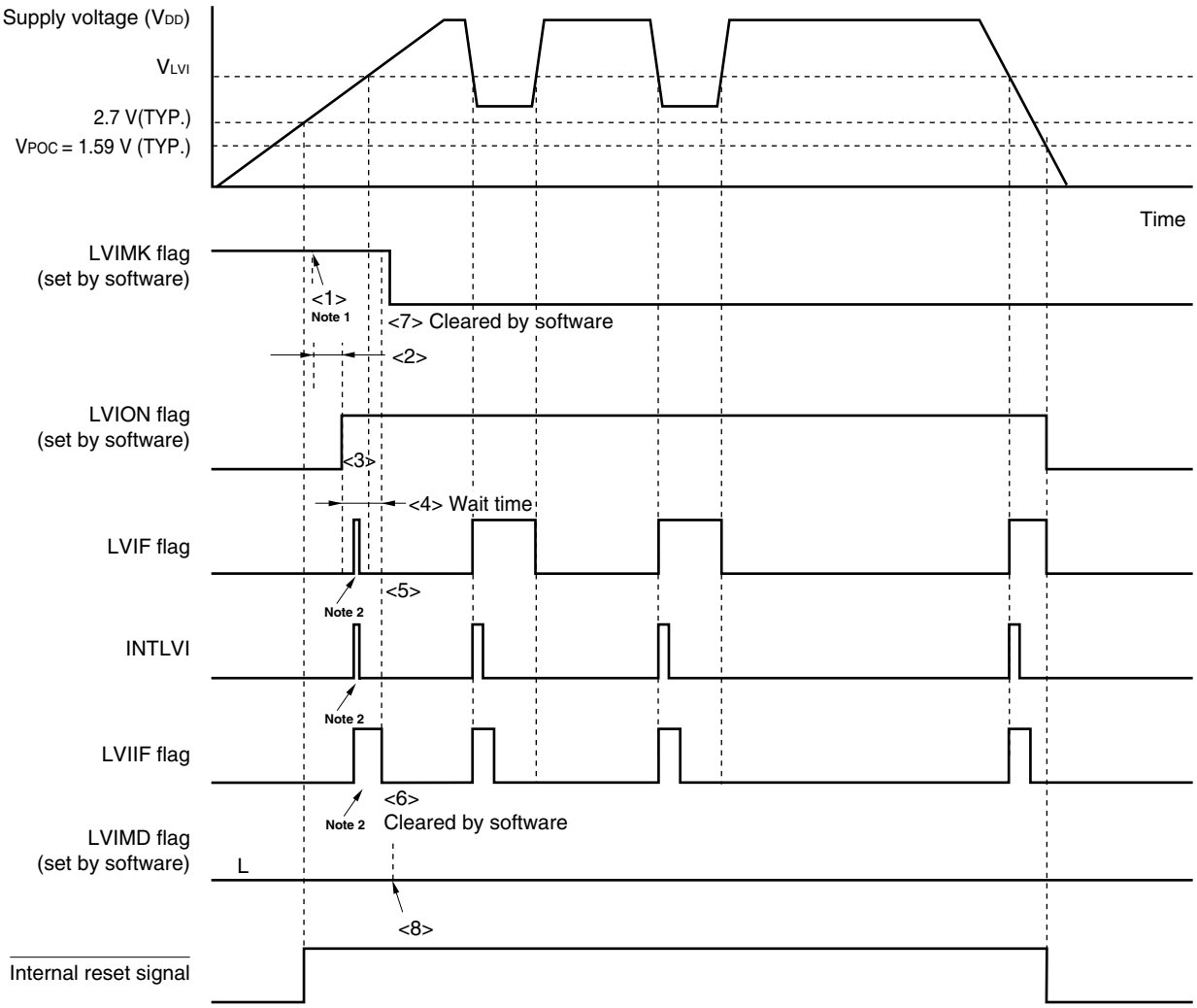
- When stopping operation
  Either of the following procedures must be executed.

  - When using 8-bit memory manipulation instruction:
    Write 00H to LVIM.

  - When using 1-bit memory manipulation instruction:
    Clear LVION to 0.

Figure 17-6. Timing of Low-Voltage Detector Interrupt Signal Generation (Detects Level of Supply Voltage (V_{DD}))

In 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)



Notes 1. The LVIMK flag is set to "1" by reset signal generation.

2. The interrupt request signal (INTLVI) is generated and the LVIF and LVIIF flags may be set (1).

Remarks 1. <1> to <8> in Figure 17-6 above correspond to <1> to <8> in the description of "When starting operation" in **17.4.2 (1) When detecting level of supply voltage (V_{DD})**.

2. For the μPD78F0730, be sure to set the 2.7 V/1.59 V POC mode by using the option byte (POCMODE = 1).

17.5 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.

In this system, take the following actions.

<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 17-7**).

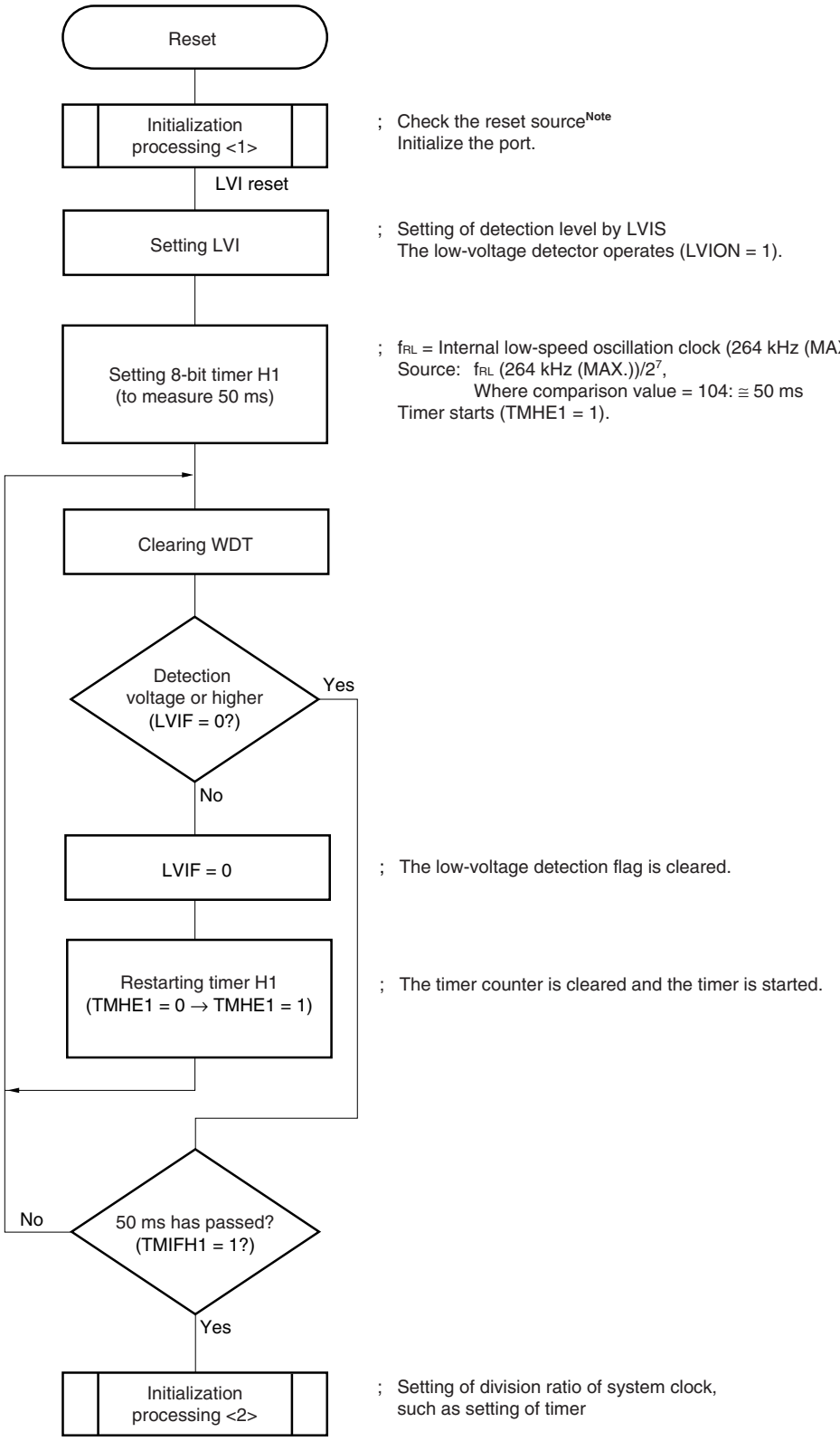
(2) When used as interrupt

(a) Check that "supply voltage (V_{DD}) ≥ detection voltage (V_{LVI})" 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, check that "supply voltage (V_{DD}) ≥ detection voltage (V_{LVI})" using the LVIF flag, and clear the LVIIF flag to 0.

Figure 17-7. Example of Software Processing After Reset Release (1/2)

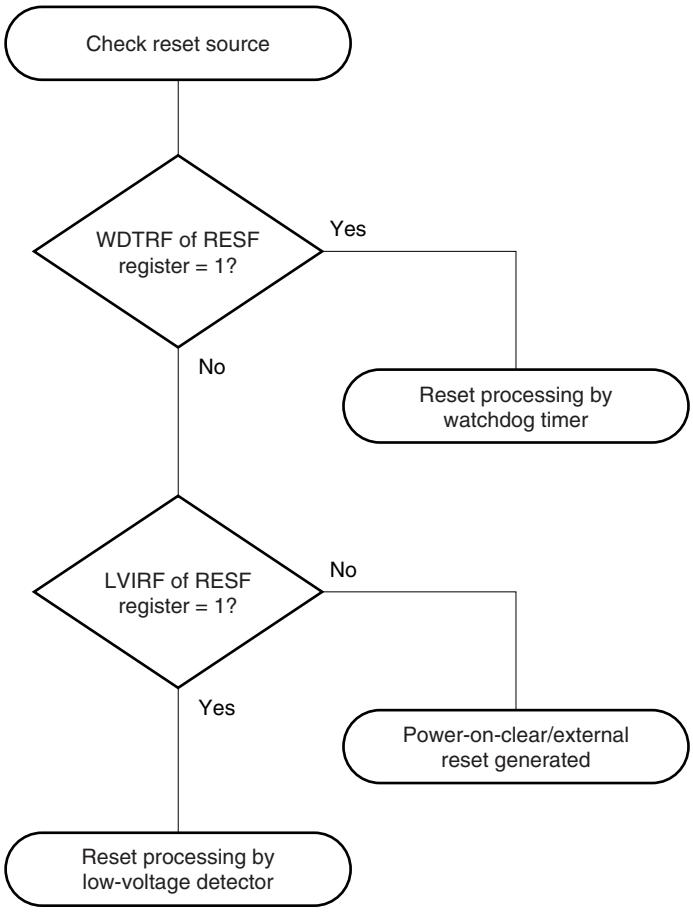
• If supply voltage fluctuation is 50 ms or less in vicinity of LVI detection voltage



Note A flowchart is shown on the next page.

Figure 17-7. Example of Software Processing After Reset Release (2/2)

• Checking reset source



CHAPTER 18 OPTION BYTE

18.1 Functions of Option Bytes

The flash memory at 0080H to 0084H of the µPD78F0730 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.

When the boot swap operation is used during self-programming, 0080H to 0084H are switched to 1080H to 1084H. Therefore, set values that are the same as those of 0080H to 0084H to 1080H to 1084H in advance.

(1) 0080H/1080H

- ○ 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) 0081H/1081H

- ○ Selecting POC mode
  - During 2.7 V/1.59 V POC mode operation (POCMODE = 1)
    The device is in the reset state upon power application and until the supply voltage reaches 2.7 V (TYP.). It is released from the reset state when the voltage exceeds 2.7 V (TYP.). After that, POC is not detected at 2.7 V but is detected at 1.59 V (TYP.).
  - During 1.59 V POC mode operation (POCMODE = 0)
    The device is in the reset state upon power application and until the supply voltage reaches 1.59 V (TYP.). It is released from the reset state when the voltage exceeds 1.59 V (TYP.). After that, POC is detected at 1.59 V (TYP.), in the same manner as on power application.

Caution For the µPD78F0730, be sure to set POCMODE to 1.

(3) 0084H/1084H

- ○ 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

Caution To use the on-chip debug function with a product equipped with the on-chip debug function (µPD78F0730), set 02H or 03H to 0084H. Set a value that is the same as that of 0084H to 1084H because 0084H and 1084H are switched at boot swapping.

Caution Be sure to set 00H to 0082H and 0083H (0082H/1082H and 0083H/1083H when the boot swap function is used).

18.2 Format of Option Byte

The format of the option byte is shown below.

Figure 18-1. Format of Option Byte (1/2)

Address: 0080H/1080H[Note]

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	Setting prohibited
0	1	
1	0	
1	1	100%

WDTON	Operation control of watchdog timer counter/illegal access detection
0	Counter operation disabled (counting stopped after reset), illegal access detection operation disabled
1	Counter operation enabled (counting started after reset), illegal access detection operation enabled

WDCS2	WDCS1	WDCS0	Watchdog timer overflow time
0	0	0	$2^{10}/f_{RL}$ (3.88 ms)
0	0	1	$2^{11}/f_{RL}$ (7.76 ms)
0	1	0	$2^{12}/f_{RL}$ (15.52 ms)
0	1	1	$2^{13}/f_{RL}$ (31.03 ms)
1	0	0	$2^{14}/f_{RL}$ (62.06 ms)
1	0	1	$2^{15}/f_{RL}$ (124.12 ms)
1	1	0	$2^{16}/f_{RL}$ (248.24 ms)
1	1	1	$2^{17}/f_{RL}$ (496.48 ms)

LSROSC	Internal low-speed oscillator operation
0	Can be stopped by software (stopped when 1 is written to bit 0 (LSRSTOP) of RCM register)
1	Cannot be stopped (not stopped even if 1 is written to LSRSTOP bit)

Note Set a value that is the same as that of 0080H to 1080H because 0080H and 1080H are switched during the boot swap operation.

Cautions 1. The watchdog timer does not stop during self-programming of the flash memory and EEPROM emulation. During processing, the interrupt acknowledge time is delayed. Set the overflow time taking this delay into consideration.

2. If LSROSC = 0 (oscillation can be stopped by software), the count clock is not supplied to the watchdog timer in the HALT and STOP modes, regardless of the setting of bit 0 (LSRSTOP) of the internal oscillation mode register (RCM).

When 8-bit timer H1 operates with the internal low-speed oscillation clock, the count clock is supplied to 8-bit timer H1 even in the HALT/STOP mode.

3. Be sure to clear bit 7 to 0.

Remarks 1. f_{RL}: Internal low-speed oscillation clock frequency

2. ( ): f_{RL} = 264 kHz (MAX.)

Figure 18-1. Format of Option Byte (2/2)

Address: 0081H/1081H[Notes 1, 2]

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	POCMODE

POCMODE	POC mode selection
0	1.59 V POC mode (default)
1	2.7 V/1.59 V POC mode

Notes 1. POCMODE can only be written by using a dedicated flash programmer. It cannot be set during self-programming or boot swap operation during self-programming (at this time, 1.59 V POC mode (default) is set). However, because the value of 1081H is copied to 0081H during the boot swap operation, it is recommended to set a value that is the same as that of 0081H to 1081H when the boot swap function is used.

2. To change the setting for the POC mode, set the value to 0081H again after batch erasure (chip erasure) of the flash memory. The setting cannot be changed after the memory of the specified block is erased.

Caution For the µPD78F0730, be sure to set 1 to bit 0, and be sure to clear bits 7 to 1 to 0.

Address: 0082H/1082H, 0083H/1083H[Note]

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

Note Be sure to set 00H to 0082H and 0083H, as these addresses are reserved areas. Also set 00H to 1082 and 1083H because 0082H and 0083H are switched with 1082H and 1083H when the boot swap operation is used.

Address: 0084H/1084H[Note]

7	6	5	4	3	2	1	0
0	0	0	0	0	0	OCDEN1	OCDEN0

OCDEN1	OCDEN0	On-chip debug operation control
0	0	Operation disabled
0	1	Setting prohibited
1	0	Operation enabled. Does not erase data of the flash memory in case authentication of the on-chip debug security ID fails.
1	1	Operation enabled. Erases data of the flash memory in case authentication of the on-chip debug security ID fails.

Note To use the on-chip debug function with a product equipped with the on-chip debug function (µPD78F0730), set 02H or 03H to 0084H. Set a value that is the same as that of 0084H to 1084H because 0084H and 1084H are switched at boot swapping.

Remark For the on-chip debug security ID, see **CHAPTER 20 ON-CHIP DEBUG FUNCTION**.

Here is an example of description of the software for setting the option bytes.

```
OPT     CSEG   AT 0080H
OPTION: DB     70H      ; Enables watchdog timer operation (illegal access detection operation),
                        ; Window open period of watchdog timer: 100%,
                        ; Overflow time of watchdog timer: 2^10/fRL,
                        ; Internal low-speed oscillator can be stopped by software.
        DB     01H      ; 2.7/1.59 V POC mode
        DB     00H      ; Reserved area
        DB     00H      ; Reserved area
        DB     00H      ; On-chip debug operation disabled
```

Remark Referencing of the option byte is performed during reset processing. For the reset processing timing, see **CHAPTER 15 RESET FUNCTION**.

CHAPTER 19 FLASH MEMORY

The μPD78F0730 incorporates the flash memory to which a program can be written, erased, and overwritten while mounted on the board.

19.1 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 IMS to C4H after a reset release.

Figure 19-1. 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
1	1	0	1024 bytes
Other than above			Setting prohibited

ROM3	ROM2	ROM1	ROM0	Internal ROM capacity selection
0	1	0	0	16 KB
Other than above				Setting prohibited

19.2 Internal Expansion RAM Size Switching Register

The internal expansion RAM capacity can be selected using the internal expansion RAM size switching register (IXS).
IXS is set by an 8-bit memory manipulation instruction.
Reset signal generation sets IXS to 0CH.

Caution Be sure to set to 08H after a reset release.

Figure 19-2. Format of Internal Expansion RAM Size Switching Register (IXS)

Address: FFF4H After reset: 0CH R/W

Symbol	7	6	5	4	3	2	1	0
IXS	0	0	0	IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0

IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	Internal expansion RAM capacity selection
0	1	0	0	0	2048 bytes
Other than above					Setting prohibited

19.3 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 μPD78F0730 has 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 μPD78F0730 is mounted on the target system.

Remark The FA series is a product of Naito Densei Machida Mfg. Co., Ltd.

Table 19-1. Wiring Between μPD78F0730 and Dedicated Flash Memory Programmer

Pin Configuration of Dedicated Flash Memory Programmer			With CSI10		With UART6	
Signal Name	I/O	Pin Function	Pin Name	Pin No.	Pin Name	Pin No.
SI/RxD	Input	Receive signal	SO10/P12	28	TxD6/P13	27
SO/TxD	Output	Transmit signal	SI10/P11	29	RxD6/P14	26
SCK	Output	Transfer clock	$\overline{\text{SCK10}}$/P10	30	–	–
CLK	Output	Clock to μPD78F0730	–Note 1	–	EXCLK/X2/P122Note 2	7
/RESET	Output	Reset signal	$\overline{\text{RESET}}$	5	$\overline{\text{RESET}}$	5
FLMD0	Output	Mode signal	FLMD0	6	FLMD0	6
V_{DD}	I/O	V_{DD} voltage generation/ power monitoring	V_{DD}	11	V_{DD}	11
			EV_{DD}	20	EV_{DD}	20
GND	–	Ground	V_{SS}	10	V_{SS}	10
			EV_{SS}	21	EV_{SS}	21

Notes 1. Only the internal high-speed oscillation clock (f_{RH}) can be used when CSI10 is used.

2. Only the X1 clock (f_X) or external main system clock (f_{EXCLK}) can be used when UART6 is used. 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.

- PG-FP5, FL-PR5, QB-MINI2: Connect CLK of the programmer to EXCLK/X2/P122 (pin 7).
- PG-FPL3, FP-LITE3: Connect CLK of the programmer to X1/P121 (pin 8), and connect its inverted signal to X2/EXCLK/P122 (pin 7).

Examples of the recommended connection when using the adapter for flash memory writing are shown below.

