To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

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.

RENESAS

User's Manual

V850ES/SG1

32-bit Single-Chip Microcontrollers

Hardware

μPD703249Y
μPD703252Y
μPD703253Y

Document No. U17329EJ2V0UD00 (2nd edition)
Date Published May 2008 N

© NEC Electronics Corporation 2005, 2008
Printed in Japan

[MEMO]

NOTES FOR CMOS DEVICES

① VOLTAGE APPLICATION WAVEFORM AT INPUT PIN

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (MAX) and V_{IH} (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (MAX) and V_{IH} (MIN).

② HANDLING OF UNUSED INPUT PINS

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

③ PRECAUTION AGAINST ESD

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

④ STATUS BEFORE INITIALIZATION

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

⑤ POWER ON/OFF SEQUENCE

In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current.

The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

⑥ INPUT OF SIGNAL DURING POWER OFF STATE

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements.

Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

IECUBE is a registered trademark of NEC Electronics Corporation in Japan and Germany.
MINICUBE is a registered trademark of NEC Electronics Corporation in Japan and Germany or a trademark in the United States of America.
EEPROM, IEBus, and Inter Equipment Bus are trademarks of NEC Electronics Corporation.
Windows and Windows NT are either registered trademarks or a trademarks of Microsoft Corporation in the United States and/or other countries.
PC/AT is a trademark of International Business Machines Corporation.
SPARCstation is a trademark of SPARC International, Inc.
Solaris and SunOS are trademarks of Sun Microsystems, Inc.

These commodities, technology or software, must be exported in accordance with the export administration regulations of the exporting country. Diversion contrary to the law of that country is prohibited.

- **The information in this document is current as of March, 2008. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".
  The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.
  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).
  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

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

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

M8E 02. 11-1

PREFACE

Readers

This manual is intended for users who wish to understand the functions of the V850ES/SG1 and design application systems using these products.

Purpose

This manual is intended to give users an understanding of the hardware functions of the V850ES/SG1 shown in the **Organization** below.

Organization

This manual is divided into two parts: Hardware (this manual) and Architecture (**V850ES Architecture User's Manual**).

Hardware	Architecture
• Pin functions	• Data types
• CPU function	• Register set
• On-chip peripheral functions	• Instruction format and instruction set
• Electrical specifications	• Interrupts and exceptions
	• Pipeline operation

How to Read This Manual

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

To understand the overall functions of the V850ES/SG1
→Read this manual according to the **CONTENTS**.

To find the details of a register where the name is known
→Use **APPENDIX C REGISTER INDEX**.

Register format
→The name of the bit whose number is in angle brackets (<>) in the figure of the register format of each register is defined as a reserved word in the device file.

To understand the details of an instruction function
→Refer to the **V850ES Architecture User's Manual** available separately.

To know the electrical specifications of the V850ES/SG1
→See **CHAPTER 26 ELECTRICAL SPECIFICATIONS**.

The "yyy bit of the xxx register" is described as the "xxx.yyy bit" in this manual. Note with caution that if "xxx.yyy" is described as is in a program, however, the compiler/assembler cannot recognize it correctly.

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.

Conventions

Data significance:	Higher digits on the left and lower digits on the right
Active low representation:	$\overline{xxx}$ (overscore over pin or signal name)
Memory map address:	Higher addresses on the top and lower addresses on the bottom
Note:	Footnote for item marked with **Note** in the text
Caution:	Information requiring particular attention
Remark:	Supplementary information
Numeric representation:	Binary ... xxxx or xxxxB Decimal ... xxxx Hexadecimal ... xxxxH
Prefix indicating power of 2 (address space, memory capacity):	K (kilo): $2^{10} = 1,024$ M (mega): $2^{20} = 1,024^2$ G (giga): $2^{30} = 1,024^3$

Related Documents

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

Documents related to V850ES/SG1

Document Name	Document No.
V850ES Architecture User's Manual	U15943E
V850ES/SG1 Hardware User's Manual	This manual
V850ES/SG2, V850ES/SG2-H Hardware User's Manual	U16541E

Documents related to development tools

Document Name		Document No.
IE-V850ES-G1 (In-Circuit Emulator)		U16313E
IE-703288-G1-EM1 (In-Circuit Emulator Option Board)		U16697E
IE-V850E1-CD-NW (PCMCIA Card Type On-Chip Debug Emulator)		U16647E
QB-V850ESSX2 (In-Circuit Emulator)		U17091E
QB-V850MINI (On-Chip Debug Emulator)		U17638E
QB-MINI2 (On-Chip Debug Emulator with Programming Function)		U18371E
CA850 Ver. 3.20 C Compiler Package	Operation	U18512E
	C Language	U18513E
	Assembly Language	U18514E
	Link Directives	U18515E
PM+ Ver. 6.30 Project Manager		U18416E
ID850 Ver. 3.00 Integrated Debugger	Operation	U17358E
ID850QB Ver. 3.40 Integrated Debugger	Operation	U18604E
TW850 Ver. 2.00 Performance Analysis Tuning Tool		U17241E
SM+ System Simulator	Operation	U18601E
	User Open Interface	U18212E
RX850 Ver. 3.20 Real-Time OS	Basics	U13430E
	Installation	U17419E
	Technical	U13431E
	Task Debugger	U17420E
RX850 Pro Ver. 3.21 Real-Time OS	Basics	U18165E
	Installation	U17421E
	Technical	U13772E
	Task Debugger	U17422E
AZ850 Ver. 3.30 System Performance Analyzer		U17423E

CONTENTS

CHAPTER 1 INTRODUCTION 20

1.1 General 20
1.2 Features 22
1.3 Application Fields 23
1.4 Ordering Information 23
1.5 Pin Configuration (Top View) 24
1.6 Function Block Configuration 28
1.6.1 Internal block diagram 28
1.6.2 Internal units 30

CHAPTER 2 PIN FUNCTIONS 33

2.1 List of Pin Functions 33
2.2 Pin States 42
2.3 Pin I/O Circuit Types, I/O Buffer Power Supplies, and Connection of Unused Pins 43
2.4 Cautions 47

CHAPTER 3 CPU FUNCTION 48

3.1 Features 48
3.2 CPU Register Set 49
3.2.1 Program register set 50
3.2.2 System register set 51
3.3 Operation Modes 57
3.3.1 Specifying operation mode 57
3.4 Address Space 58
3.4.1 CPU address space 58
3.4.2 Wraparound of CPU address space 59
3.4.3 Memory map 60
3.4.4 Areas 62
3.4.5 Recommended use of address space 66
3.4.6 Peripheral I/O registers 69
<R> 3.4.7 Programmable peripheral I/O registers 79
3.4.8 Special registers 80
3.4.9 Cautions 84

CHAPTER 4 PORT FUNCTIONS 89

4.1 Features 89
4.2 Basic Port Configuration 89
4.3 Port Configuration 90
4.3.1 Port 0 95
4.3.2 Port 1 98
4.3.3 Port 3 100
4.3.4 Port 4 106
4.3.5 Port 5 109
4.3.6 Port 7 113

4.3.7 Port 9 115
4.3.8 Port CM 123
4.3.9 Port CT 125
4.3.10 Port DH 127
4.3.11 Port DL 129
4.4 Block Diagrams 132
4.5 Port Register Settings When Alternate Function Is Used 168
4.6 Cautions 175
4.6.1 Cautions on setting port pins 175
4.6.2 Cautions on bit manipulation instruction for port n register (Pn) 178
4.6.3 Cautions on on-chip debug pins 179
4.6.4 Cautions on P10, P11, and P53 pins when power is turned on 180
4.6.5 Hysteresis characteristics 180
<R> 4.6.6 Cautions on separate bus mode 180

CHAPTER 5 BUS CONTROL FUNCTION 181

5.1 Features 181
5.2 Bus Control Pins 182
5.2.1 Pin status when internal ROM, internal RAM, or on-chip peripheral I/O is accessed 182
5.2.2 Pin status in each operation mode 182
5.3 Memory Block Function 183
5.4 External Bus Interface Mode Control Function 184
5.5 Bus Access 185
5.5.1 Number of clocks for access 185
5.5.2 Bus size setting function 185
5.5.3 Access by bus size 186
5.6 Wait Function 193
5.6.1 Programmable wait function 193
5.6.2 External wait function 194
5.6.3 Relationship between programmable wait and external wait 195
5.6.4 Programmable address wait function 196
5.7 Idle State Insertion Function 197
5.8 Bus Hold Function 198
5.8.1 Functional outline 198
5.8.2 Bus hold procedure 199
5.8.3 Operation in power save mode 199
5.9 Bus Priority 200
5.10 Bus Timing 201