Figure 19-3. Example of Wiring Adapter for Flash Memory Writing in 3-Wire Serial I/O (CSI10) Mode

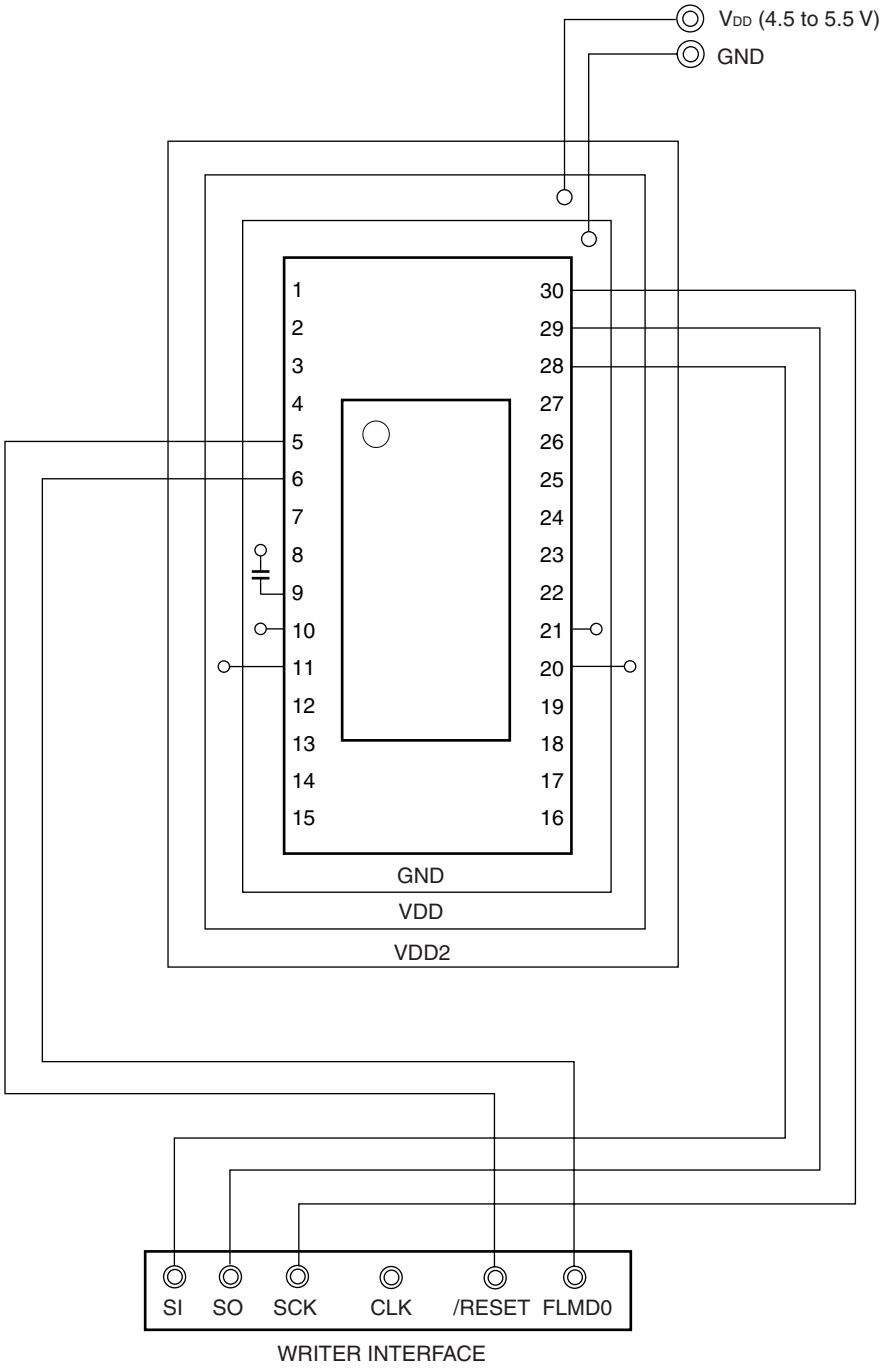
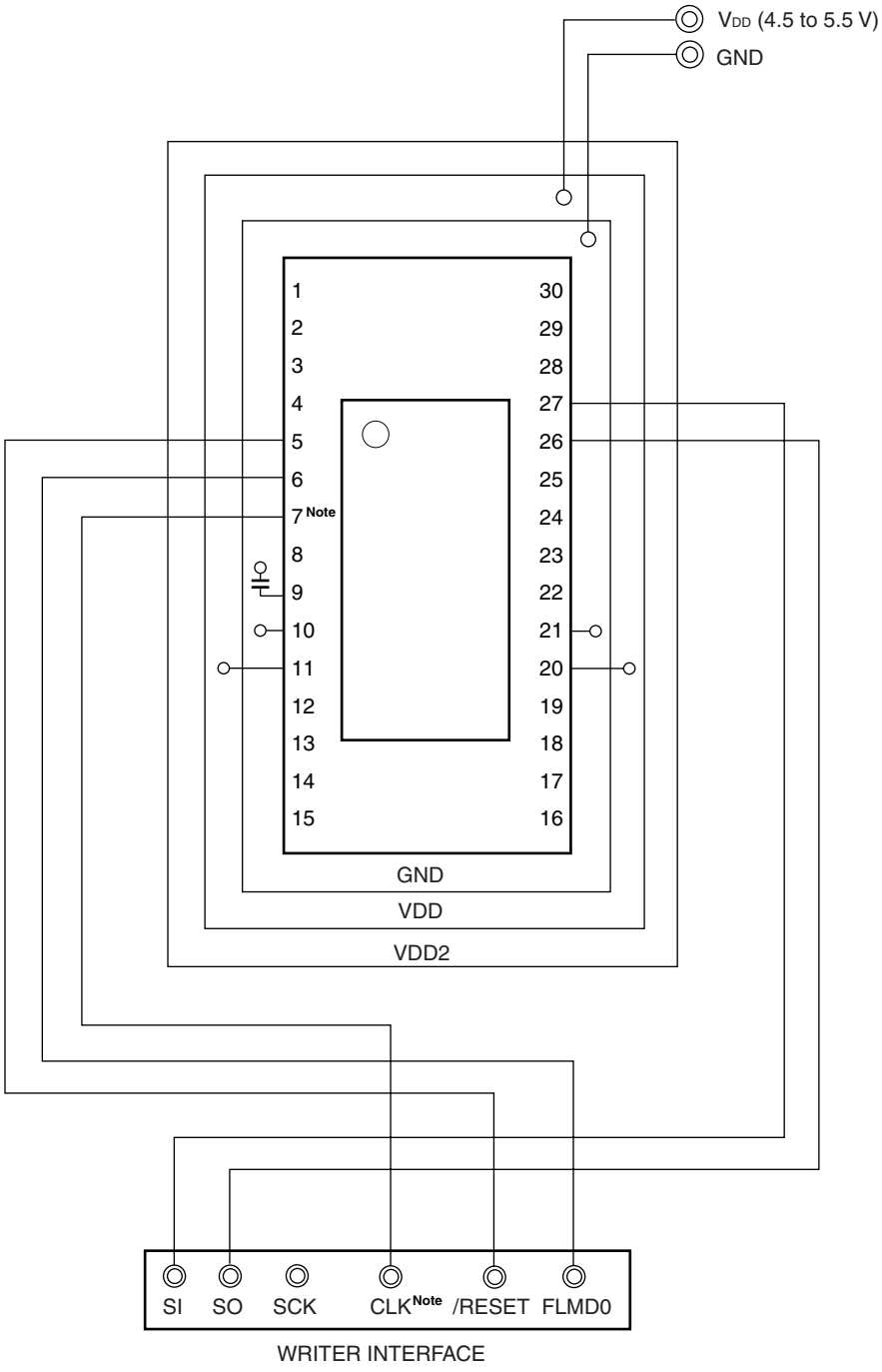


Figure 19-4. Example of Wiring Adapter for Flash Memory Writing in UART (UART6) Mode



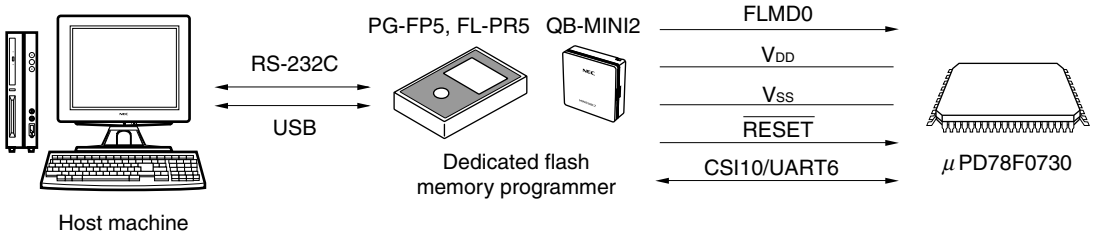
Note The above figure illustrates an example of wiring when using the clock output from the PG-FP5, FL-PR5, or QB-MINI2.

When using the clock output from the PG-FPL3 or FP-LITE3, connect CLK to X1/P121 (pin 8), and connect its inverted signal to X2/EXCLK/P122 (pin 7).

19.4 Programming Environment

The environment required for writing a program to the flash memory of the µPD78F0730 is illustrated below.

Figure 19-5. 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 µPD78F0730, 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.

19.5 Communication Mode

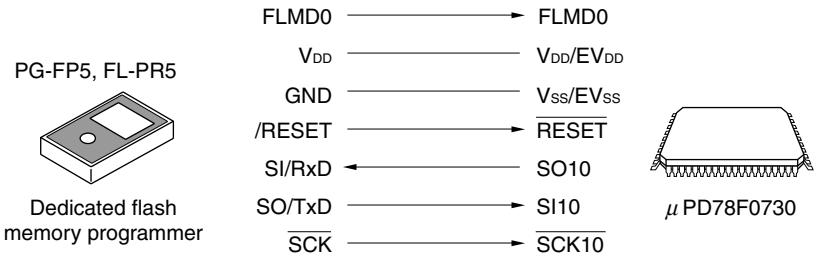
Communication between the dedicated flash memory programmer and the µPD78F0730 is established by serial communication via CSI10 or UART6 of the µPD78F0730.

(1) CSI10

Transfer rate: 2.4 kHz to 2.5 MHz

Figure 19-6. Communication with Dedicated Flash Memory Programmer (CSI10)

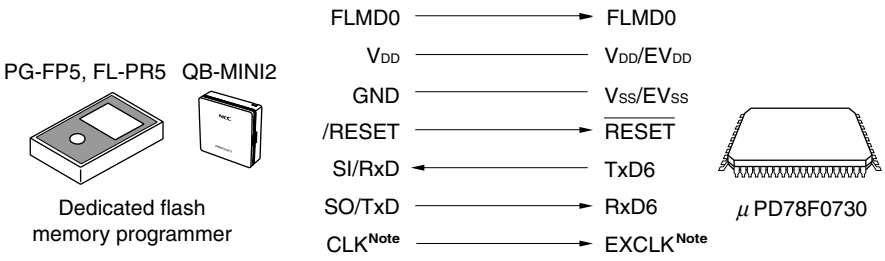
<R>



(2) UART6

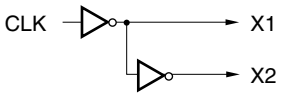
Transfer rate: 115200 bps

Figure 19-7. Communication with Dedicated Flash Memory Programmer (UART6)



Note The above figure illustrates an example of wiring when using the clock output from the PG-FP5, FL-PR5, or QB-MINI2.

When using the clock output from the PG-FPL3 or FP-LITE3, connect CLK to X1/P121 (pin 8), and connect its inverted signal to X2/EXCLK/P122 (pin 7).



The dedicated flash memory programmer generates the following signals for the µPD78F0730. For details, refer to the user's manual for the PG-FP5, FL-PR5, QB-MINI2, PG-FPL3, or FP-LITE3.

Table 19-2. Pin Connection

Dedicated Flash Memory Programmer			µPD78F0730	Connection	
Signal Name	I/O	Pin Function	Pin Name	CSI10	UART6
FLMD0	Output	Mode signal	FLMD0	◎	◎
VDD	I/O	VDD voltage generation/power monitoring	VDD, EVDD	◎	◎
GND	–	Ground	VSS, EVSS	◎	◎
CLK	Output	Clock output to µPD78F0730	**Note 1**	×Note 2	○Note 1
/RESET	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎
SI/RxD	Input	Receive signal	SO10/TxD6	◎	◎
SO/TxD	Output	Transmit signal	SI10/RxD6	◎	◎
SCK	Output	Transfer clock	$\overline{\text{SCK10}}$	◎	×

Notes 1. Only the X1 clock (fX) or external main system clock (fEXCLK) can be used when UART6 is 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.

- PG-FP5, FL-PR5, QB-MINI2: Connect CLK of the programmer to EXCLK/X2/P122 (pin 7).
- PG-FPL3, FP-LITE3: Connect CLK of the programmer to X1/P121 (pin 8), and connect its inverted signal to X2/EXCLK/P122 (pin 7).

2. Only the internal high-speed oscillation clock (fRH) can be used when CSI10 is used.

Remark ◎: Be sure to connect the pin.

○: The pin does not have to be connected if the signal is generated on the target board.

×: The pin does not have to be connected.

19.6 Connection of Pins on Board

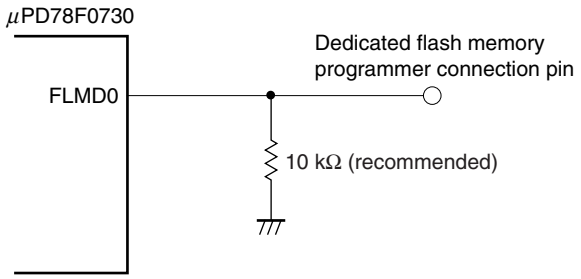
To write the flash memory on-board, connectors that connect the dedicated flash memory programmer must be provided on the target system. First provide a function that selects the normal operation mode or flash memory programming mode on the board.

When the flash memory programming mode is set, all the pins not used for programming the flash memory are in the same status as immediately after reset. Therefore, if the external device does not recognize the state immediately after reset, the pins must be connected as described below.

19.6.1 FLMD0 pin

In the normal operation mode, 0 V is input to the FLMD0 pin. In the flash memory programming mode, the V_{DD} write voltage is supplied to the FLMD0 pin. An FLMD0 pin connection example is shown below.

Figure 19-8. FLMD0 Pin Connection Example



19.6.2 Serial interface pins

The pins used by each serial interface are listed below.

Table 19-3. Pins Used by Each Serial Interface

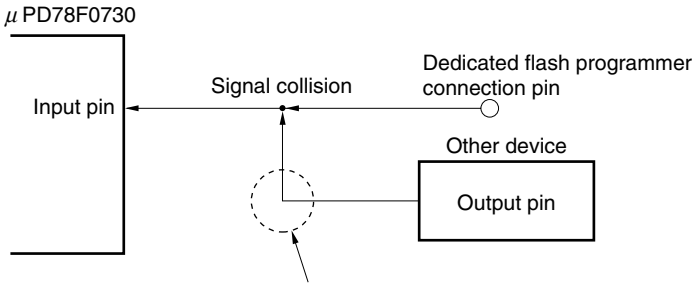
Serial Interface	Pins Used
CSI10	SO10, SI10, $\overline{\text{SCK}}$10
UART6	TxD6, RxD6

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 19-9. Signal Collision (Input Pin of Serial Interface)

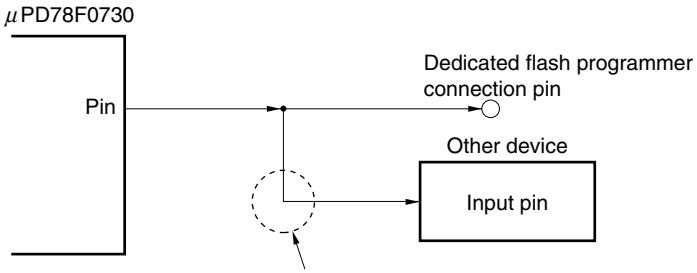


In the flash memory programming mode, the signal output by the device collides with the signal sent from the dedicated flash programmer. Therefore, isolate the signal of the other device.

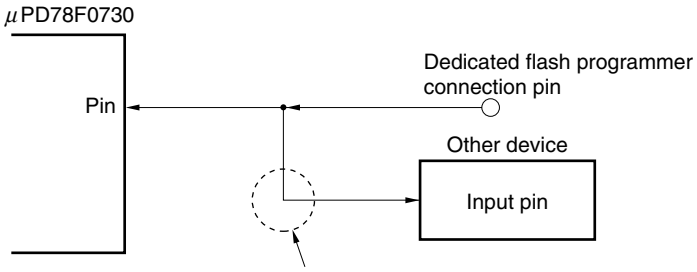
(2) Malfunction of other device

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 19-10. Malfunction of Other Device



If the signal output by the µPD78F0730 in the flash memory programming mode affects the other device, isolate the signal of the other device.



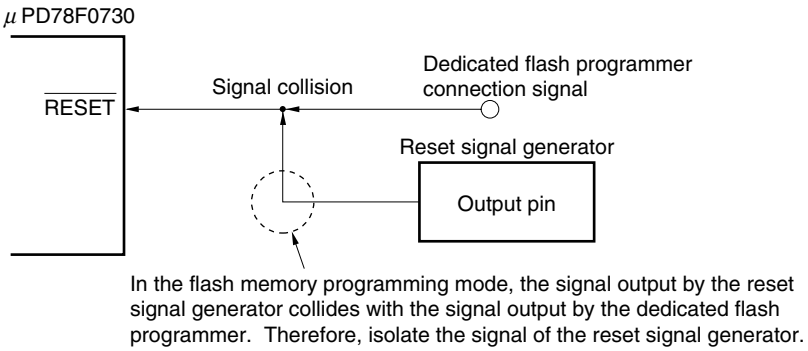
If the signal output by the dedicated flash programmer in the flash memory programming mode affects the other device, isolate the signal of the other device.

19.6.3 $\overline{\text{RESET}}$ pin

If the reset signal of the dedicated flash memory programmer is connected to the $\overline{\text{RESET}}$ pin that is connected to the reset signal generator on the board, signal collision takes place. To prevent this collision, isolate the connection with the reset signal generator.

If the reset signal is input from the user system while the flash memory programming mode is set, the flash memory will not be correctly programmed. Do not input any signal other than the reset signal of the dedicated flash memory programmer.

Figure 19-11. Signal Collision ($\overline{\text{RESET}}$ Pin)



19.6.4 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.

19.6.5 REGC pin

Connect the REGC pin to GND via a capacitor (0.47 to 1 µF: recommended) in the same manner as during normal operation.

19.6.6 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-FP5, FL-PR5, QB-MINI2: Connect CLK of the programmer to EXCLK/X2/P122.
- PG-FPL3, FP-LITE3: Connect CLK of the programmer to X1/P121, and connect its inverted signal to X2/EXCLK/P122.

Cautions 1. Only the internal high-speed oscillation clock (f_{RH}) can be used when CSI10 is used.

2. Only the X1 clock (f_X) or external main system clock (f_{EXCLK}) can be used when UART6 is used.

3. When writing the flash memory with a flash memory programmer, connect P31/INTP2/OCD1A and P121/X1/OCD0A as follows.

- **P31/INTP2/OCD1A: Connect to EV_{SS} via a resistor (10 kΩ: recommended).**
- **P121/X1/OCD0A: When using this pin as a port, connect it to V_{SS} via a resistor (10 kΩ: recommended) (in the input mode) or leave it open (in the output mode).**

The above connection is not necessary when writing the flash memory by means of self programming.

19.6.7 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.

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.

To use the on-board supply voltage, connect in compliance with the normal operation mode.

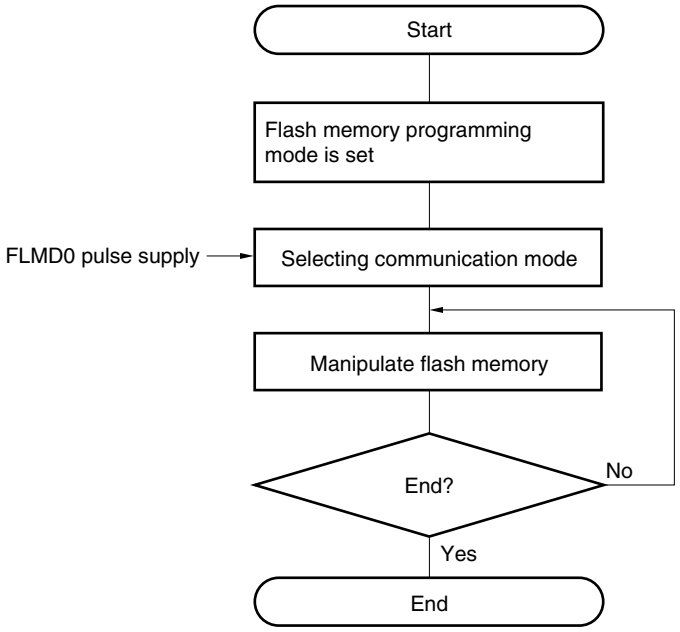
Supply the same other power supplies (EV_{DD} and EV_{SS}) as those in the normal operation mode.

19.7 Programming Method

19.7.1 Controlling flash memory

The following figure illustrates the procedure to manipulate the flash memory.

Figure 19-12. Flash Memory Manipulation Procedure



19.7.2 Flash memory programming mode

To rewrite the contents of the flash memory by using the dedicated flash memory programmer, set the µPD78F0730 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 19-13. Flash Memory Programming Mode

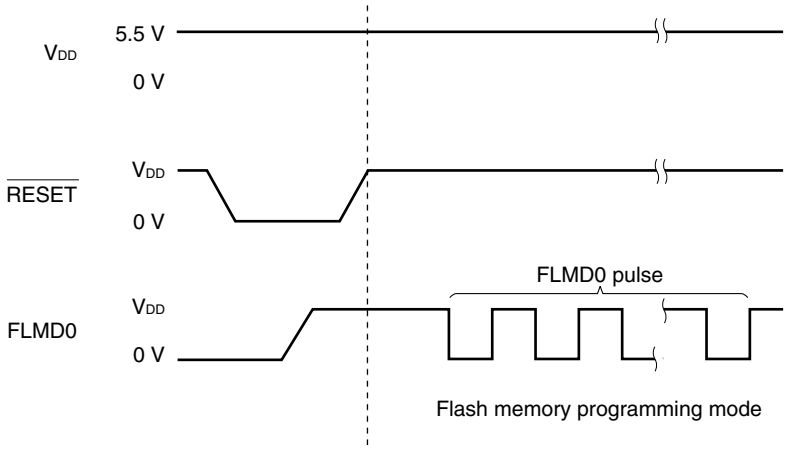


Table 19-4. Relationship Between FLMD0 Pin and Operation Mode After Reset Release

FLMD0	Operation Mode
0	Normal operation mode
V_{DD}	Flash memory programming mode

19.7.3 Selecting communication mode

In the µPD78F0730, a communication mode is selected by inputting pulses (up to 11 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 19-5. Communication Modes

Communication Mode	Standard Setting[Note]				Pins Used	Peripheral Clock	Number of FLMD0 Pulses
	Port	Speed	Frequency	Multiply Rate			
UART (UART6)	UART-Ext-Osc	115,200 bps	16 MHz	1.0	TxD6, RxD6	f_X	0
	UART-Ext-FP5CK					f_{EXCLK}	3
3-wire serial I/O (CSI10)	CSI-Internal-OSC	2.4 kHz to 2.5 MHz	16 MHz		SO10, SI10, $\overline{SCK10}$	f_{RH}	8

Note Selection items for Standard settings on GUI of the dedicated flash memory programmer.

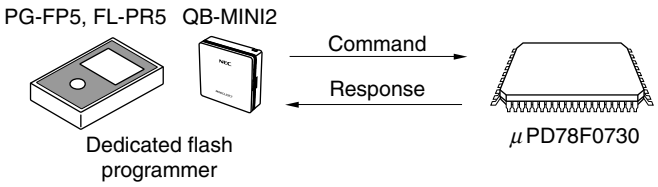
Caution When UART6 is selected, the receive clock is calculated based on the reset command sent from the dedicated flash memory programmer after the FLMD0 pulse has been received.

Remark f_X: X1 clock
f_{EXCLK}: External main system clock
f_{RH}: Internal high-speed oscillation clock

19.7.4 Communication commands

The μPD78F0730 communicates with the dedicated flash memory programmer by using commands. The signals sent from the flash memory programmer to the μPD78F0730 are called commands, and the signals sent from the μPD78F0730 to the dedicated flash memory programmer are called response.

Figure 19-14. Communication Commands



The flash memory control commands of the μPD78F0730 are listed in the table below. All these commands are issued from the programmer and the μPD78F0730 perform processing corresponding to the respective commands.

Table 19-6. Flash Memory Control Commands

Classification	Command Name	Function
Verify	Verify	Compares the contents of a specified area of the flash memory with data transmitted from the programmer.
Erase	Chip Erase	Erases the entire flash memory.
	Block Erase	Erases a specified area in the flash memory.
Blank check	Block Blank Check	Checks if a specified block in the flash memory has been correctly erased.
Write	Programming	Writes data to a specified area in the flash memory.
Getting information	Status	Gets the current operating status (status data).
	Silicon Signature	Gets 78K0/Kx2 information (such as the part number and flash memory configuration).
	Version Get	Gets the 78K0/Kx2 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 μPD78F0730 returns a response for the command issued by the dedicated flash memory programmer. The response names sent from the μPD78F0730 are listed below.

Table 19-7. Response Names

Response Name	Function
ACK	Acknowledges command/data.
NAK	Acknowledges illegal command/data.

19.8 Security Settings

The µPD78F0730 supports a security function that prohibits rewriting the user program written to the internal flash memory, so that the program cannot be changed by an unauthorized person.

The operations shown below can be performed using the Security Set command. The security setting is valid when the programming mode is set next.

- Disabling batch erase (chip erase)
  Execution of the block erase and batch erase (chip erase) commands for entire blocks in the flash memory is prohibited by this setting during on-board/off-board programming. Once execution of the batch erase (chip erase) command is prohibited, all of the prohibition settings (including prohibition of batch erase (chip erase)) can no longer be cancelled.

  Caution After the security setting for the batch erase is set, erasure cannot be performed for the device. In addition, even if a write command is executed, data different from that which has already been written to the flash memory cannot be written, because the erase command is disabled.

- Disabling block erase
  Execution of the block erase command for a specific block in the flash memory is prohibited during on-board/off-board programming. 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 boot cluster 0
  Execution of the batch erase (chip erase) command, block erase command, and write command on boot cluster 0 (0000H to 0FFFH) in the flash memory is prohibited by this setting.

  Caution If a security setting that rewrites boot cluster 0 has been applied, boot cluster 0 of that device will not be rewritten.

The batch erase (chip erase), block erase, write commands, and rewriting boot cluster 0 are enabled by the default setting when the flash memory is shipped. Security can be set by on-board/off-board programming and self programming. Each security setting can be used in combination.

Prohibition of erasing blocks and writing is cleared by executing the batch erase (chip erase) command.

Table 19-8 shows the relationship between the erase and write commands when the µPD78F0730 security function is enabled.

Table 19-8. Relationship Between Enabling Security Function and Command

(1) During on-board/off-board programming

Valid Security	Executed Command		
	Batch Erase (Chip Erase)	Block Erase	Write
Prohibition of batch erase (chip erase)	Cannot be erased in batch	Blocks cannot be erased.	Can be performed[Note].
Prohibition of block erase	Can be erased in batch.		Can be performed.
Prohibition of writing			Cannot be performed.
Prohibition of rewriting boot cluster 0	Cannot be erased in batch	Boot cluster 0 cannot be erased.	Boot cluster 0 cannot be written.

Note Confirm that no data has been written to the write area. Because data cannot be erased after batch erase (chip erase) is prohibited, do not write data if the data has not been erased.

(2) During self programming

Valid Security	Executed Command	
	Block Erase	Write
Prohibition of batch erase (chip erase)	Blocks can be erased.	Can be performed.
Prohibition of block erase		
Prohibition of writing		
Prohibition of rewriting boot cluster 0	Boot cluster 0 cannot be erased.	Boot cluster 0 cannot be written.

Table 19-9 shows how to perform security settings in each programming mode.

Table 19-9. Setting Security in Each Programming Mode

(1) On-board/off-board programming

Security	Security Setting	How to Disable Security Setting
Prohibition of batch erase (chip erase)	Set via GUI of dedicated flash programmer, etc.	Cannot be disabled after set.
Prohibition of block erase		Execute batch erase (chip erase) command
Prohibition of writing		
Prohibition of rewriting boot cluster 0		Cannot be disabled after set.

(2) Self programming

Security	Security Setting	How to Disable Security Setting
Prohibition of batch erase (chip erase)	Set by using information library.	Cannot be disabled after set.
Prohibition of block erase		Execute batch erase (chip erase) command during on-board/off-board programming (cannot be disabled during self programming)
Prohibition of writing		
Prohibition of rewriting boot cluster 0		Cannot be disabled after set.

19.9 Flash Memory Programming by Self-Programming

The µPD78F0730 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 the µPD78F0730 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 the 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.

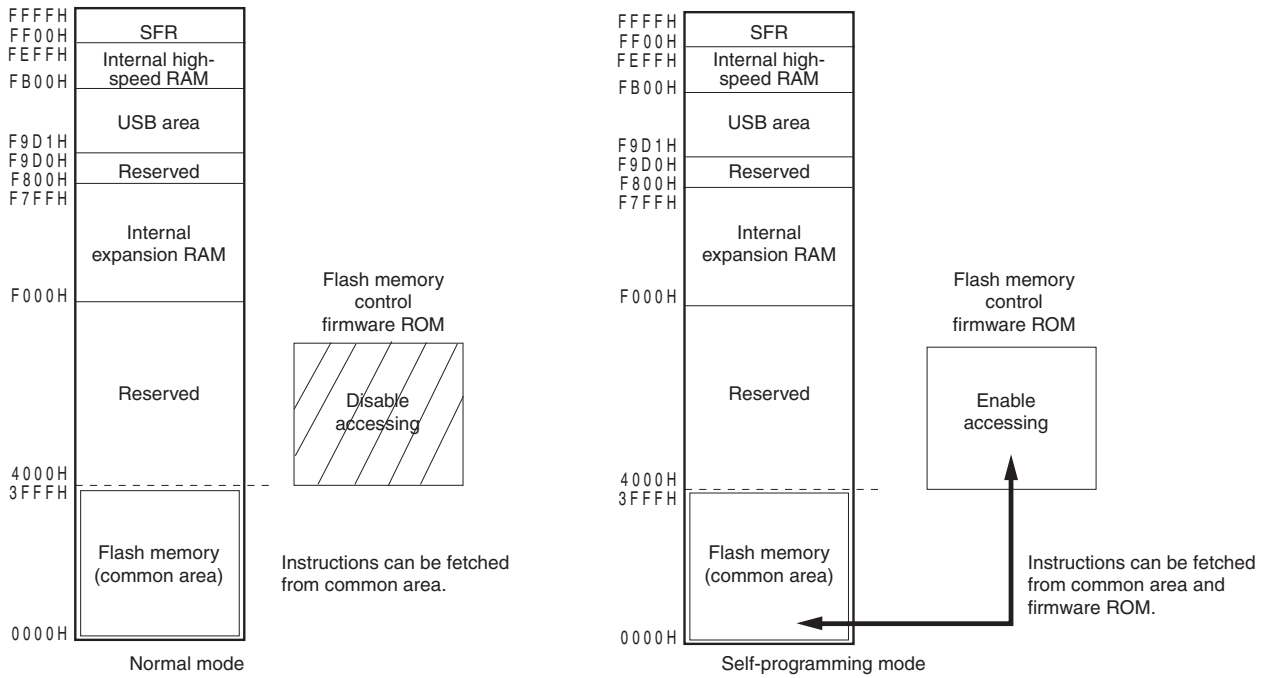
2. Be sure to execute the DI instruction before starting self-programming.
The self-programming function checks the interrupt request flags (IF0L, IF0H, IF1L, and IF1H). If an interrupt request is generated, self-programming is stopped.

3. Self-programming is also stopped by an interrupt request that is not masked even in the DI status. To prevent this, mask the interrupt by using the interrupt mask flag registers (MK0L, MK0H, MK1L, and MK1H).

4. Self-programming is executed with the internal high-speed oscillation clock. If the CPU operates with the X1 clock or external main system clock, the oscillation stabilization wait time of the internal high-speed oscillation clock elapses during self-programming.

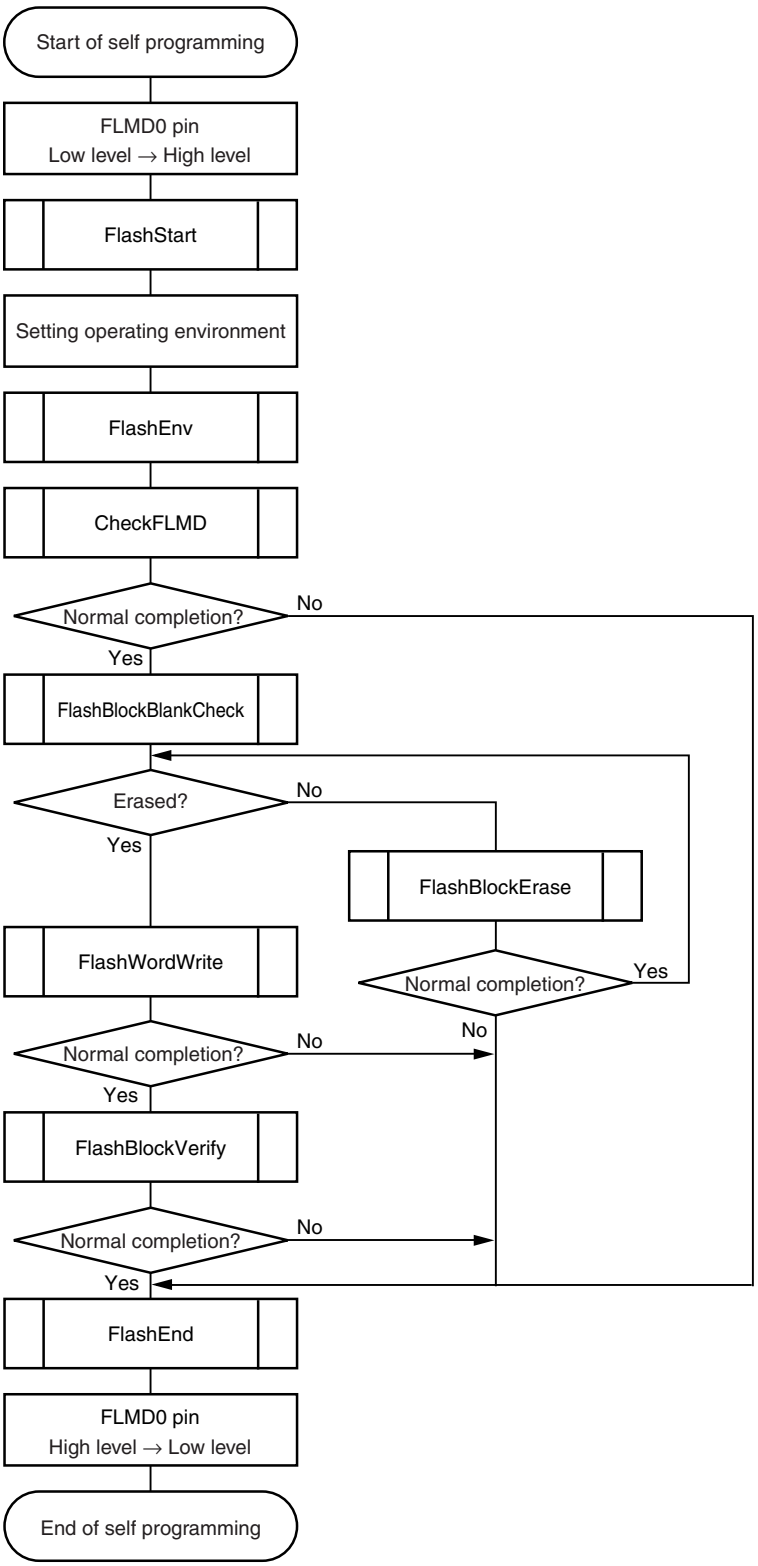
5. Allocate the entry program for self-programming in the common area of 0000H to 3FFFH.

Figure 19-15. Operation Mode and Memory Map for Self-Programming



The following figure illustrates a flow of rewriting the flash memory by using a self programming sample library.

Figure 19-16. Flow of Self Programming (Rewriting Flash Memory)



Remark For details of the self-programming function and the self-programming library, refer to **78K0 Microcontroller Self Programming Library Type01 User's Manual (U18274E)**.

19.9.1 Boot swap function

If rewriting the boot area has failed during self-programming due to a power failure or some other cause, the data in the boot area may be lost and the program may not be restarted by resetting.

The boot swap function is used to avoid this problem.

Before erasing boot cluster 0[Note], which is a boot program area, by self-programming, write a new boot program to boot cluster 1 in advance. When the program has been correctly written to boot cluster 1, swap this boot cluster 1 and boot cluster 0 by using the set information function of the firmware of the μPD78F0730, so that boot cluster 1 is used as a boot area. After that, erase or write the original boot program area, boot cluster 0.

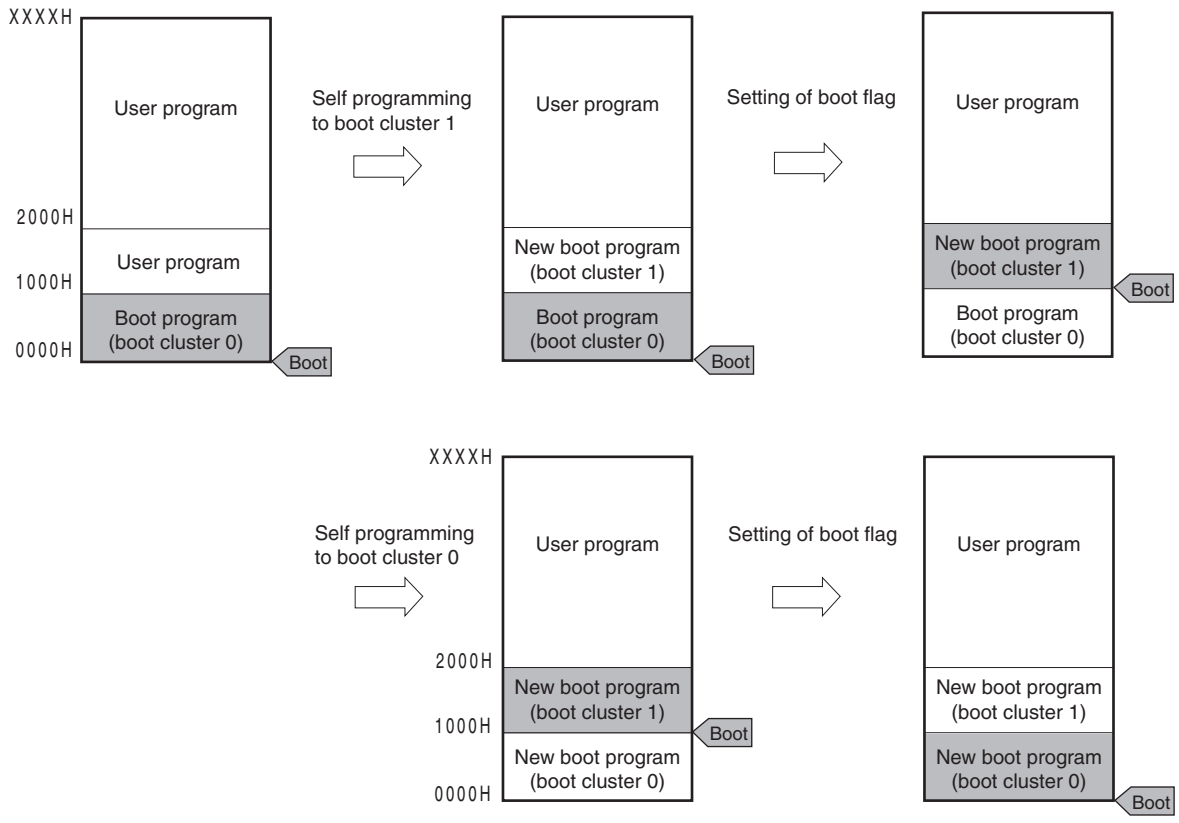
As a result, even if a power failure occurs while the boot programming area is being rewritten, the program is executed correctly because it is booted from boot cluster 1 to be swapped when the program is reset and started next.

If the program has been correctly written to boot cluster 0, restore the original boot area by using the set information function of the firmware of the μPD78F0730.

Note A boot cluster is a 4 KB area and boot clusters 0 and 1 are swapped by the boot swap function.