CHAPTER 6 CLOCK GENERATION FUNCTION 207

6.1 Overview 207
6.2 Configuration 208
6.3 Registers 210
6.4 Operation 215
6.4.1 Operation of each clock 215
6.4.2 Clock output function 215
6.5 PLL Function 216

6.5.1 Overview 216
6.5.2 Registers 216
6.5.3 Usage 220

CHAPTER 7 16-BIT TIMER/EVENT COUNTER P (TMP) 221

7.1 Overview 221
7.2 Functions 221
7.3 Configuration 222
7.4 Registers 224
7.5 Timer Output Operations 237
7.6 Operation 238
7.6.1 Interval timer mode (TPnMD2 to TPnMD0 bits = 000) 245
7.6.2 External event count mode (TPnMD2 to TPnMD0 bits = 001) 257
7.6.3 External trigger pulse output mode (TPnMD2 to TPnMD0 bits = 010) 266
7.6.4 One-shot pulse output mode (TPnMD2 to TPnMD0 bits = 011) 278
7.6.5 PWM output mode (TPnMD2 to TPnMD0 bits = 100) 285
7.6.6 Free-running timer mode (TPnMD2 to TPnMD0 bits = 101) 294
7.6.7 Pulse width measurement mode (TPnMD2 to TPnMD0 bits = 110) 312

CHAPTER 8 16-BIT INTERVAL TIMER M (TMM) 318

8.1 Overview 318
8.2 Configuration 319
8.3 Register 321
8.4 Operation 322
8.4.1 Interval timer mode 322
8.4.2 Cautions 326

CHAPTER 9 WATCH TIMER FUNCTIONS 327

9.1 Functions 327
9.2 Configuration 328
9.3 Control Registers 330
9.4 Operation 334
9.4.1 Operation as watch timer 334
9.4.2 Operation as interval timer 335
9.4.3 Cautions 336

CHAPTER 10 FUNCTIONS OF WATCHDOG TIMER 2 337

10.1 Functions 337
10.2 Configuration 338
10.3 Registers 339
10.4 Operation 342

CHAPTER 11 A/D CONVERTER 343

11.1 Overview 343
11.2 Functions 343
11.3 Configuration 344

11.4 Registers 347
11.5 Operation 358
11.5.1 Basic operation 358
11.5.2 Conversion operation timing 359
11.5.3 Trigger mode 360
11.5.4 Operation mode 362
11.5.5 Power-fail compare mode 366
11.6 Cautions 371
11.7 How to Read A/D Converter Characteristics Table 376

<R> **CHAPTER 12 D/A CONVERTER 380**

12.1 Functions 380
12.2 Configuration 380
12.3 Registers 381
12.4 Operation 383
12.4.1 Operation in normal mode 383
12.4.2 Operation in real-time output mode 383
12.4.3 Cautions 384

CHAPTER 13 ASYNCHRONOUS SERIAL INTERFACE A (UARTA) 385

13.1 Features 385
13.2 Configuration 386
13.3 Registers 388
13.4 Interrupt Request Signals 394
13.5 Operation 395
13.5.1 Data format 395
13.5.2 UART transmission 397
13.5.3 Continuous transmission procedure 398
13.5.4 UART reception 400
13.5.5 Reception errors 401
13.5.6 Parity types and operations 403
13.5.7 Receive data noise filter 404
13.6 Dedicated Baud Rate Generator 405
13.7 Cautions 413

CHAPTER 14 3-WIRE VARIABLE-LENGTH SERIAL I/O (CSIB) 414

14.1 Mode Switching of CSIB and Other Serial Interfaces 414
14.1.1 CSIB0 and I^2C01 mode switching 414
14.2 Features 415
14.3 Configuration 416
14.4 Registers 418
<R> **14.5 Interrupt Request Signals 426**
14.6 Operation 427
<R> 14.6.1 Single transfer mode (master mode, transmission mode) 427
14.6.2 Single transfer mode (master mode, reception mode) 429
14.6.3 Single transfer mode (master mode, transmission/reception mode) 431
<R> 14.6.4 Single transfer mode (slave mode, transmission mode) 433

<R> 14.6.5 Single transfer mode (slave mode, reception mode) ... 435
<R> 14.6.6 Single transfer mode (slave mode, transmission/reception mode) ... 437
<R> 14.6.7 Continuous transfer mode (master mode, transmission mode) ... 439
14.6.8 Continuous transfer mode (master mode, reception mode) ... 441
14.6.9 Continuous transfer mode (master mode, transmission/reception mode) ... 444
<R> 14.6.10 Continuous transfer mode (slave mode, transmission mode) ... 448
14.6.11 Continuous transfer mode (slave mode, reception mode) ... 450
14.6.12 Continuous transfer mode (slave mode, transmission/reception mode) ... 453
14.6.13 Reception error ... 457
14.6.14 Clock timing ... 458
14.7 Output Pins ... 460
14.8 Baud Rate Generator ... 461
14.8.1 Baud rate generation ... 462
14.9 Cautions ... 463

CHAPTER 15 I^2C BUS ... 464

15.1 Mode Switching of I^2C Bus and Other Serial Interfaces ... 464
15.1.1 CSIB0 and I^2C01 mode switching ... 464
15.2 Features ... 465
15.3 Configuration ... 466
15.4 Registers ... 470
15.5 I^2C Bus Mode Functions ... 485
15.5.1 Pin configuration ... 485
15.6 I^2C Bus Definitions and Control Methods ... 486
15.6.1 Start condition ... 486
15.6.2 Addresses ... 487
15.6.3 Transfer direction specification ... 488
15.6.4 $\overline{ACK}$ ... 489
15.6.5 Stop condition ... 490
15.6.6 Wait state ... 491
15.6.7 Wait state cancellation method ... 493
15.7 I^2C Interrupt Request Signals (INTIICn) ... 494
15.7.1 Master device operation ... 495
15.7.2 Slave device operation (when receiving slave address (address match)) ... 498
15.7.3 Slave device operation (when receiving extension code) ... 502
15.7.4 Operation without communication ... 506
15.7.5 Arbitration loss operation (operation as slave after arbitration loss) ... 507
15.7.6 Operation when arbitration loss occurs (no communication after arbitration loss) ... 509
15.8 Interrupt Request Signal (INTIICn) Generation Timing and Wait Control ... 516
15.9 Address Match Detection Method ... 517
15.10 Error Detection ... 517
15.11 Extension Code ... 518
15.12 Arbitration ... 519
15.13 Wakeup Function ... 520
15.14 Communication Reservation ... 521
15.14.1 When communication reservation function is enabled (IICFn.IICRSVn bit = 0) ... 521
15.14.2 When communication reservation function is disabled (IICFn.IICRSVn bit = 1) ... 525

15.15 Cautions 526
15.16 Communication Operations 527
15.16.1 **Master operation in single master system** 528
15.16.2 Master operation in multimaster system 529
15.16.3 Slave operation 532
15.17 Timing of Data Communication 536

CHAPTER 16 IEBus CONTROLLER 543

16.1 Functions 543
16.1.1 Communication protocol of IEBus 543
16.1.2 Determination of bus mastership (arbitration) 544
16.1.3 Communication mode 544
16.1.4 Communication address 544
16.1.5 Broadcast communication 545
16.1.6 Transfer format of IEBus 545
16.1.7 Transfer data 555
16.1.8 Bit format 557
16.2 Configuration 558
16.3 Registers 560
16.4 Interrupt Operations of IEBus Controller 591
16.4.1 Interrupt control block 591
16.4.2 Example of identifying interrupt 594
16.4.3 Interrupt source list 597
16.4.4 Communication error source processing list 598
16.5 Interrupt Request Signal Generation Timing and Main CPU Processing 600
16.5.1 Master transmission 600
16.5.2 Master reception 602
16.5.3 Slave transmission 604
16.5.4 Slave reception 606
16.5.5 Interval of occurrence of interrupt request signal for IEBus control 608

<R> **CHAPTER 17 CAN CONTROLLER 612**

17.1 Overview 612
17.1.1 Features 612
17.1.2 Overview of functions 613
17.1.3 Configuration 614
17.2 CAN Protocol 615
17.2.1 Frame format 615
17.2.2 Frame types 616
17.2.3 Data frame and remote frame 616
17.2.4 Error frame 624
17.2.5 Overload frame 625
17.3 Functions 626
17.3.1 Determining bus priority 626
17.3.2 Bit stuffing 626
17.3.3 Multi masters 626
17.3.4 Multi cast 626

17.3.5 CAN sleep mode/CAN stop mode function ... 627
17.3.6 Error control function ... 627
17.3.7 Baud rate control function ... 632
17.4 Connection with Target System ... 636
17.5 Internal Registers of CAN Controller ... 637
17.5.1 CAN controller configuration ... 637
17.5.2 Register access type ... 638
17.5.3 Register bit configuration ... 655
17.6 Registers ... 659
17.7 Bit Set/Clear Function ... 692
17.8 CAN Controller Initialization ... 694
17.8.1 Initialization of CAN module ... 694
17.8.2 Initialization of message buffer ... 694
17.8.3 Redefinition of message buffer ... 694
17.8.4 Transition from initialization mode to operation mode ... 696
17.8.5 Resetting error counter C0ERC of CAN module ... 696
17.9 Message Reception ... 697
17.9.1 Message reception ... 697
17.9.2 Reading reception data ... 698
17.9.3 Receive history list function ... 699
17.9.4 Mask function ... 701
17.9.5 Multi buffer receive block function ... 703
17.9.6 Remote frame reception ... 704
17.10 Message Transmission ... 705
17.10.1 Message transmission ... 705
17.10.2 Transmit history list function ... 707
17.10.3 Automatic block transmission (ABT) ... 709
17.10.4 Transmission abort process ... 710
17.10.5 Remote frame transmission ... 711
17.11 Power Saving Modes ... 712
17.11.1 CAN sleep mode ... 712
17.11.2 CAN stop mode ... 714
17.11.3 Example of using power saving modes ... 715
17.12 Interrupt Function ... 716
17.13 Diagnosis Functions and Special Operational Modes ... 717
17.13.1 Receive-only mode ... 717
17.13.2 Single-shot mode ... 718
17.13.3 Self-test mode ... 719
17.13.4 Transmission/reception operation in each operation mode ... 720
17.14 Baud Rate Settings ... 721
17.14.1 Bit rate setting conditions ... 721
17.14.2 Representative examples of baud rate settings ... 725
17.15 Operation of CAN Controller ... 729

<R> **CHAPTER 18 DMA FUNCTION (DMA CONTROLLER) ... 755**

18.1 Features ... 755
18.2 Configuration ... 756

18.3 Registers 757
18.4 Transfer Targets 765
18.5 Transfer Modes 765
18.6 Transfer Types 766
18.7 DMA Channel Priorities 767
18.8 Time Related to DMA Transfer 767
18.9 DMA Transfer Start Factors 768
18.10 DMA Abort Factors 769
18.11 End of DMA Transfer 769
18.12 Operation Timing 769
18.13 Cautions 774

CHAPTER 19 INTERRUPT/EXCEPTION PROCESSING FUNCTION 779

19.1 Features 779
19.2 Non-Maskable Interrupts 783
19.2.1 Operation 785
19.2.2 Restoration 786
19.2.3 NP flag 787
19.3 Maskable Interrupts 788
19.3.1 Operation 788
19.3.2 Restoration 790
19.3.3 Priorities of maskable interrupts 791
19.3.4 Interrupt control register (xxICn) 795
19.3.5 Interrupt mask registers 0 to 3 (IMR0 to IMR3) 797
19.3.6 In-service priority register (ISPR) 799
19.3.7 ID flag 800
19.3.8 Watchdog timer mode register 2 (WDTM2) 800
19.4 Software Exception 801
19.4.1 Operation 801
19.4.2 Restoration 802
19.4.3 EP flag 803
19.5 Exception Trap 804
19.5.1 Illegal opcode definition 804
19.5.2 Debug trap 806
19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7) 808
19.6.1 Noise elimination 808
19.6.2 Edge detection 808
19.7 Interrupt Acknowledge Time of CPU 813
19.8 Periods in Which Interrupts Are Not Acknowledged by CPU 815
19.9 Cautions 815

CHAPTER 20 KEY INTERRUPT FUNCTION 816

20.1 Function 816
20.2 Register 817
20.3 Cautions 817

CHAPTER 21 STANDBY FUNCTION 818

21.1 Overview 818
21.2 Registers 820
21.3 HALT Mode 823
21.3.1 Setting and operation status 823
21.3.2 Releasing HALT mode 823
21.4 IDLE1 Mode 825
21.4.1 Setting and operation status 825
21.4.2 Releasing IDLE1 mode 825
21.5 IDLE2 Mode 827
21.5.1 Setting and operation status 827
21.5.2 Releasing IDLE2 mode 827
21.5.3 Securing setup time when releasing IDLE2 mode 829
21.6 STOP Mode 830
21.6.1 Setting and operation status 830
21.6.2 Releasing STOP mode 830
21.6.3 Securing oscillation stabilization time when releasing STOP mode 833
21.7 Subclock Operation Mode 834
21.7.1 Setting and operation status 834
21.7.2 Releasing subclock operation mode 834
21.8 Sub-IDLE Mode 836
21.8.1 Setting and operation status 836
21.8.2 Releasing sub-IDLE mode 836

CHAPTER 22 RESET FUNCTIONS 838

22.1 Overview 838
22.2 Registers to Check Reset Source 839
22.3 Operation 840
22.3.1 Reset operation via $\overline{\text{RESET}}$ pin 840
22.3.2 Reset operation by watchdog timer 2 (WDT2RES) 842
<R> 22.3.3 Reset operation by clock monitor (CLMRES) 844
22.3.4 Operation after reset release 846
22.3.5 Reset function operation flow 847
22.4 Valid/Invalid of Internal RAM Data 848
22.4.1 RAM retention voltage detection operation 849
22.4.2 Emulation function 850

CHAPTER 23 CLOCK MONITOR 851

23.1 Functions 851
23.2 Configuration 851
23.3 Register 852
23.4 Operation 853

CHAPTER 24 REGULATOR 856

24.1 Outline 856
24.2 Operation 857

CHAPTER 25 ROM CORRECTION FUNCTION 858

25.1 Overview 858
25.2 Registers 859
25.3 ROM Correction Operation and Program Flow 860
25.4 Cautions 862

CHAPTER 26 ELECTRICAL SPECIFICATIONS 863

26.1 Absolute Maximum Ratings 863
26.2 Capacitance 865
26.3 Operating Conditions 865
26.4 Oscillator Characteristics 866
26.4.1 Main clock oscillator characteristics 866
26.4.2 Subclock oscillator characteristics 869
26.4.3 PLL characteristics 870
26.4.4 Internal oscillator characteristics 870
26.5 Regulator Characteristics 870
26.6 DC Characteristics 871
26.6.1 I/O level 871
26.6.2 Supply current 873
26.7 Data Retention Characteristics 874
26.8 AC Characteristics 875
26.8.1 CLKOUT output timing 876
26.8.2 Bus timing 877
26.9 Basic Operation 890

CHAPTER 27 PACKAGE DRAWING 898

CHAPTER 28 RECOMMENDED SOLDERING CONDITIONS 900

APPENDIX A DEVELOPMENT TOOLS 901

A.1 Software Package 906
A.2 Language Processing Software 906
A.3 Control Software 906
A.4 Debugging Tools (Hardware) 907
A.4.1 When using in-circuit emulator IE-V850ES-G1 907
A.4.2 When using IECUBE QB-V850ESSX2 909
<R> A.4.3 When using on-chip debug emulator IE-V850E1-CD-NW 912
<R> A.4.4 When using MINICUBE QB-V850MINI 913
A.5 Debugging Tools (Software) 914
A.6 Embedded Software 915
A.7 Flash Memory Writing Tools 916

APPENDIX B MAJOR DIFFERENCES BETWEEN V850ES/SG1 (μPD703249Y, 703252Y) AND V850ES/SG1 (μPD703253Y) AND V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y) 917

APPENDIX C REGISTER INDEX 919

APPENDIX D INSTRUCTION SET LIST 929

D.1 Conventions 929
D.2 Instruction Set (in Alphabetical Order) 932

<R> APPENDIX E LIST OF CAUTIONS 939

<R> APPENDIX F REVISION HISTORY 972

F.1 Major Revisions in This Edition 972

CHAPTER 1 INTRODUCTION

The V850ES/SG1 is one of the products in the NEC Electronics V850 single-chip microcontrollers designed for low-power operation for real-time control applications.