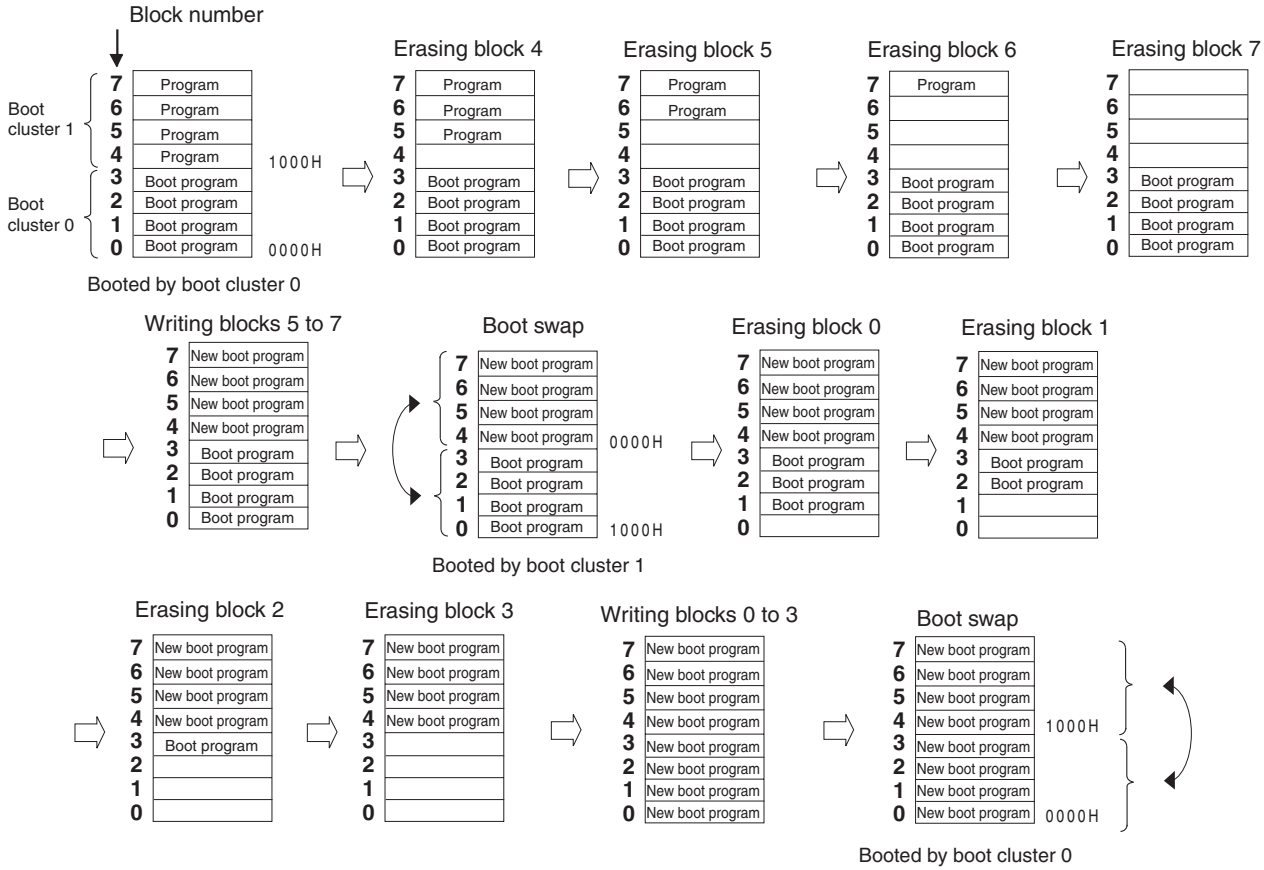
Boot cluster 0 (0000H to 0FFFH): Original boot program area
Boot cluster 1 (1000H to 1FFFH): Area subject to boot swap function

Figure 19-17. Boot Swap Function



Remark Boot cluster 1 becomes 0000H to 0FFFH when a reset is generated after the boot flag has been set.

Figure 19-18. Example of Executing Boot Swapping



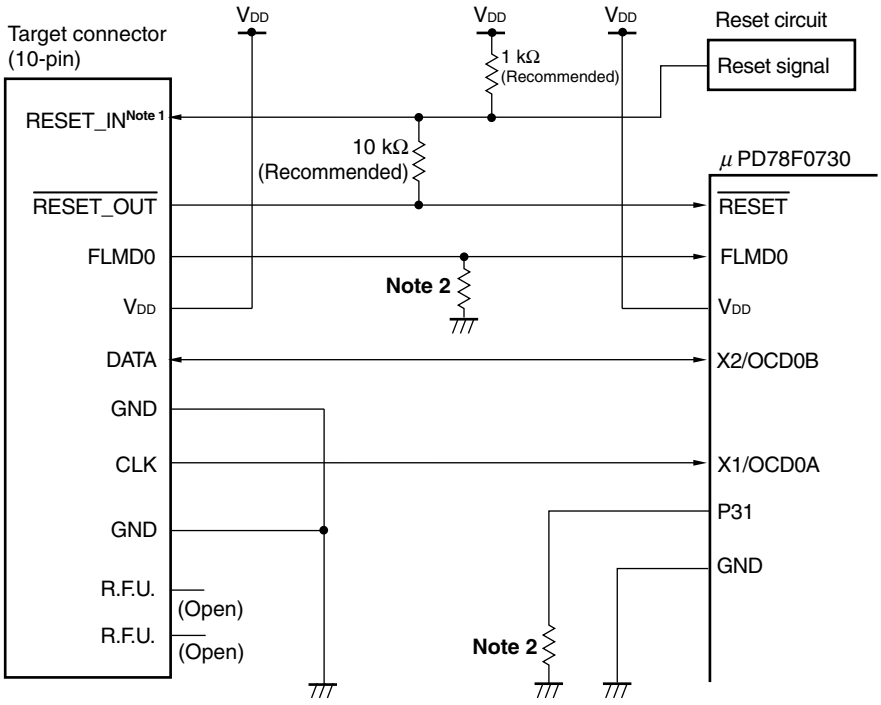
CHAPTER 20 ON-CHIP DEBUG FUNCTION

20.1 Connecting QB-MINI2 to µPD78F0730

The µPD78F0730 uses the V_{DD}, FLMD0, $\overline{\text{RESET}}$, OCD0A/X1 (or OCD1A/P31), OCD0B/X2 (or OCD1B/P32), and V_{SS} pins to communicate with the host machine via an on-chip debug emulator (QB-MINI2). Whether OCD0A/X1 and OCD1A/P31, or OCD0B/X2 and OCD1B/P32 are used can be selected.

Caution The µPD78F0730 has an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.

Figure 20-1. Connection Example of QB-MINI2 and µPD78F0730 (When OCD0A/X1 and OCD0B/X2 Are Used)



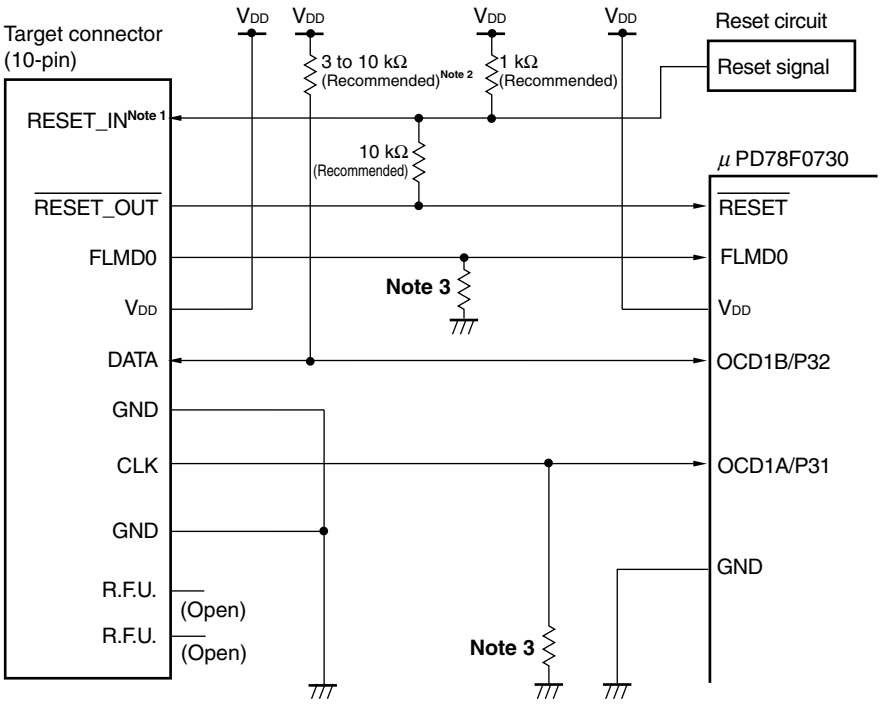
Notes 1. This connection is designed assuming that the reset signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less). For details, refer to **QB-MINI2 User's Manual (U18371E)**.

2. Make pull-down resistor 470 Ω or more (10 kΩ: recommended).

Cautions 1. Input the clock from the OCD0A/X1 pin during on-chip debugging.

2. Control the OCD0A/X1 and OCD0B/X2 pins by externally pulling down the OCD1A/P31 pin.

Figure 20-2. Connection Example of QB-MINI2 and µPD78F0730 (When OCD1A and OCD1B Are Used)



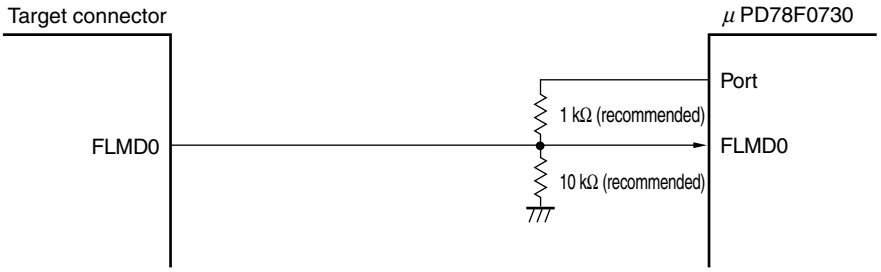
Notes 1. This connection is designed assuming that the reset signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less). For details, refer to **QB-MINI2 User's Manual (U18371E)**.

2. This is the processing of the pin when OCD1B/P32 is set as the input port (to prevent the pin from being left opened when not connected to QB-MINI2).

3. Make pull-down resistor 470 Ω or more (10 kΩ: recommended).

Connect the FLMD0 pin as follows when performing self programming by means of on-chip debugging.

Figure 20-3. Connection of FLMD0 Pin for Self Programming by Means of On-Chip Debugging



Caution When using the port that controls the FLMD0 pin, make sure that it satisfies the values of the high-level output current and FLMD0 supply voltage (minimum value: 0.8VDD) stated in CHAPTER 22 ELECTRICAL SPECIFICATIONS.

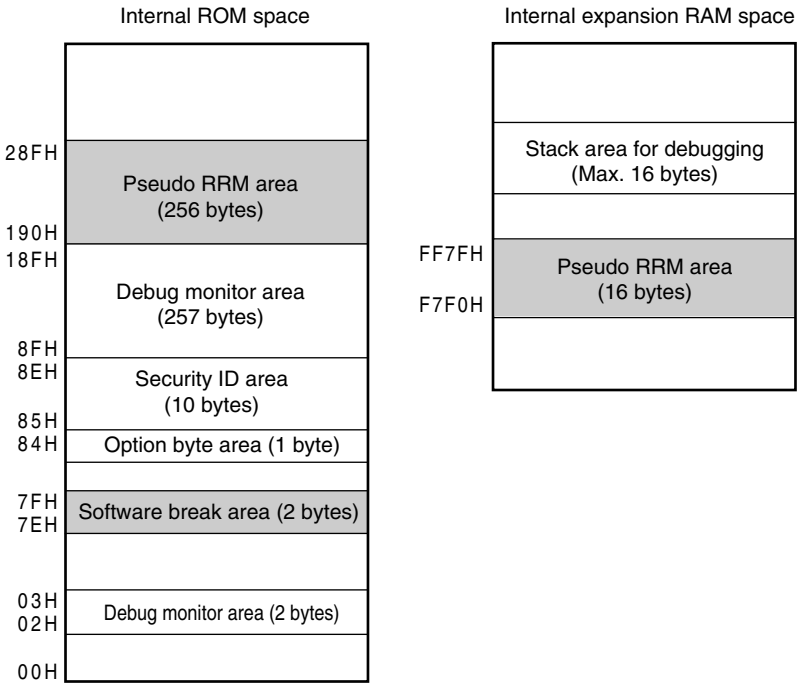
20.2 Reserved Area Used by QB-MINI2

QB-MINI2 uses the reserved areas shown in Figure 20-4 below to implement communication with the μPD78F0730, 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.

When using a boot swap operation during self programming, set the same value to boot cluster 1 beforehand.

For details on reserved area, refer to **QB-MINI2 User's Manual (U18371E)**.

Figure 20-4. 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 21 INSTRUCTION SET

This chapter lists each instruction set of the µPD78F0730 in table form. For details of each operation and operation code, refer to the separate document **78K/0 Series Instructions User's Manual (U12326E)**.

21.1 Conventions Used in Operation List

21.1.1 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 21-1. 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[Note]
sfrp	Special function register symbol (16-bit manipulatable register even addresses only)[Note]
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 3-7 Special Function Register List**.

21.1.2 Description of operation column

A: A register; 8-bit accumulator
X: X register
B: B register
C: C register
D: D register
E: E register
H: H register
L: L register
AX: AX register pair; 16-bit accumulator
BC: BC register pair
DE: DE register pair
HL: HL register pair
PC: Program counter
SP: Stack pointer
PSW: Program status word
CY: Carry flag
AC: Auxiliary carry flag
Z: Zero flag
RBS: Register bank select flag
IE: Interrupt request enable flag
( ): Memory contents indicated by address or register contents in parentheses
X_H, X_L: Higher 8 bits and lower 8 bits of 16-bit register
∧: Logical product (AND)
∨: Logical sum (OR)
∨: Exclusive logical sum (exclusive OR)
$\overline{\ \ }$: Inverted data
addr16: 16-bit immediate data or label
jdisp8: Signed 8-bit data (displacement value)

21.1.3 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

21.2 Operation List

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit data transfer	**MOV**	r, #byte	2	4	–	r ← byte	
		saddr, #byte	3	6	7	(saddr) ← byte	
		sfr, #byte	3	–	7	sfr ← byte	
		A, r Note 3	1	2	–	A ← r	
		r, A Note 3	1	2	–	r ← A	
		A, saddr	2	4	5	A ← (saddr)	
		saddr, A	2	4	5	(saddr) ← A	
		A, sfr	2	–	5	A ← sfr	
		sfr, A	2	–	5	sfr ← A	
		A, !addr16	3	8	9	A ← (addr16)	
		!addr16, A	3	8	9	(addr16) ← A	
		PSW, #byte	3	–	7	PSW ← byte	× × ×
		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	
	XCH	A, r Note 3	1	2	–	A ↔ r	
		A, saddr	2	4	6	A ↔ (saddr)	
		A, sfr	2	–	6	A ↔ (sfr)	
		A, !addr16	3	8	10	A ↔ (addr16)	
		A, [DE]	1	4	6	A ↔ (DE)	
		A, [HL]	1	4	6	A ↔ (HL)	
		A, [HL + byte]	2	8	10	A ↔ (HL + byte)	
		A, [HL + B]	2	8	10	A ↔ (HL + B)	
		A, [HL + C]	2	8	10	A ↔ (HL + C)	

Notes 1. When the internal high-speed RAM area is accessed or for an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except “r = A”

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
16-bit data transfer	**MOVW**	rp, #word	3	6	–	rp ← word	
		saddrp, #word	4	8	10	(saddrp) ← word	
		sfrp, #word	4	–	10	sfrp ← word	
		AX, saddrp	2	6	8	AX ← (saddrp)	
		saddrp, AX	2	6	8	(saddrp) ← AX	
		AX, sfrp	2	–	8	AX ← sfrp	
		sfrp, AX	2	–	8	sfrp ← AX	
		AX, rp Note 3	1	4	–	AX ← rp	
		rp, AX Note 3	1	4	–	rp ← AX	
		AX, !addr16	3	10	12	AX ← (addr16)	
		!addr16, AX	3	10	12	(addr16) ← AX	
	XCHW	AX, rp Note 3	1	4	–	AX ↔ rp	
8-bit operation	**ADD**	A, #byte	2	4	–	A, CY ← A + byte	× × ×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte	× × ×
		A, r Note 4	2	4	–	A, CY ← A + r	× × ×
		r, A	2	4	–	r, CY ← r + A	× × ×
		A, saddr	2	4	5	A, CY ← A + (saddr)	× × ×
		A, !addr16	3	8	9	A, CY ← A + (addr16)	× × ×
		A, [HL]	1	4	5	A, CY ← A + (HL)	× × ×
		A, [HL + byte]	2	8	9	A, CY ← A + (HL + byte)	× × ×
		A, [HL + B]	2	8	9	A, CY ← A + (HL + B)	× × ×
		A, [HL + C]	2	8	9	A, CY ← A + (HL + C)	× × ×
	ADDC	A, #byte	2	4	–	A, CY ← A + byte + CY	× × ×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte + CY	× × ×
		A, r Note 4	2	4	–	A, CY ← A + r + CY	× × ×
		r, A	2	4	–	r, CY ← r + A + CY	× × ×
		A, saddr	2	4	5	A, CY ← A + (saddr) + CY	× × ×
		A, !addr16	3	8	9	A, CY ← A + (addr16) + C	× × ×
		A, [HL]	1	4	5	A, CY ← A + (HL) + CY	× × ×
		A, [HL + byte]	2	8	9	A, CY ← A + (HL + byte) + CY	× × ×
		A, [HL + B]	2	8	9	A, CY ← A + (HL + B) + CY	× × ×
		A, [HL + C]	2	8	9	A, CY ← A + (HL + C) + CY	× × ×

Notes 1. When the internal high-speed RAM area is accessed or for an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Only when rp = BC, DE or HL

4. Except “r = A”

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	**SUB**	A, #byte	2	4	–	A, CY ← A – byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte	×	×	×
		A, r Note 3	2	4	–	A, CY ← A – r	×	×	×
		r, A	2	4	–	r, CY ← r – A	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr)	×	×	×
		A, !addr16	3	8	9	A, CY ← A – (addr16)	×	×	×
		A, [HL]	1	4	5	A, CY ← A – (HL)	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A – (HL + byte)	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A – (HL + B)	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A – (HL + C)	×	×	×
	SUBC	A, #byte	2	4	–	A, CY ← A – byte – CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte – CY	×	×	×
		A, r Note 3	2	4	–	A, CY ← A – r – CY	×	×	×
		r, A	2	4	–	r, CY ← r – A – CY	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr) – CY	×	×	×
		A, !addr16	3	8	9	A, CY ← A – (addr16) – CY	×	×	×
		A, [HL]	1	4	5	A, CY ← A – (HL) – CY	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A – (HL + byte) – CY	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A – (HL + B) – CY	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A – (HL + C) – CY	×	×	×
	AND	A, #byte	2	4	–	A ← A ∧ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∧ byte	×		
		A, r Note 3	2	4	–	A ← A ∧ r	×		
		r, A	2	4	–	r ← r ∧ A	×		
		A, saddr	2	4	5	A ← A ∧ (saddr)	×		
		A, !addr16	3	8	9	A ← A ∧ (addr16)	×		
		A, [HL]	1	4	5	A ← A ∧ (HL)	×		
		A, [HL + byte]	2	8	9	A ← A ∧ (HL + byte)	×		
		A, [HL + B]	2	8	9	A ← A ∧ (HL + B)	×		
		A, [HL + C]	2	8	9	A ← A ∧ (HL + C)	×		

Notes 1. When the internal high-speed RAM area is accessed or for an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except “r = A”