1.1 General

The V850ES/SG1 is a 32-bit single-chip microcontroller that includes the V850ES CPU core and peripheral functions such as ROM/RAM, a timer/counter, serial interfaces, an A/D converter, and a D/A converter.

<R> The V850ES/SG1 is provided with IEBus™ (Inter Equipment Bus™) or CAN (Controller Area Network) as an automotive LAN.

In addition to high real-time response characteristics and 1-clock-pitch basic instructions, the V850ES/SG1 features multiply instructions, saturated operation instructions, bit manipulation instructions, etc., realized by a hardware multiplier, as optimum instructions for digital servo control applications. Moreover, as a real-time control system, the V850ES/SG1 enables an extremely high cost-performance for applications that require low power consumption, such as audio and car audio.

Table 1-1 lists the products of the V850ES/SG1.

Table 1-1. V850ES/SG1 Product List

Function / Part Number	ROM		RAM Size	I^2C	IEBus	CAN	Maskable Interrupts		Non-Maskable Interrupts
	Type	Size					External	Internal	
μPD703249Y	Mask ROM	256 KB	12 KB	On-chip	None	None	8	31	2
μPD703252Y					On-chip			35	
<R> μPD703253Y		128 KB	8 KB		None	On-chip		42	

The V850ES/SG2 can be used as the flash memory version for developing the V850ES/SG1 software. Use the V850ES/SG2 (flash memory version) corresponding to the V850ES/SG1 (mask ROM version) as shown below.

- For the flash memory version of the μPD703249Y: Use the μPD70F3261Y.
- For the flash memory version of the μPD703252Y: Use the μPD70F3271Y.
- <R> For the flash memory version of the μPD703253Y: Use the μPD70F3281Y.

<R> The following table lists the differences between the V850ES/SG1 and V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y).

<R> **Table 1-2. Differences Between V850ES/SG1 and V850ES/SG2 (μPD70F3261Y, μPD70F3271Y, μPD70F3281Y)**

Function		μPD703249Y	μPD703252Y	μPD703253Y	μPD70F3261Y	μPD70F3271Y	μPD70F3281Y
		V850ES/SG1			V850ES/SG2		
ROM		256 KB (mask ROM)		128 KB (mask ROM)	384 KB (flash memory)		
RAM		12 KB		8 KB	32 KB		
Timer	TMP	5 ch		6 ch			
	TMQ	None			1 ch		
	TMM	1 ch					
	Watch timer	1 ch					
	WDT2	1 ch					
RTO		None			Provided (6 bits × 1 ch)		
Serial interface	UARTA	2 ch			3 ch		
	I^2C	2 ch			3 ch		
	CSIB	3 ch			5 ch		
IEBus controller		None	Provided (1 ch)	None		Provided (1 ch)	None
CAN controller		None		Provided (1 ch)	None		Provided (1 ch)
	Time stamp function	None					Provided
Internal maskable interrupt		31 sources	35 sources	42 sources	47 sources	51 sources	
D/A converter		None		Provided (8 bit resolution × 2 ch)			
DMA controller		None		Provided (4 ch)			
CRC function		None			Provided		
On-chip debug function		None			Provided		
Low-voltage detector		None		None (RAM retention voltage detection function is provided)	Provided		

Remark For details on the flash memory versions, refer to the **V850ES/SG2, V850ES/SG2-H Hardware User's Manual (U16541E)**.

1.2 Features

- Minimum instruction execution time: 50 ns (operating with main clock (fxx) of 20 MHz)
- General-purpose registers: 32 bits × 32 registers
- CPU features:
  - Signed multiplication (16 × 16 → 32): 1 to 2 clocks
  - Signed multiplication (32 × 32 → 64): 1 to 5 clocks
  - Saturated operations (overflow and underflow detection functions included)
  - 32-bit shift instruction: 1 clock
  - Bit manipulation instructions
  - Load/store instructions with long/short format
- Memory space: 64 MB of linear address space (for programs and data)
  - External expansion: Up to 16 MB
  - <R> • Internal memory:
    - RAM: 8/12 KB (see **Table 1-1**)
    - Mask ROM: 128/256 KB (see **Table 1-1**)
  - • External bus interface:
    - Separate bus/multiplexed bus output selectable
    - 8/16 bit data bus sizing function
    - Wait function
      - • Programmable wait function
      - • External wait function
    - Idle state function
    - Bus hold function
- Interrupts and exceptions:
  - Non-maskable interrupts: 2 sources
  - <R> Maskable interrupts: 39/43/50 sources (see **Table 1-1**)
  - Software exceptions: 32 sources
  - Exception trap: 2 sources
- I/O lines:
  - I/O ports: 84
- Timer function:
  - 16-bit interval timer M (TMM): 1 channel
  - <R> 16-bit timer/event counter P (TMP): 5/6 channels (see **Table 1-2**)
  - Watch timer: 1 channel
  - Watchdog timer: 1 channel
- Serial interface:
  - Asynchronous serial interface A (UARTA)
  - 3-wire variable-length serial interface B (CSIB)
  - I^2C bus interface (I^2C)
    - CSIB/I^2C: 1 channel
    - CSIB: 2 channels
    - UART: 2 channels
    - I^2C: 1 channel
- IEBus controller: 1 channel (μPD703252Y only)
- <R> CAN controller: 1 channel (μPD703253Y only)
- A/D converter: 10-bit resolution: 12 channels
- <R> D/A converter: 8-bit resolution: 2 channels (μPD703253Y only)
- <R> DMA controller: 4 channels (μPD703253Y only)
- ROM correction: 4 correction addresses specifiable
- Clock generator: During main clock or subclock operation
  - 7-level CPU clock (fxx, fxx/2, fxx/4, fxx/8, fxx/16, fxx/32, fxT)
  - Clock-through mode/PLL mode selectable
- Internal oscillation clock: 200 kHz (TYP.)
- Power-save functions: HALT/IDLE1/IDLE2/STOP/subclock/sub-IDLE mode

○ Package: 100-pin plastic QFP (14 × 20) (μPD703249Y, 703252Y only (see **1.4 Ordering Information**))
100-pin plastic LQFP (fine pitch) (14 × 14)

1.3 Application Fields

Audio, car audio, consumer devices

<R>

1.4 Ordering Information

Order Number	Package	Internal ROM
μPD703249YGF-xxx-JBT-A	100-pin plastic QFP (14 × 20)	256 KB (mask ROM)
μPD703249YGC-xxx-8EA-A	100-pin plastic LQFP (fine pitch) (14 × 14)	256 KB (mask ROM)
μPD703252YGF-xxx-JBT-A	100-pin plastic QFP (14 × 20)	256 KB (mask ROM)
μPD703252YGC-xxx-8EA-A	100-pin plastic LQFP (fine pitch) (14 × 14)	256 KB (mask ROM)
μPD703253YGC-xxx-8EA-A	100-pin plastic LQFP (fine pitch) (14 × 14)	128 KB (mask ROM)

Remarks 1. xxx indicates the ROM code suffix.