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit operation	**OR**	A, #byte	2	4	–	A ← A ∨ byte	×
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∨ byte	×
		A, r Note 3	2	4	–	A ← A ∨ r	×
		r, A	2	4	–	r ← r ∨ A	×
		A, saddr	2	4	5	A ← A ∨ (saddr)	×
		A, !addr16	3	8	9	A ← A ∨ (addr16)	×
		A, [HL]	1	4	5	A ← A ∨ (HL)	×
		A, [HL + byte]	2	8	9	A ← A ∨ (HL + byte)	×
		A, [HL + B]	2	8	9	A ← A ∨ (HL + B)	×
		A, [HL + C]	2	8	9	A ← A ∨ (HL + C)	×
	XOR	A, #byte	2	4	–	A ← A ⊻ byte	×
		saddr, #byte	3	6	8	(saddr) ← (saddr) ⊻ byte	×
		A, r Note 3	2	4	–	A ← A ⊻ r	×
		r, A	2	4	–	r ← r ⊻ A	×
		A, saddr	2	4	5	A ← A ⊻ (saddr)	×
		A, !addr16	3	8	9	A ← A ⊻ (addr16)	×
		A, [HL]	1	4	5	A ← A ⊻ (HL)	×
		A, [HL + byte]	2	8	9	A ← A ⊻ (HL + byte)	×
		A, [HL + B]	2	8	9	A ← A ⊻ (HL + B)	×
		A, [HL + C]	2	8	9	A ← A ⊻ (HL + C)	×
	CMP	A, #byte	2	4	–	A – byte	× × ×
		saddr, #byte	3	6	8	(saddr) – byte	× × ×
		A, r Note 3	2	4	–	A – r	× × ×
		r, A	2	4	–	r – A	× × ×
		A, saddr	2	4	5	A – (saddr)	× × ×
		A, !addr16	3	8	9	A – (addr16)	× × ×
		A, [HL]	1	4	5	A – (HL)	× × ×
		A, [HL + byte]	2	8	9	A – (HL + byte)	× × ×
		A, [HL + B]	2	8	9	A – (HL + B)	× × ×
		A, [HL + C]	2	8	9	A – (HL + C)	× × ×

Notes 1. When the internal high-speed RAM area is accessed or for an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except “r = A”

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit operation	**ADDW**	AX, #word	3	6	–	AX, CY ← AX + word	×	×	×
	SUBW	AX, #word	3	6	–	AX, CY ← AX – word	×	×	×
	CMPW	AX, #word	3	6	–	AX – word	×	×	×
Multiply/ divide	**MULU**	X	2	16	–	AX ← A × X			
	DIVUW	C	2	25	–	AX (Quotient), C (Remainder) ← AX ÷ C			
Increment/ decrement	**INC**	r	1	2	–	r ← r + 1	×	×	
		saddr	2	4	6	(saddr) ← (saddr) + 1	×	×	
	DEC	r	1	2	–	r ← r – 1	×	×	
		saddr	2	4	6	(saddr) ← (saddr) – 1	×	×	
	INCW	rp	1	4	–	rp ← rp + 1			
	DECW	rp	1	4	–	rp ← rp – 1			
Rotate	**ROR**	A, 1	1	2	–	(CY, A_7 ← A_0, A_{m-1} ← A_m) × 1 time			×
	ROL	A, 1	1	2	–	(CY, A_0 ← A_7, A_{m+1} ← A_m) × 1 time			×
	RORC	A, 1	1	2	–	(CY ← A_0, A_7 ← CY, A_{m-1} ← A_m) × 1 time			×
	ROLC	A, 1	1	2	–	(CY ← A_7, A_0 ← CY, A_{m+1} ← A_m) × 1 time			×
	ROR4	[HL]	2	10	12	A_{3-0} ← $(HL)_{3-0}$, $(HL)_{7-4}$ ← A_{3-0}, $(HL)_{3-0}$ ← $(HL)_{7-4}$			
	ROL4	[HL]	2	10	12	A_{3-0} ← $(HL)_{7-4}$, $(HL)_{3-0}$ ← A_{3-0}, $(HL)_{7-4}$ ← $(HL)_{3-0}$			
BCD adjustment	**ADJBA**		2	4	–	Decimal Adjust Accumulator after Addition	×	×	×
	ADJBS		2	4	–	Decimal Adjust Accumulator after Subtract	×	×	×
Bit manipulate	**MOV1**	CY, saddr.bit	3	6	7	CY ← (saddr.bit)			×
		CY, sfr.bit	3	–	7	CY ← sfr.bit			×
		CY, A.bit	2	4	–	CY ← A.bit			×
		CY, PSW.bit	3	–	7	CY ← PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← (HL).bit			×
		saddr.bit, CY	3	6	8	(saddr.bit) ← CY			
		sfr.bit, CY	3	–	8	sfr.bit ← CY			
		A.bit, CY	2	4	–	A.bit ← CY			
		PSW.bit, CY	3	–	8	PSW.bit ← CY	×	×	
		[HL].bit, CY	2	6	8	(HL).bit ← CY			

Notes 1. When the internal high-speed RAM area is accessed or for an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
Bit manipulate	**AND1**	CY, saddr.bit	3	6	7	CY ← CY ∧ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ∧ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ∧ A.bit	×
		CY, PSW.bit	3	–	7	CY ← CY ∧ PSW.bit	×
		CY, [HL].bit	2	6	7	CY ← CY ∧ (HL).bit	×
	OR1	CY, saddr.bit	3	6	7	CY ← CY ∨ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ∨ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ∨ A.bit	×
		CY, PSW.bit	3	–	7	CY ← CY ∨ PSW.bit	×
		CY, [HL].bit	2	6	7	CY ← CY ∨ (HL).bit	×
	XOR1	CY, saddr.bit	3	6	7	CY ← CY ⊻ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ⊻ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ⊻ A.bit	×
		CY, PSW. bit	3	–	7	CY ← CY ⊻ PSW.bit	×
		CY, [HL].bit	2	6	7	CY ← CY ⊻ (HL).bit	×
	SET1	saddr.bit	2	4	6	(saddr.bit) ← 1	
		sfr.bit	3	–	8	sfr.bit ← 1	
		A.bit	2	4	–	A.bit ← 1	
		PSW.bit	2	–	6	PSW.bit ← 1	× × ×
		[HL].bit	2	6	8	(HL).bit ← 1	
	CLR1	saddr.bit	2	4	6	(saddr.bit) ← 0	
		sfr.bit	3	–	8	sfr.bit ← 0	
		A.bit	2	4	–	A.bit ← 0	
		PSW.bit	2	–	6	PSW.bit ← 0	× × ×
		[HL].bit	2	6	8	(HL).bit ← 0	
	SET1	CY	1	2	–	CY ← 1	1
	CLR1	CY	1	2	–	CY ← 0	0
	NOT1	CY	1	2	–	CY ← $\overline{CY}$	×

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_{CPU}) selected by the processor clock control register (PCC).

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
Call/return	**CALL**	!addr16	3	7	–	(SP – 1) ← (PC + 3)H, (SP – 2) ← (PC + 3)L, PC ← addr16, SP ← SP – 2	
	CALLF	!addr11	2	5	–	(SP – 1) ← (PC + 2)H, (SP – 2) ← (PC + 2)L, PC15 – 11 ← 00001, PC10 – 0 ← addr11, SP ← SP – 2	
	CALLT	[addr5]	1	6	–	(SP – 1) ← (PC + 1)H, (SP – 2) ← (PC + 1)L, PCH ← (addr5 + 1), PCL ← (addr5), SP ← SP – 2	
	BRK		1	6	–	(SP – 1) ← PSW, (SP – 2) ← (PC + 1)H, (SP – 3) ← (PC + 1)L, PCH ← (003FH), PCL ← (003EH), SP ← SP – 3, IE ← 0	
	RET		1	6	–	PCH ← (SP + 1), PCL ← (SP), SP ← SP + 2	
	RETI		1	6	–	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3	R R R
	RETB		1	6	–	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3	R R R
Stack manipulate	**PUSH**	PSW	1	2	–	(SP – 1) ← PSW, SP ← SP – 1	
		rp	1	4	–	(SP – 1) ← rpH, (SP – 2) ← rpL, SP ← SP – 2	
	POP	PSW	1	2	–	PSW ← (SP), SP ← SP + 1	R R R
		rp	1	4	–	rpH ← (SP + 1), rpL ← (SP), SP ← SP + 2	
	MOVW	SP, #word	4	–	10	SP ← word	
		SP, AX	2	–	8	SP ← AX	
		AX, SP	2	–	8	AX ← SP	
Unconditional branch	**BR**	!addr16	3	6	–	PC ← addr16	
		$addr16	2	6	–	PC ← PC + 2 + jdisp8	
		AX	2	8	–	PCH ← A, PCL ← X	
Conditional branch	**BC**	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if CY = 1	
	BNC	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if CY = 0	
	BZ	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if Z = 1	
	BNZ	$addr16	2	6	–	PC ← PC + 2 + 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 (fCPU) selected by the processor clock control register (PCC).

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
Conditional branch	**BT**	saddr.bit, $addr16	3	8	9	PC ← PC + 3 + jdisp8 if (saddr.bit) = 1	
		sfr.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if sfr.bit = 1	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 1	
		PSW.bit, $addr16	3	–	9	PC ← PC + 3 + jdisp8 if PSW.bit = 1	
		[HL].bit, $addr16	3	10	11	PC ← PC + 3 + jdisp8 if (HL).bit = 1	
	BF	saddr.bit, $addr16	4	10	11	PC ← PC + 4 + jdisp8 if (saddr.bit) = 0	
		sfr.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if sfr.bit = 0	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 0	
		PSW.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if PSW. bit = 0	
		[HL].bit, $addr16	3	10	11	PC ← PC + 3 + jdisp8 if (HL).bit = 0	
	BTCLR	saddr.bit, $addr16	4	10	12	PC ← PC + 4 + jdisp8 if (saddr.bit) = 1 then reset (saddr.bit)	
		sfr.bit, $addr16	4	–	12	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit	
		PSW.bit, $addr16	4	–	12	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	× × ×
		[HL].bit, $addr16	3	10	12	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit	
	DBNZ	B, $addr16	2	6	–	B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0	
		C, $addr16	2	6	–	C ← C –1, then PC ← PC + 2 + jdisp8 if C ≠ 0	
		saddr, $addr16	3	8	10	(saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if (saddr) ≠ 0	
CPU control	**SEL**	RBn	2	4	–	RBS1, 0 ← n	
	NOP		1	2	–	No Operation	
	EI		2	–	6	IE ← 1 (Enable Interrupt)	
	DI		2	–	6	IE ← 0 (Disable Interrupt)	
	HALT		2	6	–	Set HALT Mode	
	STOP		2	6	–	Set STOP Mode	

Notes 1. When the internal high-speed RAM area is accessed or for an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

2. This clock cycle applies to the internal ROM program.

21.3 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

First Operand \ Second Operand	#byte	A	r[Note]	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL + byte] [HL + B] [HL + C]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH 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[Note]	sfrp	saddrp	!addr16	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW[Note]						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
SP	MOVW	MOVW						

Note Only when rp = BC, DE, HL

(3) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

Second Operand / First Operand	A.bit	sfr.bit	saddr.bit	PSW.bit	[HL].bit	CY	$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call instructions/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

Second Operand / First Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruction					BT BF BTCLR DBNZ

(5) Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

CHAPTER 22 ELECTRICAL SPECIFICATIONS

Caution The µPD78F0730 has an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.

Absolute Maximum Ratings (T_A = 25°C)

Parameter	Symbol	Conditions		Ratings	Unit
Supply voltage	V_{DD}			−0.5 to +6.5	V
	EV_{DD}			−0.5 to +6.5	V
	V_{SS}			−0.5 to +0.3	V
	EV_{SS}			−0.5 to +0.3	V
Input voltage	V_{I1}	P00, P01, P10 to P17, P30 to P33, P120 to P122, X1, X2, $\overline{\text{RESET}}$, EXCLK, FLMD0		−0.3 to V_{DD} + 0.3 Note1	V
	V_{I2}	P60, P61 (N-ch open drain)		−0.3 to +6.5	V
	V_{I3}	USBP, USBM		−0.5 to +3.8	
	V_{I4}	REGC		−0.5 to + 3.6 Note2	
	V_{I5}	USBREGC		−0.5 to + 3.8 Note2	
Output voltage	V_{O1}	Other than USBP, USBM, USBPUC		−0.3 to V_{DD} + 0.3 Note1	V
	V_{O2}	USBP, USBM, USBPUC		−0.5 to +3.8	V
Output current, high	I_{OH1}	Per pin		−10	mA
		Total of all pins −45 mA	P00, P01, P10 to P17, P30, P33, P120	−25	mA
			P31, P32	−20	mA
	I_{OH2}	Per pin	P121, P122	−1	mA
		Total of all pins	P121, P122	−4	mA
Output current, low	I_{OL1}	Per pin		30	mA
		Total of all pins 160 mA	P00, P01, P10 to P17, P30, P33, P120	60	mA
			P31, P32, P60, P61	100	mA
	I_{OL2}	Per pin	P121, P122	4	mA
		Total of all pins	P121, P122	10	mA
Operating ambient temperature	T_A	In normal operation mode		−40 to +85	°C
		In flash memory programming mode			
Storage temperature	T_{stg}			−65 to +150	°C

Notes 1. Must be 6.5 V or lower.

2. Must be V_{DD} or lower.

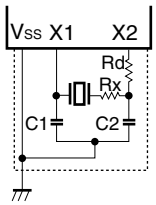
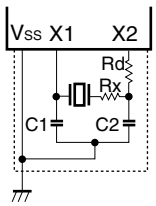
Caution Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge

of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

X1 Oscillator Characteristics

(T_A = –40 to +85°C, 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = 0 V)

Resonator	Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Crystal resonator		X1 clock oscillation frequency (f_X)Note	4.0 V ≤ V_{DD} ≤ 5.5 V			16.0	MHz
Ceramic resonator		X1 clock oscillation frequency (f_X)Note	4.0 V ≤ V_{DD} ≤ 5.5 V			16.0	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.
In addition, make sure that V_{DD} ≥ 4.0 V before switching the CPU clock to the X1 clock.

Internal Oscillator Characteristics

(T_A = -40 to +85°C, 2.7 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = 0 V)

Resonator	Parameter	Conditions		MIN.	TYP.	MAX.	Unit
16 MHz internal oscillator	Internal high-speed oscillation clock frequency (f_{RH})[Note 1]	RSTS = 1	V_{DD} ≥ 2.7 V	14.4	16.0	17.6	MHz
		RSTS = 0		2.48	5.6	9.86	MHz
240 kHz internal oscillator	Internal low-speed oscillation clock frequency (f_{RL})	2.7 V ≤ V_{DD} ≤ 5.5 V		216	240	264	kHz

Note Indicates only oscillator characteristics. Refer to **AC Characteristics** for instruction execution time.

Remark RSTS : Bit 7 of the internal oscillation mode register (RCM)).

Recommended Oscillator Constants

(1) X1 oscillation: Crystal resonator (AMPH =1, RMC = 00H, T_A = –40 to +85°C)

Manufacturer	Part Number	Frequency (MHz)	Recommended Circuit Constants				Oscillation Voltage Range	
			C1 (pF)	C2 (pF)	Rd (Ω)	Rx (Ω)	MIN. (V)	MAX. (V)
KYOCERA KINSEKI Corporation	CX3225SB1200D0PPTZ1	12.0	12	12	220	270	4.0	5.5
	CX3225SB1600D0PPTZ1	16.0	12	12	220	270		

(2) X1 oscillation: Ceramic resonator (AMPH =1, RMC = 00H, T_A = –40 to +85°C)

Manufacturer	Part Number	Frequency (MHz)	Recommended Circuit Constants				Oscillation Voltage Range	
			C1 (pF)	C2 (pF)	Rd (Ω)	Rx (Ω)	MIN. (V)	MAX. (V)
Toyama Murata Mfg. Co., Ltd.	CSTCE12M0GH5L99-R0	12.0	Internal (33)	Internal (33)	0	0	4.0	5.5
	CSTCE16M0VH3L99-R0	16.0	Internal (15)	Internal (15)	0	0		

Caution The oscillator constants shown above are reference values based on evaluation in a specific environment by the resonator manufacturer. If it is necessary to optimize the oscillator characteristics in the actual application, apply to the resonator manufacturer for evaluation on the implementation circuit. The oscillation voltage and oscillation frequency only indicate the oscillator characteristic. Use the µPD78F0730 so that the internal operation conditions are within the specifications of the DC and AC characteristics.

DC Characteristics (1/3)

(T_A = –40 to +85°C, 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = 0 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Output current, high[Note 1]	I_{OH1}	Per pin for P00, P01, P10 to P17, P30 to P33, P120	4.0 V ≤ V_{DD} ≤ 5.5 V			–3.0	mA
		Total of P00, P01, P10 to P17, P30, P33, P120[Note 3]	4.0 V ≤ V_{DD} ≤ 5.5 V			–20.0	mA
		Total of P31 and P32[Note 3]	4.0 V ≤ V_{DD} ≤ 5.5 V			–6.0	mA
		Total[Note 3] of all pins	4.0 V ≤ V_{DD} ≤ 5.5 V			–26.0	mA
	I_{OH2}	Per pin for P121, P122	4.0 V ≤ V_{DD} ≤ 5.5 V			–100	μA
Output current, low[Note 2]	I_{OL1}	Per pin for P00, P01, P10 to P17, P30 to P33, P120	4.0 V ≤ V_{DD} ≤ 5.5 V			8.5	mA
		Per pin for P60, P61	4.0 V ≤ V_{DD} ≤ 5.5 V			15.0	mA
		Total of P00, P01, P10 to P17, P30, P33, P120[Note 3]	4.0 V ≤ V_{DD} ≤ 5.5 V			20.0	mA
		Total of P31, P32, P60, P61[Note 3]	4.0 V ≤ V_{DD} ≤ 5.5 V			45.0	mA
		Total of all pins[Note 3]	4.0 V ≤ V_{DD} ≤ 5.5 V			65.0	mA
	I_{OL2}	Per pin for P121, P122	4.0 V ≤ V_{DD} ≤ 5.5 V			400	μA

Notes 1. Value of current at which the device operation is guaranteed even if the current flows from V_{DD} to an output pin.

2. Value of current at which the device operation is guaranteed even if the current flows from an output pin to GND.

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_{OH} is n%: Total output current of pins = (I_{OH} × 0.7)/(n × 0.01)

<Example> Where the duty factor is 50%, I_{OH} = 20.0 mA

Total output current of pins = (20.0 × 0.7)/(50 × 0.01) = 28.0 mA

However, the current that is allowed to flow into one pin does not vary depending on the duty factor. A current higher than the absolute maximum rating must not flow into one pin.

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

DC Characteristics (2/3)

(T_A = –40 to +85°C, 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = 0 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Input voltage, high	V_{IH1}	P12, P13, P15, P16		0.7V_{DD}		V_{DD}	V
	V_{IH2}	P00, P01, P10, P11, P14, P17, P30 to P33, P120 to P122, $\overline{\text{RESET}}$, EXCLK		0.8V_{DD}		V_{DD}	V
	V_{IH3}	USBP, USBM		2.0		USBREGC[Note]	V
	V_{IH4}	P60, P61		0.7V_{DD}		6.0	V
Input voltage, low	V_{IL1}	P12, P13, P15, P16, P60, P61		0		0.3V_{DD}	V
	V_{IL2}	P00, P01, P10, P11, P14, P17, P30 to P33, P120 to P122, $\overline{\text{RESET}}$, EXCLK		0		0.2V_{DD}	V
	V_{IL3}	USBP, USBM		0		0.8	V
Output voltage, high	V_{OH1}	P00, P01, P10 to P17, P30 to P33, P120	4.0 V ≤ V_{DD} ≤ 5.5 V, I_{OH} = –3.0 mA	V_{DD} – 0.7			V
	V_{OH2}	P121, P122	I_{OH} = –100 µA	V_{DD} – 0.5			V
	V_{OH3}	USBP, USBM	R_L = 15 kΩ, V_{SS} connected	2.8		3.6	V
	V_{OH4}	USBPUC	I_{OH} = –100 µA	USBREGC[Note] – 0.5			V
Output voltage, low	V_{OL1}	P00, P01, P10 to P17, P30 to P33, P120	4.0 V ≤ V_{DD} ≤ 5.5 V, I_{OL} = 8.5 mA			0.7	V
	V_{OL2}	P121, P122	I_{OL} = 0.4 mA			0.4	V
	V_{OL3}	USBP, USBM	R_L = 1.5 kΩ, USBREGC connected			0.3	V
	V_{OL4}	USBPUC	I_{OL} = 1 mA			0.4	V
	V_{OL5}	P60, P61	4.0 V ≤ V_{DD} ≤ 5.5 V, I_{OL} = 15 mA			2.0	V
			4.0 V ≤ V_{DD} ≤ 5.5 V, I_{OL} = 5 mA			0.4	V

Note Voltage output to the USBREGC pin (3.3 V ±0.3 V).

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

DC Characteristics (3/3)

(T_A = –40 to +85°C, 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = 0 V)

Parameter	Symbol	Conditions			MIN.	TYP.	MAX.	Unit
Input leakage current, high	I_{LIH1}	P00, P01, P10 to P17, P30 to P33, P60, P61, P120	$V_I = V_{DD}$				1	μA
	I_{LIH2}	P121, 122 (X1, X2)	$V_I = V_{DD}$	I/O port mode			1	μA
				OSC mode			20	μA
	I_{LIH3}	USBP, USBM	V_I = USBRECG Note 1				10	μA
Input leakage current, low	I_{LIL1}	P00, P01, P10 to P17, P30 to P33, P60, P61, P120	$V_I = V_{SS}$				–1	μA
	I_{LIL2}	P121, 122 (X1, X2)	$V_I = V_{SS}$	I/O port mode			–1	μA
				OSC mode			–20	μA
	I_{LIH3}	USBP, USBM	$V_I = V_{SS}$				–10	μA
Pull-up resistor	R_U	$V_I = V_{SS}$			10	20	100	kΩ
FLMD0 supply voltage	V_{IL}	In normal operation mode			0		$0.2V_{DD}$	V
	V_{IH}	In self-programming mode			$0.8V_{DD}$		V_{DD}	V
Supply current Note 2	I_{DD1} Note 3, 4	Operating mode	f_{XP} = 16 MHz, V_{DD} = 5.0 V			17.5	33	mA
	I_{DD3} Note 5	STOP mode	V_{DD} = 5.0 V			2.6	38	μA

Notes 1. Voltage output to the USBREGC pin (3.3 V ±0.3 V).

2. Total current flowing into the internal power supply (V_{DD}, EV_{DD}), including the peripheral USB, input leakage current when input pin fixed as V_{DD} or V_{SS} and operation current (however, the current flowing into the pull-up resistor of the port is not included).

3. TYP value is current of the state that supplied operating clock to USB.
Not USB data communication. Not include current used in an USB buffer.

4. MAX value is current in USB data communication.
Includes current used in an USB buffer.

5. Current of internal 240 kHz oscillator circuit does not include.

Remarks 1. Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

2. f_{XP}: System clock frequency

AC Characteristics

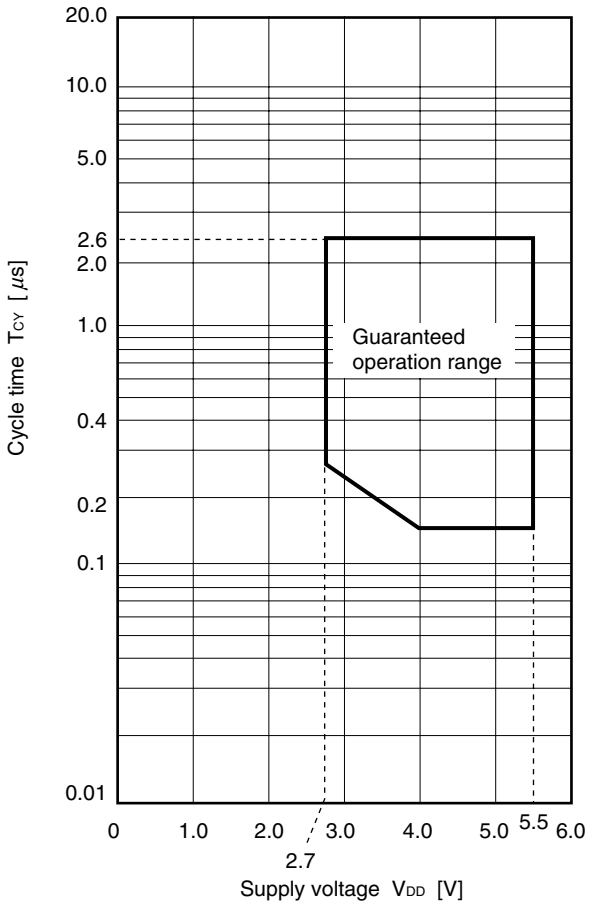
(1) Basic operation

(T_A = –40 to +85°C, 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = 0 V)

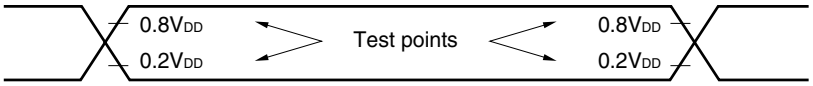
Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Instruction cycle (minimum instruction execution time)	T_{CY}	Main system clock (f_{XP}) operation	4.0 V ≤ V_{DD} ≤ 5.5 V	0.125		2.6	µs
			2.7 V ≤ V_{DD} < 4.0 V	0.25		2.6	µs
External main system clock frequency	f_{EXCLK}	4.0 V ≤ V_{DD} ≤ 5.5 V				16.0	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_{TIH0}, t_{TIL0}	4.0 V ≤ V_{DD} ≤ 5.5 V		$2/f_{sam}$ + 0.1 Note			µs
TI50, TI51 input frequency	f_{TI5}	4.0 V ≤ V_{DD} ≤ 5.5 V				10	MHz
TI50, TI51 input high-level width, low-level width	t_{TIH5}, t_{TIL5}	4.0 V ≤ V_{DD} ≤ 5.5 V		50			ns
Interrupt input high-level width, low-level width	t_{INTH}, t_{INTL}			1			µs
$\overline{\text{RESET}}$ low-level width	t_{RSL}			10			µs

Note f_{sam} : sampling clock.
Selection of f_{sam} = f_{PRS} or $f_{PRS}/4$ is possible using bits 0 and 1 (PRM000 and PRM001) of prescaler mode register 00 (PRM00). Note that when selecting the TI000 valid edge as the count clock, f_{sam} = f_{PRS}.

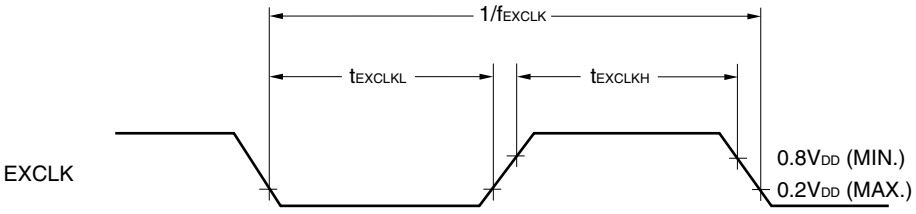
T_{CY} vs. V_{DD} (Main System Clock Operation)



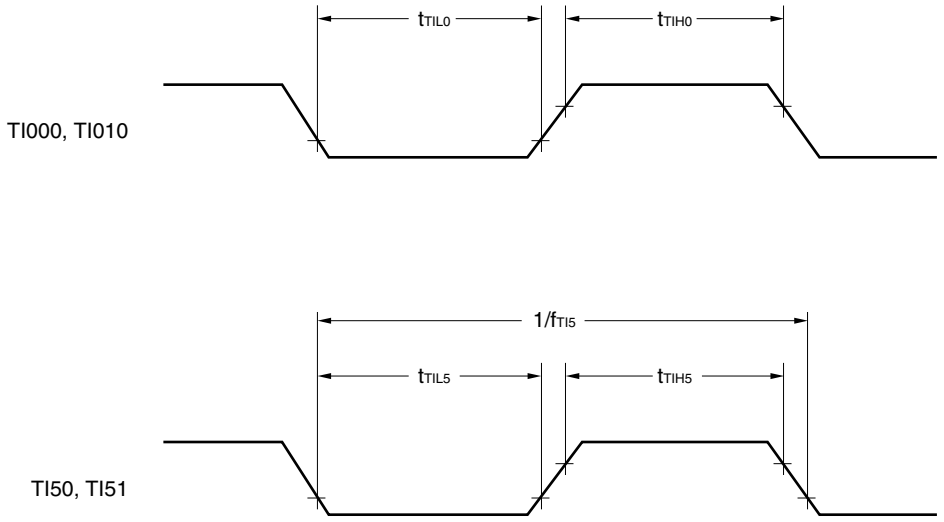
AC Timing Test Points (except Excluding External Main System Clock input)



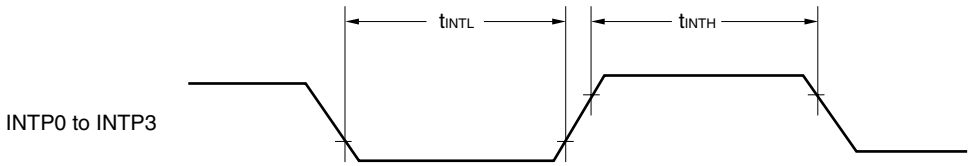
External Main System Clock Timing



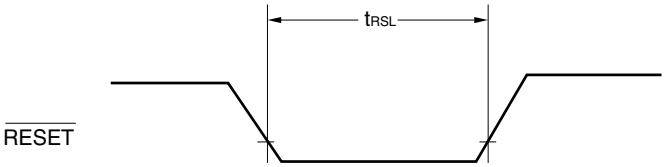
TI Timing



Interrupt Request Input Timing



$\overline{\text{RESET}}$ Input Timing



(2) Serial interface

(T_A = –40 to +85°C, 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = 0 V)

(a) UART6 (dedicated baud rate generator output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate					625	kbps

(b) CSI10 (master mode, $\overline{SCK10}$... internal clock output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK10}$ cycle time	t_{KCY1}	4.0 V ≤ V_{DD} ≤ 5.5 V	250			ns
$\overline{SCK10}$ high-/low-level width	t_{KH1}, t_{KL1}		$t_{KCY1}/2 - 20$ Note 1			ns
SI10 setup time (to $\overline{SCK10}$↑)	t_{SIK1}		70			ns
SI10 hold time (from $\overline{SCK10}$↑)	t_{KSI1}		30			ns
Delay time from $\overline{SCK10}$↓ to SO10 output	t_{KSO1}	C = 50 pF Note 2			40	ns

Notes 1. This value is when high-speed system clock (f_{XH}) is used.

2. C is the load capacitance of the $\overline{SCK10}$ and SO10 output lines.

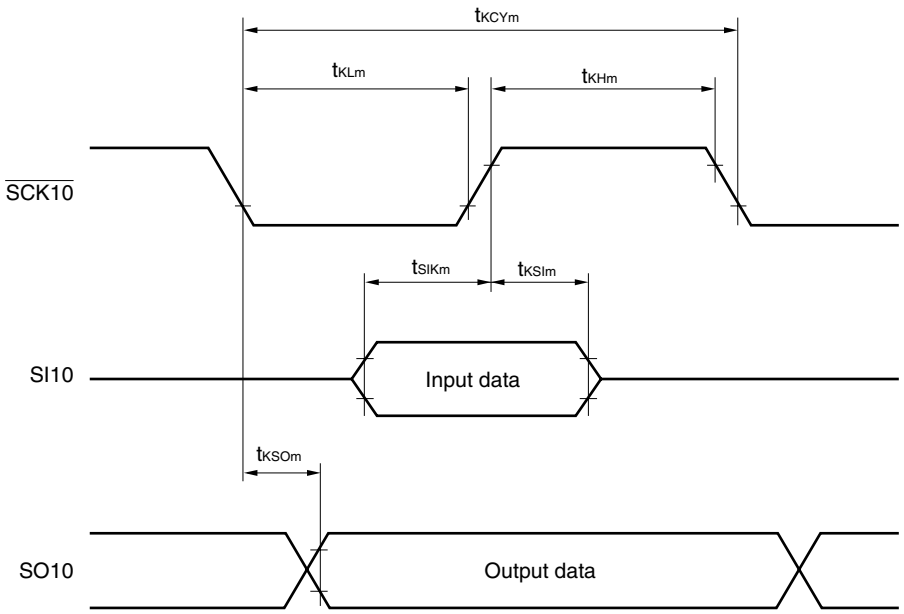
(c) CSI10 (slave mode, $\overline{SCK10}$... external clock input)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK10}$ cycle time	t_{KCY2}		400			ns
$\overline{SCK10}$ high-/low-level width	t_{KH2}, t_{KL2}		$t_{KCY2}/2$			ns
SI10 setup time (to $\overline{SCK10}$↑)	t_{SIK2}		80			ns
SI10 hold time (from $\overline{SCK10}$↑)	t_{KSI2}		50			ns
Delay time from $\overline{SCK10}$↓ to SO10 output	t_{KSO2}	C = 50 pF Note			120	ns

Note C is the load capacitance of the SO10 output line.

Serial Transfer Timing

CSI10:

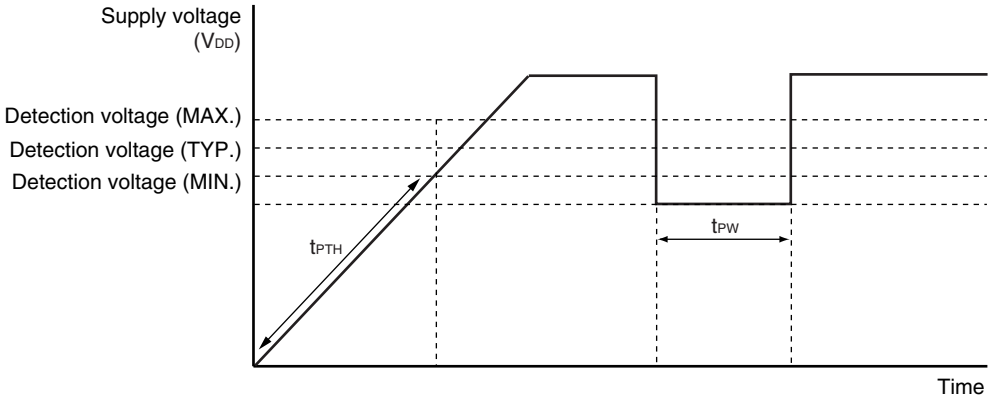


Remark m = 1, 2

1.59 V POC Circuit Characteristics (T_A = –40 to +85°C, V_{SS} = EV_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Detection voltage	V_{POC}		1.44	1.59	1.74	V
Power voltage rise inclination	t_{PTH}	V_{DD}: 0 V → change inclination of V_{POC}	0.75			V/ms
Minimum pulse width	t_{PW}	When the supply voltage (V_{DD}) drops	200			µs

POC Circuit Timing



2.7 V POC Circuit Characteristics (T_A = –40 to +85°C, V_{SS} = EV_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Detection voltage on application of supply voltage	V_{DDPOC}	POCMODE (option bye) = 1	2.50	2.70	2.90	V

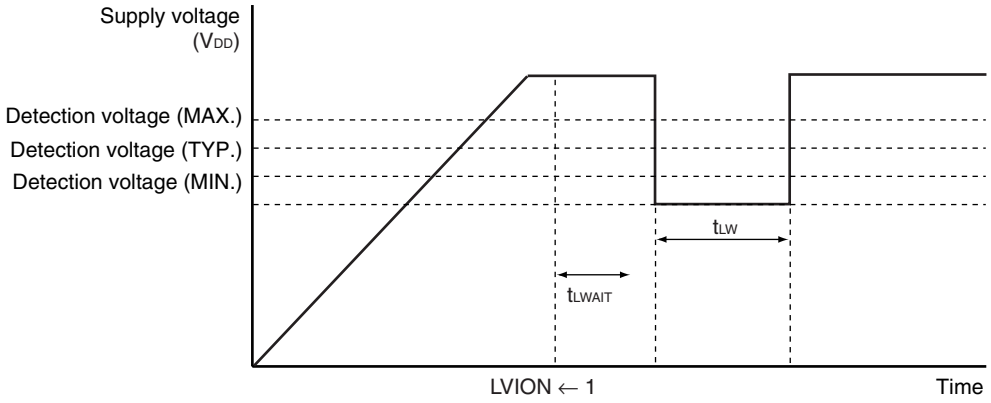
LVI Circuit Characteristics (T_A = –40 to +85°C, $V_{POC} \leq V_{DD} = EV_{DD} \leq 5.5$ V, $V_{SS} = EV_{SS}$ = 0 V)

Parameter		Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Detection voltage	Supply voltage level	V_{LVI0}	LVIS0 = 0	4.14	4.24	4.34	V
		V_{LVI1}	LVIS0 = 1	3.99	4.09	4.19	V
Minimum pulse width		t_{LW}		200			µs
Operation stabilization wait time[Note]		t_{LWAIT}				10	µs

Note Time required from setting bit 7 (LVION) of the low-voltage detection register (LVIM) to 1 to operation stabilization

Remark $V_{LVI0} > V_{LVI1}$

LVI Circuit Timing



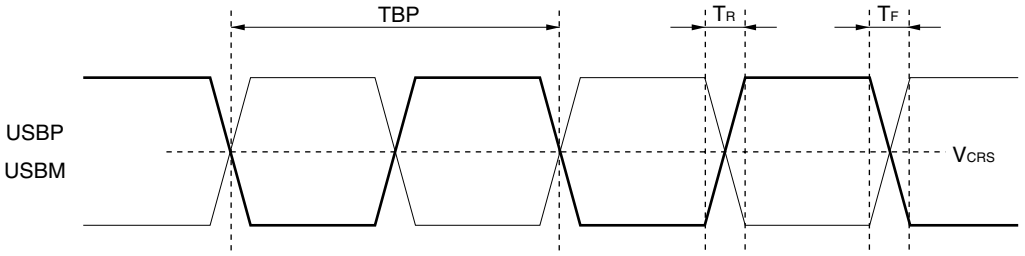
USBF Characteristics (T_A = –40 to +85°C, 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, 3.0 V ≤ $V_{USBREGC}$[Note 1] ≤ 3.6 V, V_{SS} = EV_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
USBM, USBP pin output rise/fall time	T_R, T_F	C_L = 50 pF[Note 2]	4		20	ns
Full-speed data rate	T_{DRATE}		11.97	12.00	12.03	Mbps
Bit period	TBP		83.12	83.33	83.54	ns
USBM, USBP pin rise/fall time matching	T_{RFM}	T_R/T_F	90		110	%
USBM, USBP pin output signal cross point voltage	V_{CRS}		1.3		2.0	V

Notes 1. Voltage output to the USBREGC pin

2. C_L is the load capacitance of the USBM and USBP output lines

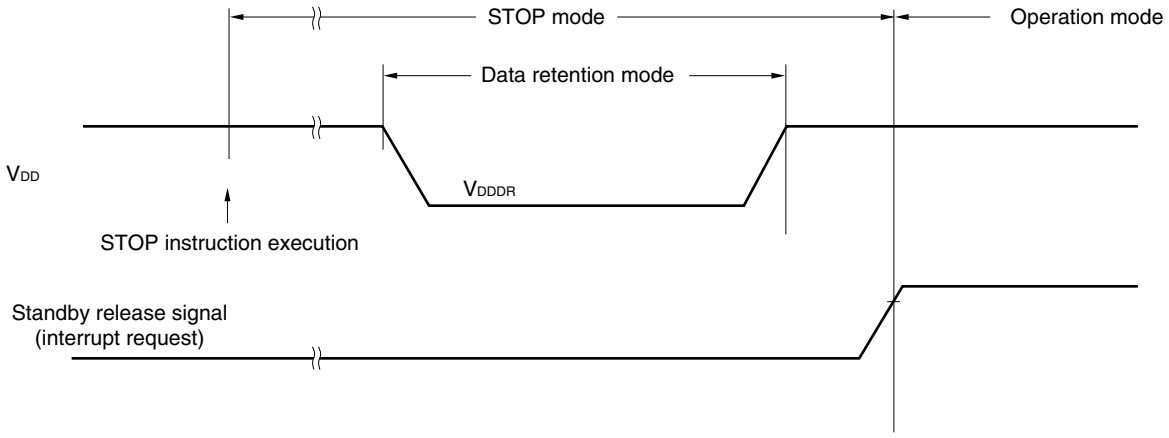
USBF Timing



Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics (T_A = –40 to +85°C)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention supply voltage	V_{DDDR}		1.44[Note]		5.5	V

Note The value depends on the POC detection voltage. When the voltage drops, the data is retained until a POC reset is effected, but data is not retained when a POC reset is effected.



Flash Memory Programming Characteristics

(T_A = –40 to +85°C, 4.0 V ≤ V_{DD} = EV_{DD} ≤ 5.5 V, V_{SS} = EV_{SS} = 0 V)

●Basic characteristics

Parameter		Symbol	Conditions	MIN.	TYP.	MAX.	Unit
V_{DD} supply current		I_{DD}	f_{XP} = 17.6 MHz (MAX.)		4.5	11.0	mA
Erase time Notes 1, 2	All block	T_{eraca}			10	200	ms
	Block unit	T_{erasa}			10	200	ms
Write time (in 8-bit units) Note 1		T_{wrwa}			10	100	µs
Number of rewrites per chip		C_{erwr}	Retention: 15 years 1 erase + 1 write after erase = 1 rewrite Note 3	100			Times

Notes 1. These are characteristics of the flash memory. These characteristic are not the rewrite time by a dedicated flash programmer (PG-FP5 or QB-MINI2) or by self programming.