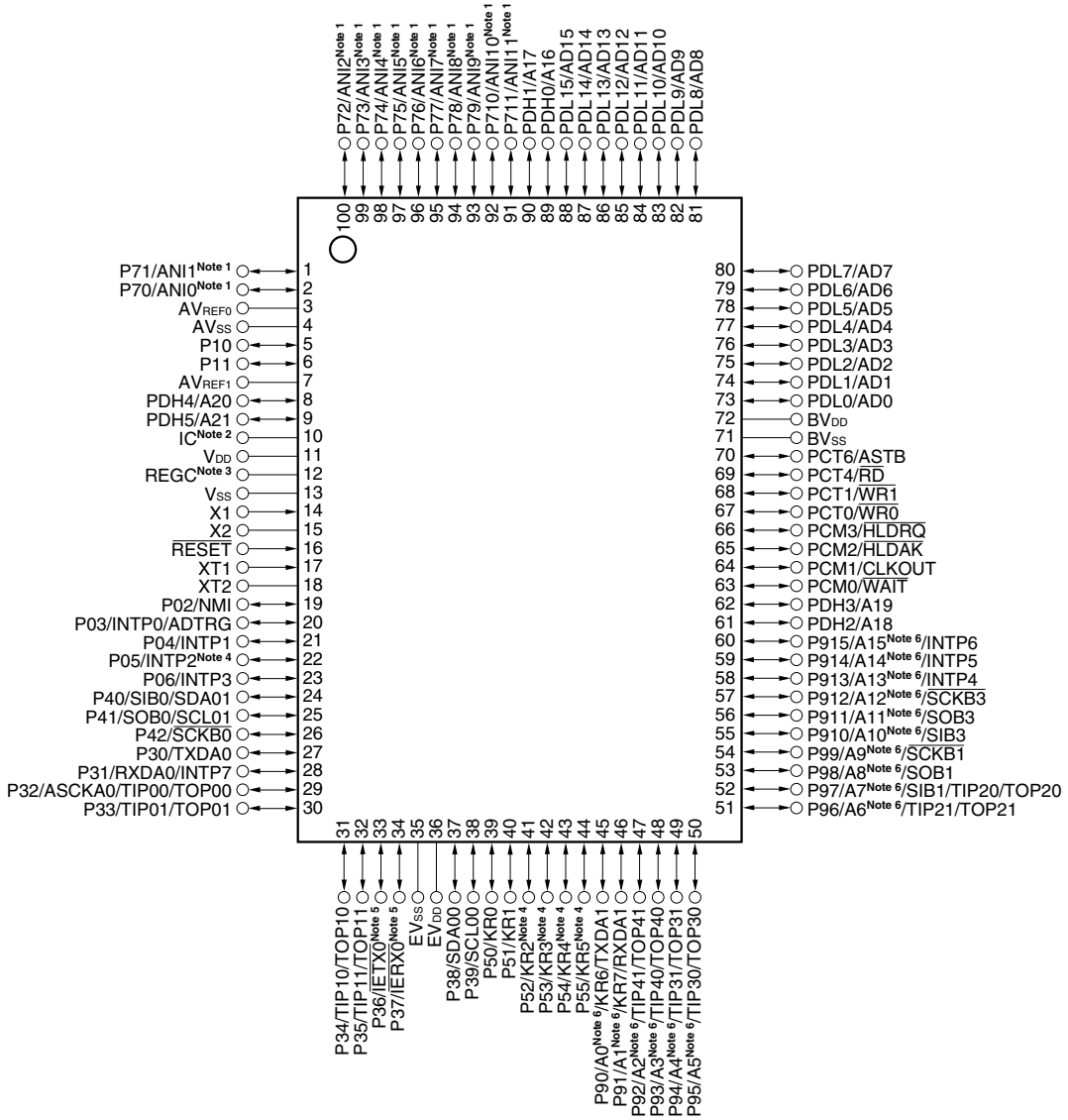
2. The V850ES/SG1 microcontrollers are lead-free products.

1.5 Pin Configuration (Top View)

- 100-pin plastic QFP (14 × 20)

<R> μPD703249YGF-xxx-JBT-A

<R> μPD703252YGF-xxx-JBT-A



<R> **Notes 1.** To use port 7 (P70/ANI0 to P711/ANI11) as A/D converter function pins and port I/O pins in mix, be sure to observe usage cautions (refer to **11.6 (4) Alternate I/O**).

2. Directly connect this pin to VSS.

3. Connect the REGC pin to VSS via a 4.7 μF capacitor.

4. This pin functions alternately as the on-chip debug function pin in the flash memory versions (V850ES/SG2 (μPD70F3261Y, 70F3271Y)).

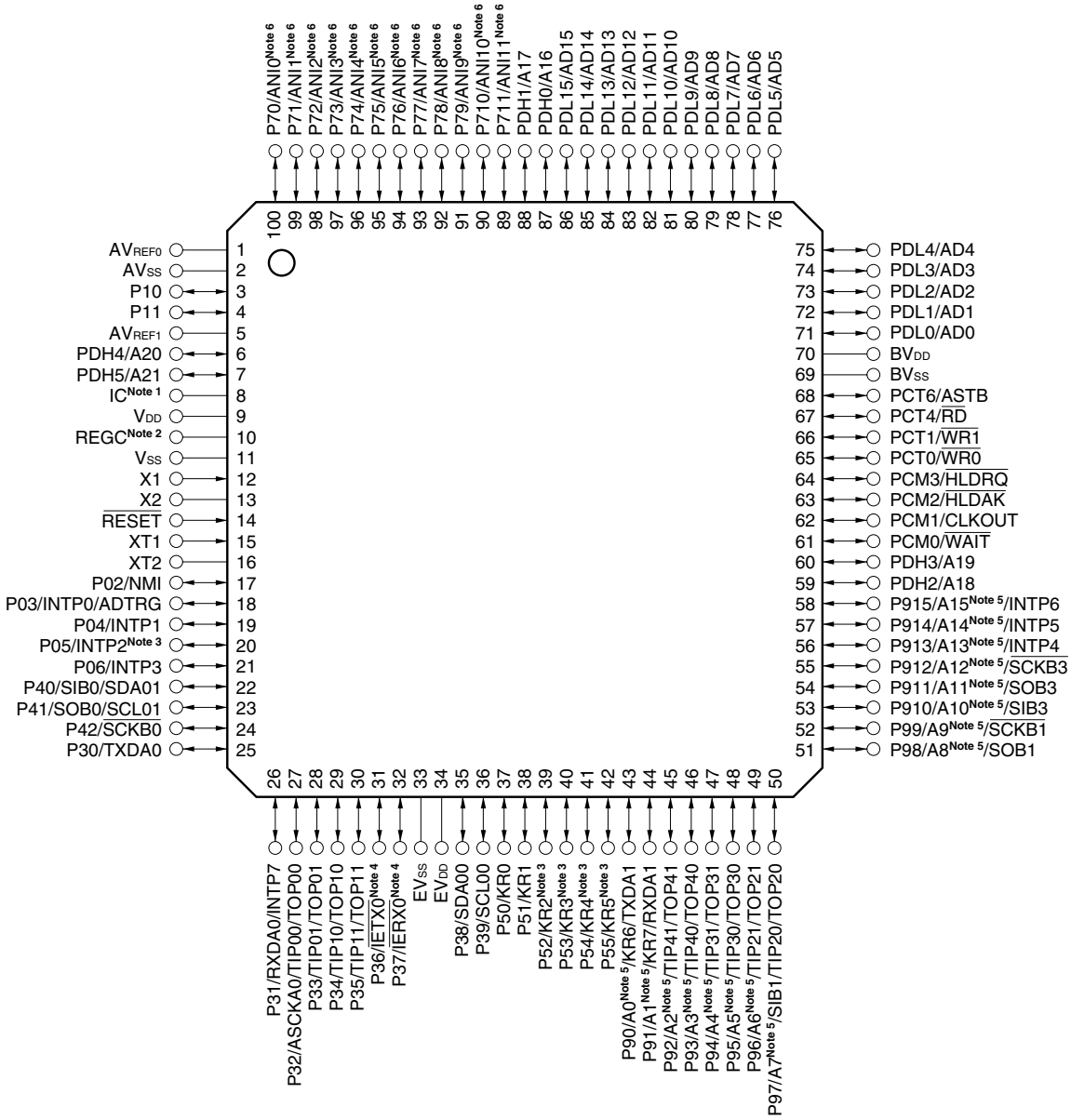
5. $\overline{\text{IETX0}}$ and $\overline{\text{IERX0}}$ are valid only in the μPD703252Y.

<R> **6.** Port 9 cannot be used as port pins or other alternate-function pins when the A0 to A15 pins are used in the separate bus mode.

- 100-pin plastic LQFP (fine pitch) (14 × 14) (1/2)

<R> μPD703249YGC-xxx-8EA-A

<R> μPD703252YGC-xxx-8EA-A



Notes 1. Directly connect this pin to V_{SS}.