2. The prewrite time before erasure and the erase verify time (writeback time) are not included.

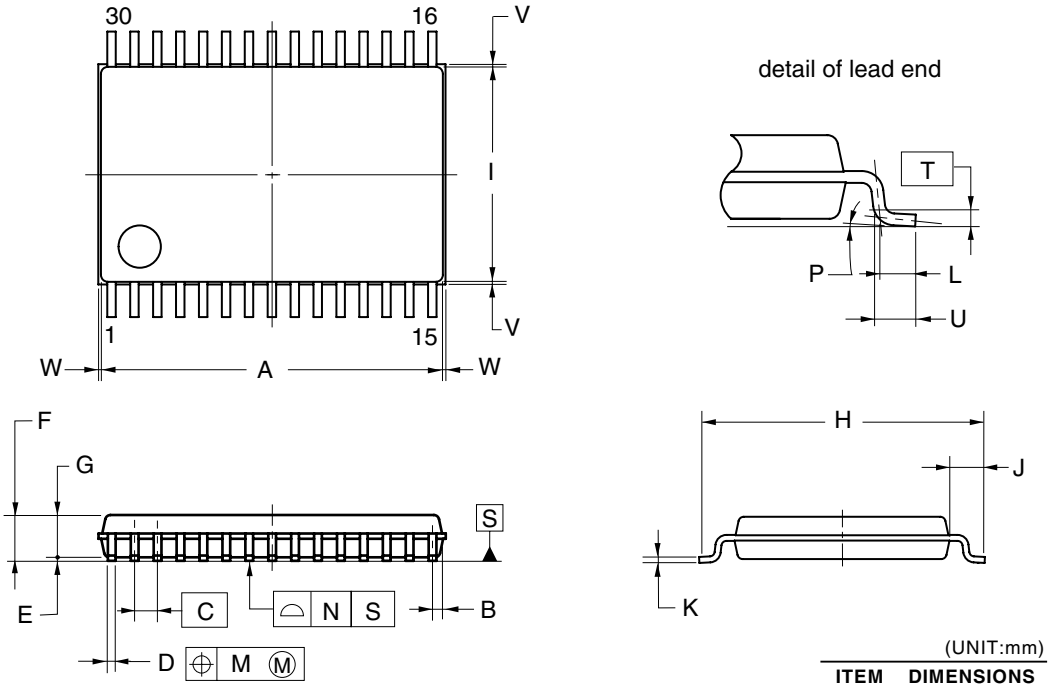
3. When a product is first written after shipment, “erase → write” and “write only” are both taken as one rewrite.

Remarks 1. f_{XP}: Main system clock oscillation frequency

2. For serial write operation characteristics, refer to **78K0/Kx2 Flash Memory Programming (Programmer) Application Note (U17739E)**.

CHAPTER 23 PACKAGE DRAWINGS

30-PIN PLASTIC SSOP (7.62mm (300))



(UNIT:mm)

ITEM	DIMENSIONS
A	9.70±0.10
B	0.30
C	0.65 (T.P.)
D	$0.22^{+0.10}_{-0.05}$
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	$0.15^{+0.05}_{-0.01}$
L	0.50
M	0.13
N	0.10
P	$3^{\circ}{}^{+5^{\circ}}_{-3^{\circ}}$
T	0.25(T.P.)
U	0.60±0.15
V	0.25 MAX.
W	0.15 MAX.

P30MC-65-CAB

NOTE

Each lead centerline is located within 0.13 mm of its true position (T.P.) at maximum material condition.

CHAPTER 24 RECOMMENDED SOLDERING CONDITIONS

These products should be soldered and mounted under the following recommended conditions.

For soldering methods and conditions other than those recommended below, please contact an Renesas Electronics sales representative.

For technical information, see the following website.

Semiconductor Device Mount Manual (http://www2.renesas.com/pkg/en/mount/index.html)

Table 24-1. Soldering Conditions of µPD78F0730

µPD78F0730MC-CAB-AX: 30-pin plastic SSOP (7.62 mm (300))

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 260°C, Time: 60 seconds max. (at 220°C or higher), Count: 3 times or less, Exposure limit: 7 days[Note] (after that, prebake at 125°C for 10 to 72 hours)	IR60-107-3
Wave soldering	Solder bath temperature: 260°C max., Time: 10 seconds max., Count: Once, Preheating temperature: 120°C max. (package surface temperature), Exposure limit: 7 days[Note] (after that, prebake at 125°C for 10 to 72 hours)	WS60-107-1
Partial heating	Pin temperature: 350°C max., Time: 3 seconds max. (per pin row)	–

Note After opening the dry pack, store it at 25°C or less and 65% RH or less for the allowable storage period.

Cautions 1. Do not use different soldering methods together (except for partial heating).

2. The µPD78F0730 has an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.

CHAPTER 25 CAUTIONS FOR WAIT

25.1 Cautions for Wait

This product has two internal system buses.

One is a CPU bus and the other is a peripheral bus that interfaces with the low-speed peripheral hardware.

Because the clock of the CPU bus and the clock of the peripheral bus are asynchronous, unexpected illegal data may be passed if an access to the CPU conflicts with an access to the peripheral hardware.

When accessing the peripheral hardware that may cause a conflict, therefore, the CPU repeatedly executes processing, until the correct data is passed.

As a result, the CPU does not start the next instruction processing but waits. If this happens, the number of execution clocks of an instruction increases by the number of wait clocks (for the number of wait clocks, see **Table 25-1**). This must be noted when real-time processing is performed.

25.2 Peripheral Hardware That Generates Wait

Table 25-1 lists the registers that issue a wait request when accessed by the CPU, and the number of CPU wait clocks.

Table 25-1. Registers That Generate Wait and Number of CPU Wait Clocks

Peripheral Hardware	Register	Access	Number of Wait Clocks
Serial interface UART6	ASIS6	Read	1 clock (fixed)

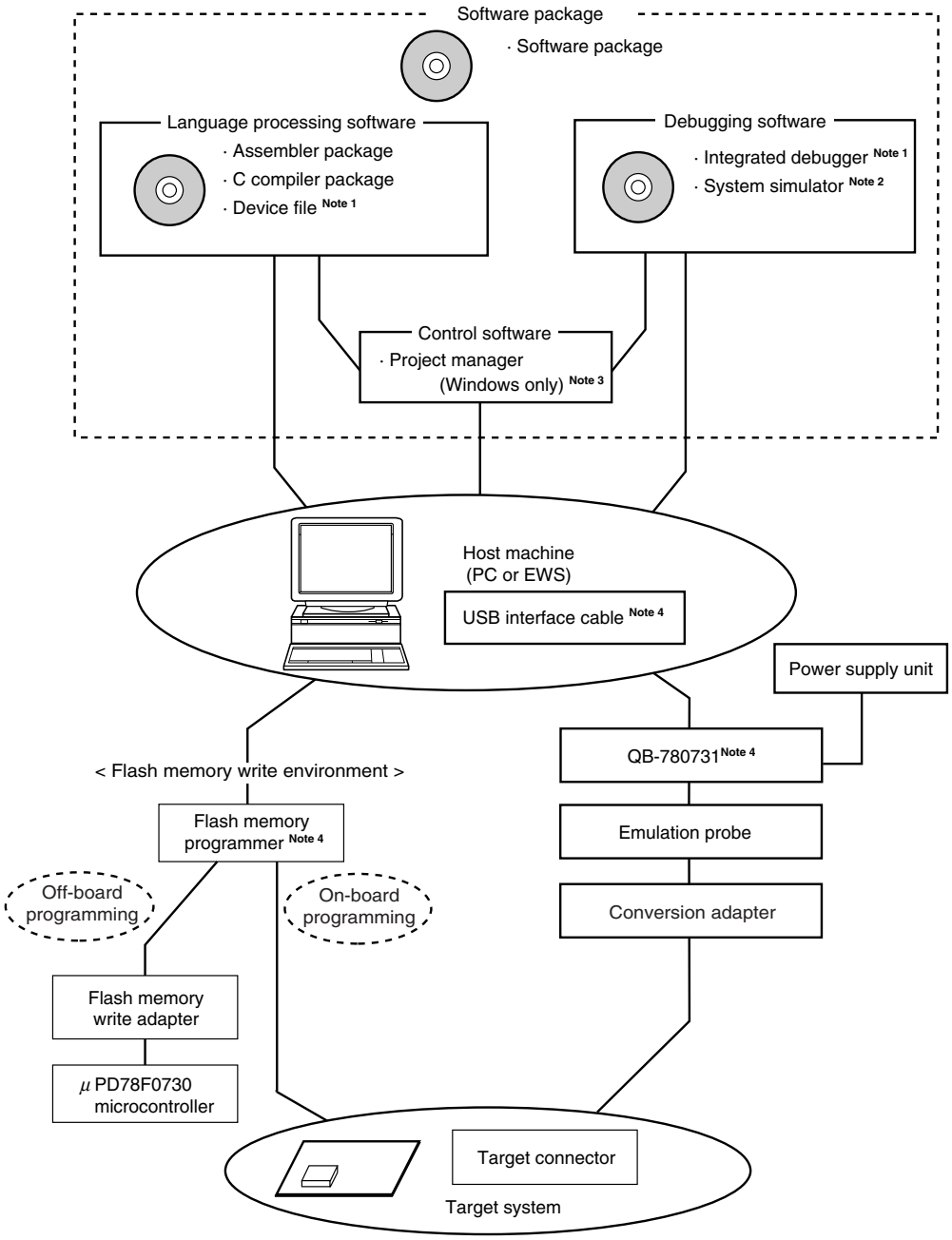
Remark The clock is the CPU clock (f_{CPU}).

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for the development of systems that employ the µPD78F0730.
Figure A-1 shows the development tool configuration.

Figure A-1. Development Tool Configuration (1/2)

(1) When using the in-circuit emulator QB-780731



Notes 1. Download the device file (DF780731) for µPD78F0730 microcontroller and the integrated debugger ID78K0-QB from the download site for development tools
<R> (http://www2.renesas.com/micro/en/ods/index.html).

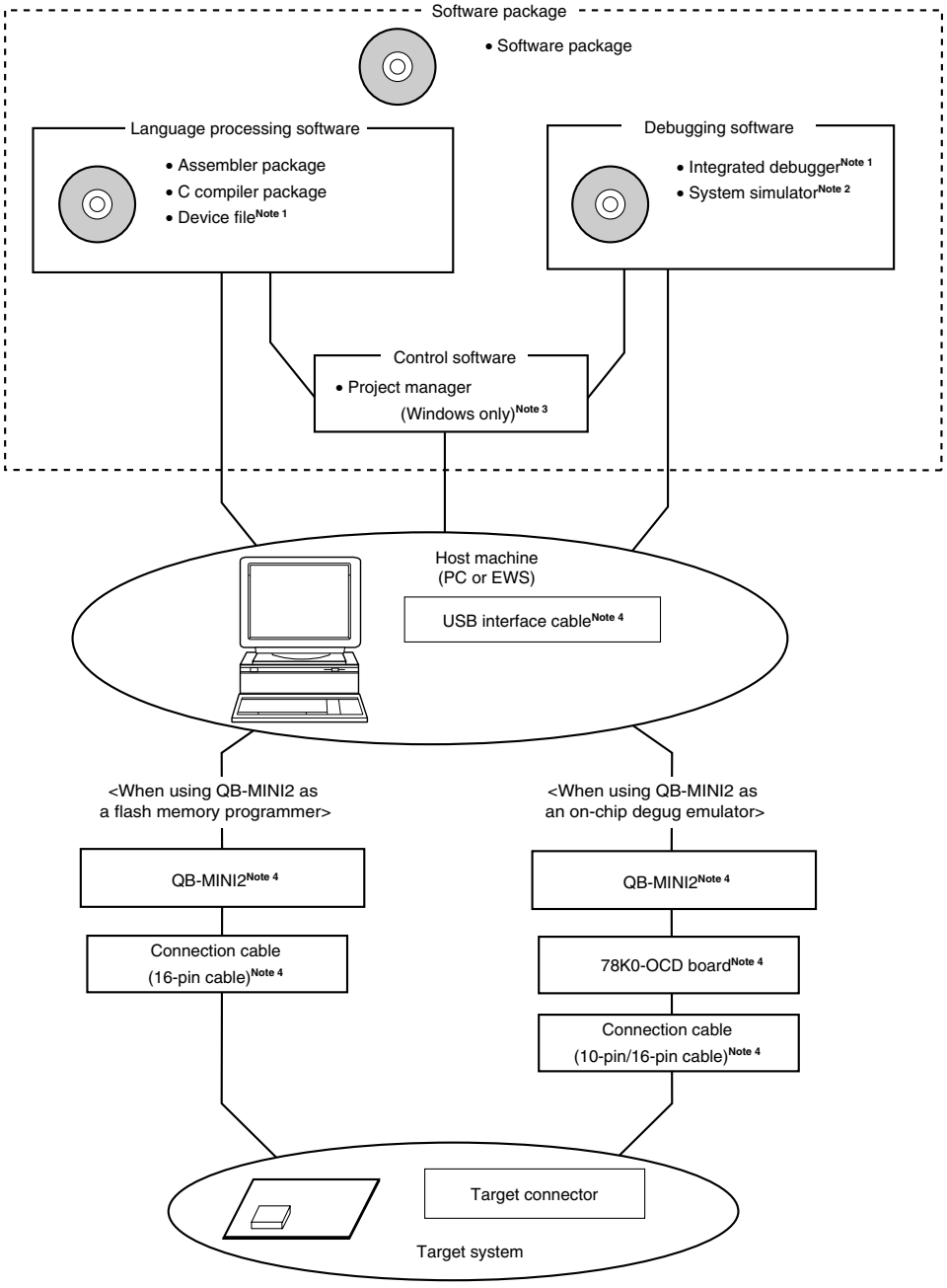
2. SM+ for 78K0 (instruction simulation version) is included in the software package.

3. The project manager PM+ is included in the assembler package.
PM+ cannot be used other than with Windows™.

4. QB-780731 is supplied with the integrated debugger ID78K0-QB, a USB interface cable, the on-chip debug emulator with programming function QB-MINI2, connection cables (10-pin and 16-pin cables), and the 78K0-OCD board. Any other products are sold separately.

Figure A-1. Development Tool Configuration (2/2)

(2) When using the on-chip debug emulator QB-MINI2



Notes 1. Download the device file (DF780731) for µPD78F0730 microcontroller and the integrated debugger ID78K0-QB from the download site for development tools
<R> (http://www2.renesas.com/micro/en/ods/index.html).

2. SM+ for 78K0 (instruction simulation version) is included in the software package.

3. The project manager PM+ is included in the assembler package.
PM+ cannot be used other than with Windows.

4. QB-MINI2 is supplied with USB interface cable, connection cables (10-pin cable and 16-pin cable), and 78K0-OCD board. Any other products are sold separately. In addition, download the software for operating the QB-MINI2 from the download site for development tools
<R> (http://www2.renesas.com/micro/en/ods/index.html).

A.1 Software Package

SP78K0 78K0 microcontrollers software package	Development tools (software) common to the 78K0 microcontrollers are combined in this package.

A.2 Language Processing Software

RA78K0 Note 1 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 (sold separately). **<Precaution when using RA78K0 in PC environment>** 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.
CC78K0 Note 1 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. **<Precaution when using CC78K0 in PC environment>** 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.
DF780731 Note 2 Device file	This file contains information peculiar to the device. This device file should be used in combination with a tool (RA78K0, CC78K0, ID78K0-QB, and the system simulator). The corresponding OS and host machine differ depending on the tool to be used.

Notes 1. If the versions of RA78K0 and CC78K0 are Ver.4.00 or later, different versions of RA78K0 and CC78K0 can be installed on the same machine.

2. The DF780731 can be used in common with the RA78K0, CC78K0, ID78K0-QB, and the system simulator. Download the DF780731 from the download site for development tools <R> (http://www2.renesas.com/micro/en/ods/index.html).

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. **<Caution>** The project manager is included in the assembler package (RA78K0). It can only be used in Windows.

A.4 Flash Memory Writing Tools

A.4.1 When using flash memory programmer FG-FP5 and FL-PR5

PG-FP5, FL-PR5 Flash memory programmer	Flash memory programmer dedicated to microcontrollers with on-chip flash memory.
QB-COMMON-PW Power supply unit	This product is mandatory for the In-circuit emulator IECUBE and for the Flash memory programmer PG-FP5. This product is commonly used for IECUBE for 78K0S, 78K0, 78K0R, V850 microcontrollers, and PG-FP5.
FA-30MC-CAB-B Flash memory programming adapter	Flash memory programming adapter used connected to the flash memory programmer. • FA-30MC-CAB-B : For 30-pin plastic SSOP (MC-CAB type)

Remarks 1. FL-PR5 and FA-30MC-CAB-B are products of Naito Densei Machida Mfg. Co., Ltd (http://www.ndk-m.co.jp/, TEL: +81-42-750-4172).

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 78K0/Kx2 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)
QB-Programmer	Software for operating the QB-MINI2

Remarks 1. The QB-MINI2 is supplied with a USB interface cable and connection cables (10-pin cable and 16-pin cable), and the 78K0-OCD board. A connection cable (10-pin cable) and the 78K0-OCD board are used only when using the on-chip debug function.

2. Download the QB-Programmer from the download site for development tools <R> (http://www2.renesas.com/micro/en/ods/index.html).

A.5 Debugging Tools (Hardware)

A.5.1 When using in-circuit emulator QB-780731

QB-780731[Note] In-circuit emulator	This in-circuit emulator serves to debug hardware and software when developing application systems using the µPD78F0730. 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-COMMON-PW Power supply unit	This product is mandatory for the In-circuit emulator IECUBE and for the Flash memory programmer PG-FP5. This product is commonly used for IECUBE for 78K0S, 78K0, 78K0R, V850 microcontrollers, and PG-FP5.
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-01T Exchange adapter	This exchange adapter is used to perform pin conversion from the in-circuit emulator to target connector. • QB-30MC-EA-01T: 30-pin plastic SSOP(MC-5A4 type)
QB-30MC-YS-01T Space adapter	This space adapter is used to adjust the height between the target system and in-circuit emulator. • QB-30MC-YS-01T: 30-pin plastic SSOP(MC-5A4 type)
QB-30MC-YQ-01T YQ connector	This YQ connector is used to connect the target connector and exchange adapter. • QB-30MC-YQ-01T: 30-pin plastic SSOP(MC-5A4 type)
QB-30MC-HQ-01T Mount adapter	This mount adapter is used to mount the target device with socket. • QB-30MC-HQ-01T: 30-pin plastic SSOP(MC-5A4 type)
QB-30MC-NQ-01T Target connector	This target connector is used to mount on the target system. • QB-30MC-NQ-01T: 30-pin plastic SSOP(MC-5A4 type)

Note The QB-780731 contains the integrated debugger ID78K0-QB and a USB interface cable. On-chip debug emulator with programming function QB-MINI2 is also contained.

Remark 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-780731-ZZZ	QB-780731	None			
QB-780731-T30MC		QB-80-EP-01T	QB-30MC-EA-01T	QB-30MC-YQ-01T	QB-30MC-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 μPD78F0730. 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 and connection cables (10-pin cable and 16-pin cable), and the 78K0-OCD board. A 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 <R> (http://www2.renesas.com/micro/en/ods/index.html).