2. Connect the REGC pin to V_{SS} via a 4.7 μF capacitor.

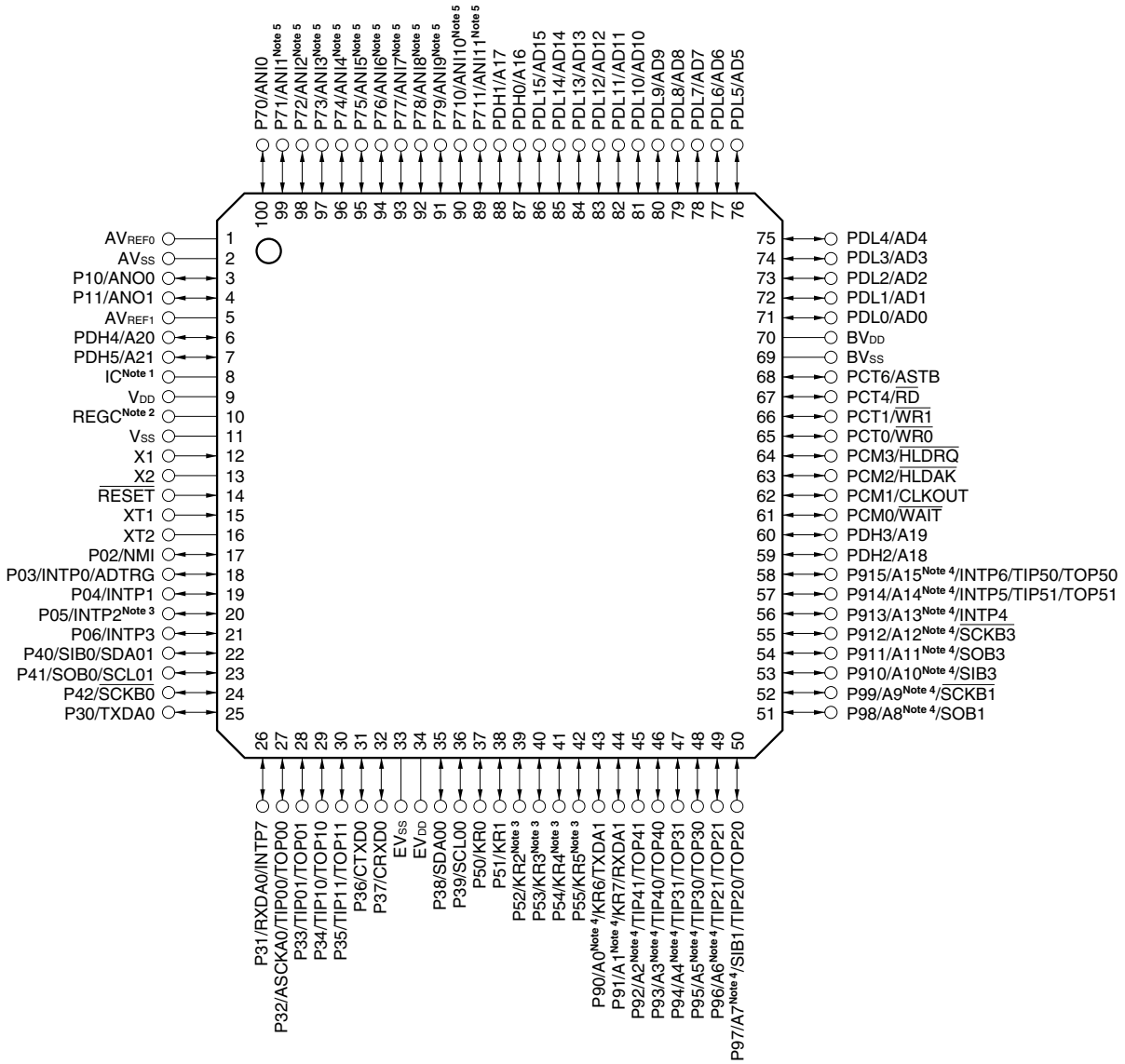
3. This pin functions alternately as the on-chip debug function pin in the flash memory versions (V850ES/SG2 (μPD70F3261Y, μPD70F3271Y)).

4. $\overline{\text{IETX0}}$ and $\overline{\text{IERX0}}$ are valid only in the μPD703252Y.

<R> **5.** Port 9 cannot be used as port pins or other alternate-function pins when the A0 to A15 pins are used in the separate bus mode.

<R> **6.** To use port 7 (P70/ANI0 to P711/ANI11) as A/D converter function pins and port I/O pins in mix, be sure to observe usage cautions (refer to **11.6 (4) Alternate I/O**).

<R> • 100-pin plastic LQFP (fine pitch) (14 × 14) (2/2)
μPD703253YGC-xxx-8EA-A



Notes 1. Directly connect this pin to VSS.

2. Connect the REGC pin to VSS via a 4.7 μF capacitor.

3. This pin functions alternately as the on-chip debug function pin in the flash memory version (V850ES/SG2 (μPD70F3281Y)).

4. Port 9 cannot be used as port pins or other alternate-function pins when the A0 to A15 pins are used in the separate bus mode.

5. To use port 7 (P70/ANI0 to P711/ANI11) as A/D converter function pins and port I/O pins in mix, be sure to observe usage cautions (refer to **11.6 (4) Alternate I/O**).

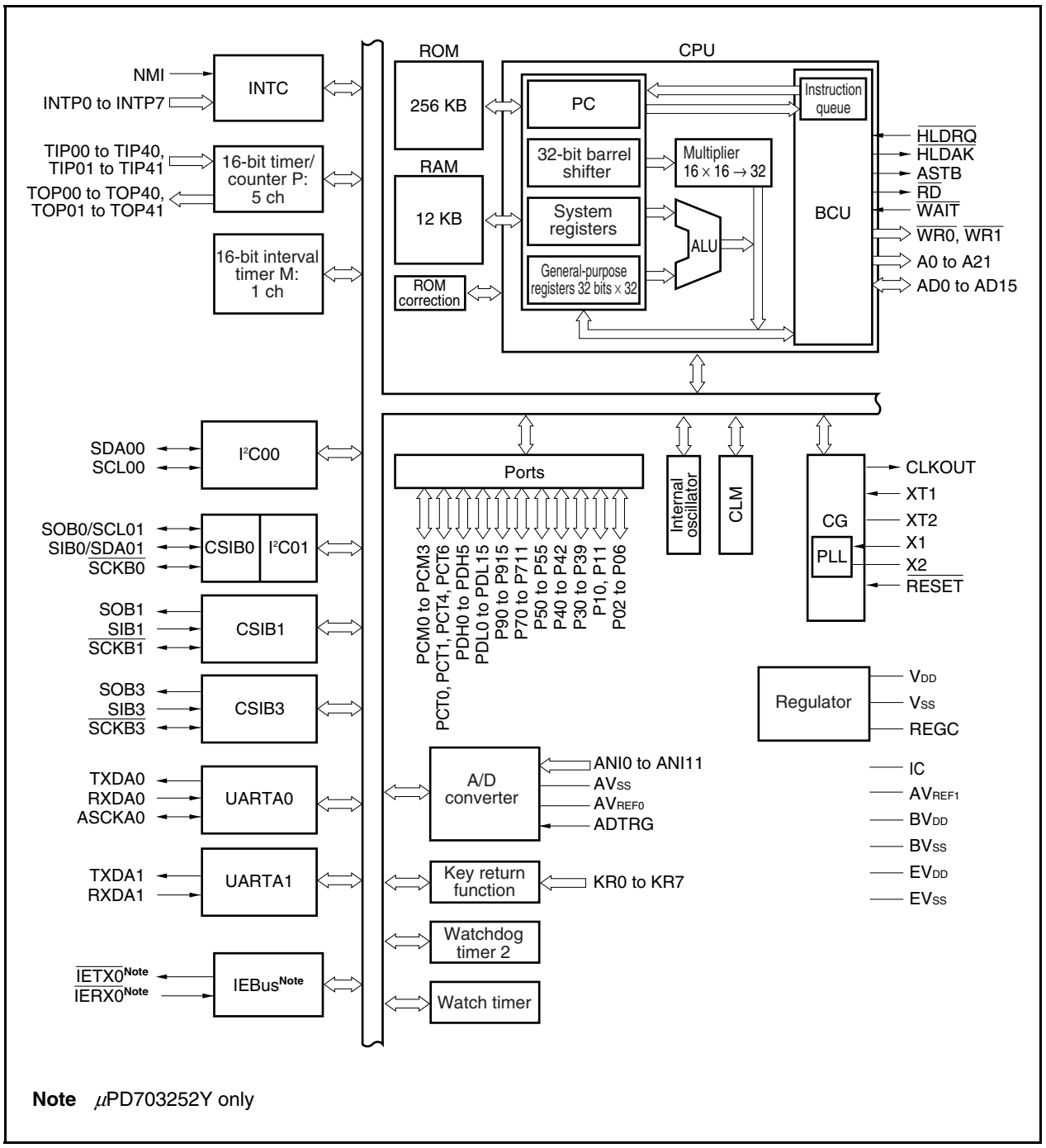
Pin names

A0 to A21:	Address bus	PDH0 to PDH5:	Port DH
AD0 to AD15:	Address/data bus	PDL0 to PDL15:	Port DL
ADTRG:	A/D trigger input	$\overline{RD}$:	Read strobe
ANI0 to ANI11:	Analog input	REGC:	Regulator control
ANO0, ANO1:	Analog output	$\overline{RESET}$:	Reset
ASCKA0:	Asynchronous serial clock	RXDA0, RXDA1:	Receive data
ASTB:	Address strobe	$\overline{SCKB0}$, $\overline{SCKB1}$,	
AV_{REF0}, AV_{REF1}:	Analog reference voltage	$\overline{SCKB3}$:	Serial clock
AV_{SS}:	Analog V_{SS}	SCL00, SCL01:	Serial clock
BV_{DD}:	Power supply for bus interface	SDA00, SDA01:	Serial data
BV_{SS}:	Ground for bus interface	SIB0, SIB1,	
CLKOUT:	Clock output	SIB3:	Serial input
CRXD0:	CAN receive data	SOB0, SOB1,	
CTXD0:	CAN transmit data	SOB3:	Serial output
EV_{DD}:	Power supply for port	TIP00, TIP01,	
EV_{SS}:	Ground for port	TIP10, TIP11,	
$\overline{HLDAK}$:	Hold acknowledge	TIP20, TIP21,	
$\overline{HLDRQ}$:	Hold request	TIP30, TIP31,	
IC:	Internally connected	TIP40, TIP41,	
$\overline{IERX0}$:	IEBus receive data	TIP50, TIP51:	Timer input
$\overline{IETX0}$:	IEBus transmit data	TOP00, TOP01,	
INTP0 to INTP7:	External interrupt input	TOP10, TOP11,	
KR0 to KR7:	Key return	TOP20, TOP21,	
NMI:	Non-maskable interrupt request	TOP30, TOP31,	
P02 to P06:	Port 0	TOP40, TOP41,	
P10, P11:	Port 1	TOP50, TOP51:	Timer output
P30 to P39:	Port 3	TXDA0, TXDA1:	Transmit data
P40 to P42:	Port 4	V_{DD}:	Power supply
P50 to P55:	Port 5	V_{SS}:	Ground
P70 to P711:	Port 7	$\overline{WAIT}$:	Wait
P90 to P915:	Port 9	$\overline{WR0}$:	Lower byte write strobe
PCM0 to PCM3:	Port CM	$\overline{WR1}$:	Upper byte write strobe
PCT0, PCT1,		X1, X2:	Crystal for main clock
PCT4, PCT6:	Port CT	XT1, XT2:	Crystal for subclock

1.6 Function Block Configuration

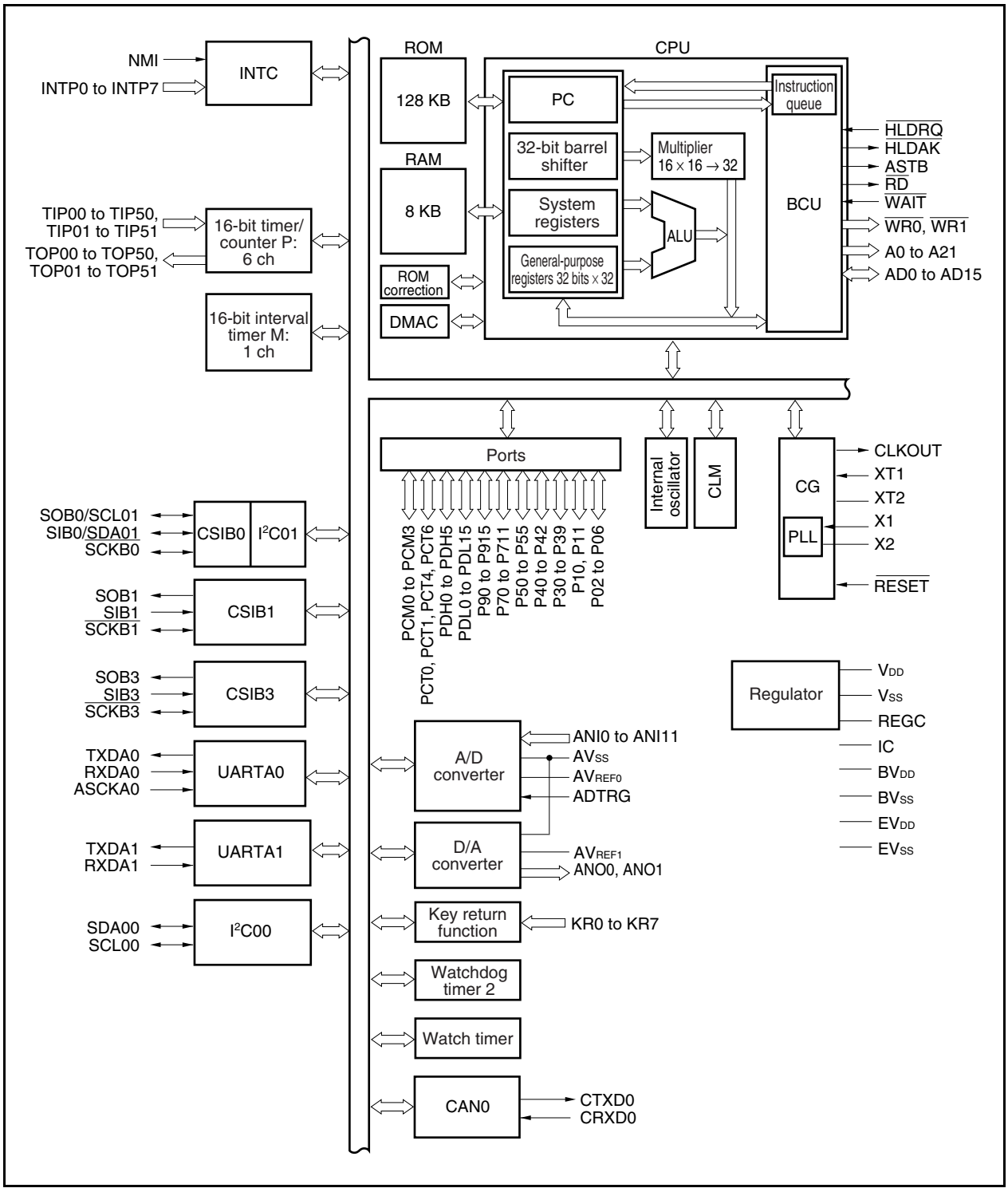
1.6.1 Internal block diagram

(1) μPD703249Y, 703252Y



Note μPD703252Y only

<R> **(2) μPD703253Y**



1.6.2 Internal units

(1) CPU

The CPU uses five-stage pipeline control to enable single-clock execution of address calculations, arithmetic logic operations, data transfers, and almost all other instruction processing.

Other dedicated on-chip hardware, such as a multiplier (16 bits × 16 bits → 32 bits) and a barrel shifter (32 bits) contribute to faster complex processing.

(2) Bus control unit (BCU)

The BCU starts a required external bus cycle based on the physical address obtained by the CPU. When an instruction is fetched from external memory space and the CPU does not send a bus cycle start request, the BCU generates a prefetch address and prefetches the instruction code. The prefetched instruction code is stored in an instruction queue.

(3) ROM

<R> This is a 256/128 KB mask ROM mapped to addresses 0000000H to 003FFFFH/0000000H to 001FFFFH. It can be accessed from the CPU in one clock during instruction fetch.

(4) RAM

<R> This is an 12/8 KB RAM mapped to addresses 3FFC000H to 3FFEFFFH/3FFD000H to 3FFEFFFH. It can be accessed from the CPU in one clock during data access.