A.6 Debugging Tools (Software)

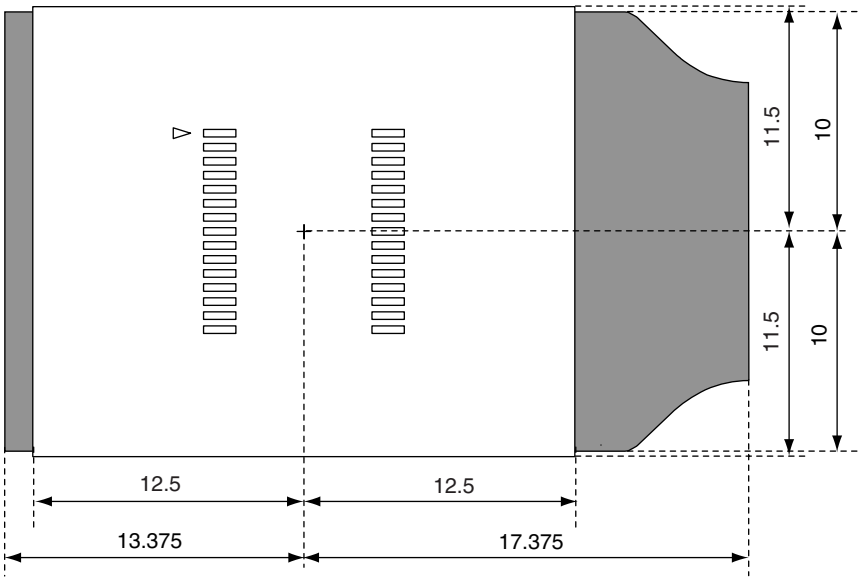
ID78K0-QB[Note] Integrated debugger	This debugger supports the in-circuit emulators for the 78K0 microcontrollers. The ID78K0-QB is Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. It should be used in combination with the device file (DF780731).
SM+ for 78K0 System simulator	System simulator is Windows-based software. It is used to perform debugging at the C source level or assembler level while simulating the operation of the target system on a host machine. Use of system simulator allows the execution of application logical testing and performance testing on an independent basis from hardware development, thereby providing higher development efficiency and software quality. SM+ for 78K0 (instruction simulation version) can only simulate a CPU. It is included in the software package. It should be used in combination with the device file (DF780731).

Note Download the ID78K0-QB from the download site for development tools <R> (http://www2.renesas.com/micro/en/ods/index.html).

APPENDIX B NOTES ON TARGET SYSTEM DESIGN

This chapter shows areas on the target system where component mounting is prohibited and areas where there are component mounting height restrictions when the QB-780731 is used.

Figure B-1. For 30-Pin MC Package



□ : Exchange adapter area: Components up to 17.45 mm in height can be mounted[Note]

■ : Emulation probe tip area: Components up to 24.45 mm in height can be mounted[Note]

Note Height can be adjusted by using space adapters (each adds 2.4 mm)

APPENDIX C REGISTER INDEX

C.1 Register Index (In Alphabetical Order with Respect to Register Names)

[1]

16-bit timer capture/compare register 000 (CR000) 116
16-bit timer capture/compare register 010 (CR010) 116
16-bit timer counter 00 (TM00) 116
16-bit timer mode control register 00 (TMC00) 119
16-bit timer output control register 00 (TOC00) 122

[8]

8-bit timer compare register 50 (CR50) 181
8-bit timer compare register 51 (CR51) 181
8-bit timer counter 50 (TM50) 181
8-bit timer counter 51 (TM51) 181
8-bit timer H carrier control register 1 (TMCYC1) 201
8-bit timer H compare register 01 (CMP01) 198
8-bit timer H compare register 11 (CMP11) 198
8-bit timer H mode register 1 (TMHMD1) 199
8-bit timer mode control register 50 (TMC50) 184
8-bit timer mode control register 51 (TMC51) 184

[A]

Asynchronous serial interface operation mode register 6 (ASIM6) 228
Asynchronous serial interface reception error status register 6 (ASIS6) 230
Asynchronous serial interface transmission status register 6 (ASIF6) 231

[B]

Baud rate generator control register 6 (BRGC6) 233

[C]

Capture/compare control register 00 (CRC00) 121
Clock operation mode select register (OSCCTL) 86
Clock selection register 6 (CKSR6) 232

[E]

External interrupt falling edge enable register (EGN) 403
External interrupt rising edge enable register (EGP) 403

[I]

Internal expansion RAM size switching register (IXS) 455
Internal memory size switching register (IMS) 454
Internal oscillation mode register (RCM) 88
Interrupt mask flag register 0H (MK0H) 401
Interrupt mask flag register 0L (MK0L) 401
Interrupt mask flag register 1H (MK1H) 401
Interrupt mask flag register 1L (MK1L) 401
Interrupt request flag register 0H (IF0H) 399

Interrupt request flag register 0L (IF0L) 399
Interrupt request flag register 1H (IF1H) 399
Interrupt request flag register 1L (IF1L) 399

[L]

Low-voltage detection level selection register (LVIS) 441
Low-voltage detection register (LVIM) 440

[M]

Main clock mode register (MCM) 90
Main OSC control register (MOC) 89

[O]

Oscillation stabilization time counter status register (OSTC) 91, 413
Oscillation stabilization time select register (OSTS) 92, 414

[P]

PLL control register (PLLC) 93
Port mode register 0 (PM0) 77
Port mode register 1 (PM1) 77
Port mode register 12 (PM12) 77
Port mode register 3 (PM3) 77
Port mode register 6 (PM6) 77
Port register 0 (P0) 78
Port register 1 (P1) 78
Port register 12 (P12) 78
Port register 3 (P3) 78
Port register 6 (P6) 78
Prescaler mode register 00 (PRM00) 124
Priority specification flag register 0H (PR0H) 402
Priority specification flag register 0L (PR0L) 402
Priority specification flag register 1H (PR1H) 402
Priority specification flag register 1L (PR1L) 402
Processor clock control register (PCC) 87
Pull-up resistor option register 0 (PU0) 79
Pull-up resistor option register 1 (PU1) 79
Pull-up resistor option register 12 (PU12) 79
Pull-up resistor option register 3 (PU3) 79

[R]

Receive buffer register 6 (RXB6) 227
Receive shift register 6 (RXS6) 227
Reset control flag register (RESF) 432

[S]

Serial clock selection register 10 (CSIC10) 259
Serial I/O shift register 10 (SIO10) 257
Serial operation mode register 10 (CSIM10) 258

[T]

Timer clock selection register 50 (TCL50) 182
Timer clock selection register 51 (TCL51) 182
Transmit buffer register 10 (SOTB10) 257
Transmit buffer register 6 (TXB6) 227
Transmit shift register 6 (TXS6) 227

[U]

UF0 active alternative setting register (UF0AAS) 319
UF0 active interface number register (UF0AIFN) 318
UF0 address register (UF0ADRS) 341
UF0 alternative setting status register (UF0ASS) 320
UF0 bulk in 1 register (UF0BI1) 333
UF0 bulk out 1 length register (UF0BO1L) 332
UF0 bulk out 1 register (UF0BO1) 329
UF0 CLR request register (UF0CLR) 290
UF0 configuration register (UF0CNF) 342
UF0 configuration/interface/endpoint descriptor registers 0 to 255 (UF0CIE0 to UF0CIE255) 347
UF0 data end register (UF0DEND) 314
UF0 descriptor length register (UF0DSCL) 345
UF0 device descriptor registers 0 to 17 (UF0DD0 to UF0DD17) 346
UF0 device status register L (UF0DSTL) 337
UF0 endpoint 1 interface mapping register (UF0E1IM) 321
UF0 endpoint 2 interface mapping register (UF0E2IM) 322
UF0 EP status 0 register (UF0EPS0) 292
UF0 EP status 1 register (UF0EPS1) 293
UF0 EP status 2 register (UF0EPS2) 294
UF0 EP0 length register (UF0E0L) 324
UF0 EP0 read register (UF0E0R) 323
UF0 EP0 setup register (UF0E0ST) 325
UF0 EP0 status register L (UF0E0SL) 338
UF0 EP0 write register (UF0E0W) 327
UF0 EP0NAK register (UF0E0N) 283
UF0 EP0NAKALL register (UF0E0NA) 285
UF0 EP1 status register L (UF0E1SL) 339
UF0 EP2 status register L (UF0E2SL) 340
UF0 EPNAK mask register (UF0ENM) 288
UF0 EPNAK register (UF0EN) 286
UF0 FIFO clear 0 register (UF0FIC0) 312
UF0 FIFO clear 1 register (UF0FIC1) 313
UF0 GPR register (UF0GPR) 315
UF0 INT clear 0 register (UF0IC0) 307
UF0 INT clear 1 register (UF0IC1) 308
UF0 INT clear 2 register (UF0IC2) 309
UF0 INT clear 3 register (UF0IC3) 310
UF0 INT clear 4 register (UF0IC4) 311
UF0 INT mask 0 register (UF0IM0) 302
UF0 INT mask 1 register (UF0IM1) 303
UF0 INT mask 2 register (UF0IM2) 304
UF0 INT mask 3 register (UF0IM3) 305

UF0 INT mask 4 register (UF0IM4) 306
UF0 INT status 0 register (UF0IS0) 295
UF0 INT status 1 register (UF0IS1) 297
UF0 INT status 2 register (UF0IS2) 299
UF0 INT status 3 register (UF0IS3) 300
UF0 INT status 4 register (UF0IS4) 301
UF0 interface 0 register (UF0IF0) 343
UF0 interface 1 to 4 registers (UF0IF1 to UF0IF4) 344
UF0 mode control register (UF0MODC) 316
UF0 mode status register (UF0MODS) 317
UF0 SET request register (UF0SET) 291
UF0 SNDSIE register (UF0SDS) 289
USB clock control register (UCKC) 94
USB function 0 buffer control register (UF0BC) 349

[W]

Watchdog timer enable register (WDTE) 220

C.2 Register Index (In Alphabetical Order with Respect to Register Symbol)

[A]

ASIF6: Asynchronous serial interface transmission status register 6 231
ASIM6: Asynchronous serial interface operation mode register 6 228
ASIS6: Asynchronous serial interface reception error status register 6 230

[B]

BRGC6: Baud rate generator control register 6 233

[C]

CKSR6: Clock selection register 6 232
CMP01: 8-bit timer H compare register 01 198
CMP11: 8-bit timer H compare register 11 198
CR000: 16-bit timer capture/compare register 000 116
CR010: 16-bit timer capture/compare register 010 116
CR50: 8-bit timer compare register 50 181
CR51: 8-bit timer compare register 51 181
CRC00: Capture/compare control register 00 121
CSIC10: Serial clock selection register 10 259
CSIM10: Serial operation mode register 10 258

[E]

EGN: External interrupt falling edge enable register 403
EGP: External interrupt rising edge enable register 403

[I]

IF0H: Interrupt request flag register 0H 399
IF0L: Interrupt request flag register 0L 399
IF1H: Interrupt request flag register 1H 399
IF1L: Interrupt request flag register 1L 399
IMS: Internal memory size switching register 454
IXS: Internal expansion RAM size switching register 455

[L]

LVIM: Low-voltage detection register 440
LVIS: Low-voltage detection level selection register 441

[M]

MCM: Main clock mode register 90
MK0H: Interrupt mask flag register 0H 401
MK0L: Interrupt mask flag register 0L 401
MK1H: Interrupt mask flag register 1H 401
MK1L: Interrupt mask flag register 1L 401
MOC: Main OSC control register 89

[O]

OSCCTL: Clock operation mode select register 86
OSTC: Oscillation stabilization time counter status register 91, 413
OSTS: Oscillation stabilization time select register 92, 414

[P]

P0: Port register 0 78
P1: Port register 1 78
P12: Port register 12 78
P3: Port register 3 78
P6: Port register 6 78
PCC: Processor clock control register 87
PLLC: PLL control register 93
PM0: Port mode register 0 77
PM1: Port mode register 1 77
PM12: Port mode register 12 77
PM3: Port mode register 3 77
PM6: Port mode register 6 77
PR0H: Priority specification flag register 0H 402
PR0L: Priority specification flag register 0L 402
PR1H: Priority specification flag register 1H 402
PR1L: Priority specification flag register 1L 402
PRM00: Prescaler mode register 00 124
PU0: Pull-up resistor option register 0 79
PU1: Pull-up resistor option register 1 79
PU12: Pull-up resistor option register 12 79
PU3: Pull-up resistor option register 3 79

[R]

RCM: Internal oscillation mode register 88
RESF: Reset control flag register 432
RXB6: Receive buffer register 6 227
RXS6: Receive shift register 6 227

[S]

SIO10: Serial I/O shift register 10 257
SOTB10: Transmit buffer register 10 257

[T]

TCL50: Timer clock selection register 50 182
TCL51: Timer clock selection register 51 182
TM00: 16-bit timer counter 00 116
TM50: 8-bit timer counter 50 181
TM51: 8-bit timer counter 51 181
TMC00: 16-bit timer mode control register 00 119
TMC50: 8-bit timer mode control register 50 184
TMC51: 8-bit timer mode control register 51 184
TMCYC1: 8-bit timer H carrier control register 1 201
TMHMD1: 8-bit timer H mode register 1 199
TOC00: 16-bit timer output control register 00 122
TXB6: Transmit buffer register 6 227
TXS6: Transmit shift register 6 227

[U]

UCKC: USB clock control register 94

UF0AAS: UF0 active alternative setting register 319
UF0ADRS: UF0 address register 341
UF0AIFN: UF0 active interface number register 318
UF0ASS: UF0 alternative setting status register 320
UF0BC: USB function 0 buffer control register 349
UF0BI1: UF0 bulk in 1 register 333
UF0BO1: UF0 bulk out 1 register 329
UF0BO1L: UF0 bulk out 1 length register 332
UF0CIE0 to UF0CIE255: UF0 configuration/interface/endpoint descriptor registers 0 to 255 347
UF0CLR: UF0 CLR request register 290
UF0CNF : UF0 configuration register 342
UF0DD0 to UF0DD17: UF0 device descriptor registers 0 to 17 346
UF0DEND: UF0 data end register 314
UF0DSCL: UF0 descriptor length register 345
UF0DSTL: UF0 device status register L 337
UF0E0L: UF0 EP0 length register 324
UF0E0N: UF0 EP0NAK register 283
UF0E0NA: UF0 EP0NAKALL register 285
UF0E0R: UF0 EP0 read register 323
UF0E0SL: UF0 EP0 status register L 338
UF0E0ST: UF0 EP0 setup register 325
UF0E0W: UF0 EP0 write register 327
UF0E1IM: UF0 endpoint 1 interface mapping register 321
UF0E1SL: UF0 EP1 status register L 339
UF0E2IM: UF0 endpoint 2 interface mapping register 322
UF0E2SL: UF0 EP2 status register L 340
UF0EN: UF0 EPNAK register 286
UF0ENM: UF0 EPNAK mask register 288
UF0EPS0: UF0 EP status 0 register 292
UF0EPS1: UF0 EP status 1 register 293
UF0EPS2: UF0 EP status 2 register 294
UF0FIC0: UF0 FIFO clear 0 register 312
UF0FIC1: UF0 FIFO clear 1 register 313
UF0GPR: UF0 GPR register 315
UF0IC0: UF0 INT clear 0 register 307
UF0IC1: UF0 INT clear 1 register 308
UF0IC2: UF0 INT clear 2 register 309
UF0IC3: UF0 INT clear 3 register 310
UF0IC4: UF0 INT clear 4 register 311
UF0IF0: UF0 interface 0 register 343
UF0IF1 to UF0IF4: UF0 interface 1 to 4 registers 344
UF0IM0: UF0 INT mask 0 register 302
UF0IM1: UF0 INT mask 1 register 303
UF0IM2: UF0 INT mask 2 register 304
UF0IM3: UF0 INT mask 3 register 305
UF0IM4: UF0 INT mask 4 register 306
UF0IS0: UF0 INT status 0 register 295
UF0IS1: UF0 INT status 1 register 297
UF0IS2: UF0 INT status 2 register 299
UF0IS3: UF0 INT status 3 register 300

UF0IS4: UF0 INT status 4 register 301
UF0MODC: UF0 mode control register 316
UF0MODS: UF0 mode status register 317
UF0SDS: UF0 SNDSIE register 289
UF0SET: UF0 SET request register 291

[W]

WDTE: Watchdog timer enable register 220

APPENDIX D REVISION HISTORY

D.1 Major Revisions in This Edition

Page	Description	Classification
CHAPTER 12 USB FUNCTION CONTROLLER USBF		
p. 393	Modification of **Figure 12-27. USB Connection Example**	(c)
CHAPTER 16 POWER-ON-CLEAR CIRCUIT		
p. 433	Modification of **Note** in **16.1 Functions of Power-on-Clear Circuit**	(c)
CHAPTER 19 FLASH MEMORY		
p. 459	Modification of **Figure 19-6. Communication with Dedicated Flash Memory Programmer (CSI10)**	(a)
APPENDIX A DEVELOPMENT TOOLS		
pp. 510 to 513, 515	Modification of URL of the download site for development tools	(c)

Remark “Classification” in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

D.2 Revision History of Preceding Editions

Here is the revision history of the preceding editions. Chapter indicates the chapter of each edition.

(1/2)

Edition	Description	Chapter
2nd Edition	Modification of related documents	Throughout
	Deletion of flash memory programmer PG-FP4 and FP-PR4	
	Deletion of on-chip debug emulator QB-78K0MINI	
	Addition of on-chip debug emulator with programming function QB-MINI2	
	Modification of state of USBPUC pin after reset in **2.1 Pin Function List**	CHAPTER 2 PIN FUNCTIONS
	Modification of **Remark** in **2.2.3 P30-P33 (Port 3)**	
	Modification of **Remark** in **2.2.5 P120-P122 (Port 12)**	
	Addition of description in **2.2.14 FLMD0**	
	Modification of **Notes 4** in **Table 2-2 Pin I/O Circuit Types**	
	Modification of **Caution** in **12.1 Overview**	CHAPTER 12 USB FUNCTION CONTROLLER USBF
	Modification of **Note** in **Figure 12-25 (b) Processing on power failure**	
	12.8 External Circuit Configuration • Addition of **12.8.1 Outline** • Modification of **12.8.2 USB connection example**	
	Addition of **12.9 Cautions for USB Function Controller USBF**	
	Modification of **Note 2** in **Table 19-1 Wiring Between µPD78F0730 and Dedicated Flash Memory Programmer**	CHAPTER 19 FLASH MEMORY
	Modification of **Note** in **Figure 19-4 Example of Wiring Adapter for Flash Memory Writing in UART (UART6) Mode**	
	Modification of **Note** in **Figure 19-7 Communication with Dedicated Flash Memory Programmer (UART6)**	
	Modification of description in **19.5 (2) UART6**	
	Modification of **Note 1** in **Table 19-2 Pin Connection**	
	Modification of capacitance of a capacitor in **19.6.5 REGC pin**	
	Modification of description in **19.6.6 Other signal pins**	
	Modification of a related document in **19.9 Flash Memory Programming by Self-Programming**	
	Modification of **Remark** in **Figure 19-16 Flow of Self Programming (Rewriting Flash Memory)**	

(2/2)

Edition	Description	Chapter
2nd Edition	Modification of **Figure 20-1 Connection Example of QB-MINI2 and µPD78F0730 (When OCD0A/X1 and OCD0B/X2 Are Used)**	CHAPTER 20 ON-CHIP DEBUG FUNCTION
	Modification of **Figure 20-2 Connection Example of QB-MINI2 and µPD78F0730 (When OCD1A and OCD1B Are Used)**	
	Addition of **Caution** in **Figure 20-3 Connection of FLMD0 Pin for Self Programming by Means of On-Chip Debugging**	
	Addition of **20.2 Reserved Area Used by QB-MINI2**	
	Change of target specification to official specification	CHAPTER 22 ELECTRICAL SPECIFICATIONS
	Addition of **Recommended Oscillator Constants**	
	Addition of this chapter	CHAPTER 24 RECOMMENDED SOLDERING CONDITIONS
	Modification of **Figure A-1 Development Tool Configuration**	APPENDIX A DEVELOPMENT TOOLS
	Modification of **Notes 1, 2** in **A.2 Language Processing Software**	
	A.4 Flash Memory Writing Tools • Modification of **A.4.1 When using flash memory programmer FG-FP5 and FL-PR5** • Addition of **A.4.2 When using on-chip debug emulator with programming function QB-MINI2**	
	A.5 Debugging Tools (Hardware) • Modification of **A.5.1 When using in-circuit emulator QB-780731** • Modification of **A.5.2 When using on-chip debug emulator with programming function QB-MINI2**	
	A.6 Debugging Tools (Software) • Addition of **Note** in **ID78K0-QB** • Addition of **SM+ for 78K0**	
	Addition of this chapter	APPENDIX B NOTES ON TARGET SYSTEM DESIGN
	Addition of this chapter	APPENDIX D REVISION HISTORY

μPD78F0730
User's Manual: Hardware

Publication Date: Rev.0.01 Dec 28, 2007
Rev.3.00 Sep 22, 2011

Published by: Renesas Electronics Corporation



SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
7F, No. 363 Fu Shing North Road Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

© 2011 Renesas Electronics Corporation. All rights reserved.

Colophon 1.0

μPD78F0730

Renesas Electronics Corporation

R01UH0308EJ0300