(5) Interrupt controller (INTC)

This controller handles hardware interrupt requests (NMI, INTP0 to INTP7) from on-chip peripheral hardware and external hardware. Eight levels of interrupt priorities can be specified for these interrupt requests, and multiplexed servicing control can be performed.

(6) Clock generator (CG)

A main clock oscillator and subclock oscillator are provided and generate the main clock oscillation frequency (f_X) and subclock frequency (f_{XT}), respectively. There are two modes: In the clock-through mode, f_X is used as the main clock frequency (f_{XX}) as is. In the PLL mode, f_X is used multiplied by 4 or 8.

The CPU clock frequency (f_{CPU}) can be selected from among f_{XX}, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, and f_{XT}.

(7) Internal oscillator

An internal oscillator is provided on chip. The oscillation frequency is 200 kHz (TYP). The internal oscillator supplies the clock for watchdog timer 2 and timer M.

(8) Timer/counter

<R> Six/five-channel 16-bit timer/event counter P (TMP) and one-channel 16-bit interval timer M (TMM) are provided on chip.

(9) Watch timer

This timer counts the reference time period (0.5 s) for counting the clock (the 32.768 kHz subclock or the 32.768 kHz clock f_{BRG} from prescaler 3). The watch timer can also be used as an interval timer for the main clock.

(10) Watchdog timer 2

A watchdog timer is provided on chip to detect inadvertent program loops, system abnormalities, etc.
The internal oscillation clock, the main clock, or the subclock can be selected as the source clock.
Watchdog timer 2 generates a non-maskable interrupt request signal (INTWDT2) or a system reset signal (WDT2RES) after an overflow occurs.

(11) Serial interface

The V850ES/SG1 includes three kinds of serial interfaces: asynchronous serial interface A (UARTA), 3-wire variable-length serial interface B (CSIB), and an I^2C bus interface (I^2C).
In the case of UARTA, data is transferred via the TXDA0, TXDA1, RXDA0, and RXDA1 pins.
In the case of CSIB, data is transferred via the SOB0, SOB1, SOB3, SIB0, SIB1, SIB3, $\overline{\text{SCKB0}}$, $\overline{\text{SCKB1}}$, and $\overline{\text{SCKB3}}$ pins.
In the case of I^2C, data is transferred via the SDA00, SDA01, SCL00, and SCL01 pins.

(12) IEBus controller

The IEBus controller is a small-scale digital data transmission system for transferring data between units.
The IEBus controller is provided only in the μPD703252Y (see **Table 1-1**).

<R> **(13) CAN controller**

The CAN controller is a small-scale digital data transmission system for transferring data between units.
The CAN controller is provided only in the μPD703253Y (see **Table 1-1**).

(14) A/D converter

This 10-bit A/D converter includes 12 analog input pins. Conversion is performed using the successive approximation method.

<R> **(15) D/A converter**

A two-channel, 8-bit-resolution D/A converter that uses the R-2R ladder method is provided on chip.
The D/A converter is provided only in the μPD703253Y (see **Table 1-2**).

<R> **(16) DMA controller**

A 4-channel DMA controller is provided on chip. This controller transfers data between the internal RAM and on-chip peripheral I/O devices in response to interrupt requests sent by on-chip peripheral I/O.
The DMA controller is provided only in the μPD703253Y (see **Table 1-2**).

(17) ROM correction

A ROM correction function that replaces part of a program in the mask ROM with a program in the internal RAM is provided. Up to four correction addresses can be specified.

(18) Key interrupt function

A key interrupt request signal (INTKR) can be generated by inputting a falling edge to the key input pins (8 channels).

(19) On-chip debug function

The V850ES/SG1 is not provided with an on-chip debug function. However, an on-chip debug function can
<R> be used when the V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y) is used as the flash memory version for developing software. For details, refer to the **V850ES/SG2, V850ES/SG2-H Hardware User's Manual (U16541E)**.

(20) Ports

The following general-purpose port functions and control pin functions are available.

Port	I/O	Alternate Function
P0	5-bit I/O	NMI, external interrupt, A/D converter trigger
<R> P1	2-bit I/O	D/A converter analog output[Note 1]
<R> P3	10-bit I/O	External interrupt, serial interface, timer I/O, IEBus data I/O[Note 2], CAN data I/O[Note 1]
P4	3-bit I/O	Serial interface
P5	6-bit I/O	Key interrupt input
P7	12-bit I/O	A/D converter analog input
P9	16-bit I/O	External address bus, serial interface, key interrupt input, timer I/O, external interrupt
PCM	4-bit I/O	External control signal
PCT	4-bit I/O	External control signal
PDH	6-bit I/O	External address bus
PDL	16-bit I/O	External address/data bus

<R> **Notes 1.** μPD703253Y only

<R> **2.** μPD703252Y only

CHAPTER 2 PIN FUNCTIONS

2.1 List of Pin Functions

The names and functions of the pins in the V850ES/SG1 are described below.

There are four types of pin I/O buffer power supplies: AV_{REF0}, AV_{REF1}, BV_{DD}, and EV_{DD}. The relationship between these power supplies and the pins is described below.

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

Power Supply	Corresponding Pins
AV_{REF0}	Port 7
AV_{REF1}	Port 1
BV_{DD}	Ports CM, CT, DH (bits 0 to 3), DL
EV_{DD}	$\overline{RESET}$, ports 0, 3 to 5, 9, DH (bits 4, 5)

(1) Port pins

(1/4)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
P02	19	17	I/O	Port 0 5-bit I/O port Input/output can be specified in 1-bit units. N-ch open-drain output can be specified in 1-bit units. 5 V tolerant.	NMI
P03	20	18			INTP0/ADTRG
P04	21	19			INTP1
P05[Note 1]	22	20			INTP2
P06	23	21			INTP3
P10	5	3	I/O	Port 1 2-bit I/O port Input/output can be specified in 1-bit units.	–
<R>	–	3			ANO0[Note 2]
P11	6	4			–
<R>	–	4			ANO1[Note 2]
P30	27	25	I/O	Port 3 10-bit I/O port Input/output can be specified in 1-bit units. N-ch open-drain output can be specified in 1-bit units. 5 V tolerant.	TXDA0
P31	28	26			RXDA0/INTP7
P32	29	27			ASCKA0/TIP00/TOP00
P33	30	28			TIP01/TOP01
P34	31	29			TIP10/TOP10
P35	32	30			TIP11/TOP11

Notes 1. A pull-down resistor is incorporated. It can be disconnected by clearing the OCDM.OCDM0 bit to 0.
<R> In the flash memory versions (V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)), this pin alternately functions as the on-chip debug pin.

<R> **2.** μPD703253Y only

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(2/4)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
P36	33	31	I/O	Port 3 10-bit I/O port Input/output can be specified in 1-bit units. N-ch open-drain output can be specified in 1-bit units. 5 V tolerant.	$\overline{\text{IETX0}}$ Note 1
<R>	–	31			CTXD0 Note 2
P37	34	32			$\overline{\text{IERX0}}$ Note 1
<R>	–	32			CRXD0 Note 2
P38	37	35			SDA00
P39	38	36			SCL00
P40	24	22	I/O	Port 4 3-bit I/O port Input/output can be specified in 1-bit units. N-ch open-drain output can be specified in 1-bit units. 5 V tolerant.	SIB0/SDA01
P41	25	23			SOB0/SCL01
P42	26	24			$\overline{\text{SCKB0}}$
P50	39	37	I/O	Port 5 6-bit I/O port Input/output can be specified in 1-bit units. N-ch open-drain output can be specified in 1-bit units. 5 V tolerant.	KR0
P51	40	38			KR1
P52 Note 3	41	39			KR2
P53 Note 3	42	40			KR3
P54 Note 3	43	41			KR4
P55 Note 3	44	42			KR5
P70	2	100	I/O	Port 7 12-bit I/O port Input/output can be specified in 1-bit units.	ANI0
P71	1	99			ANI1
P72	100	98			ANI2
P73	99	97			ANI3
P74	98	96			ANI4
P75	97	95			ANI5
P76	96	94			ANI6
P77	95	93			ANI7
P78	94	92			ANI8
P79	93	91			ANI9
P710	92	90			ANI10
P711	91	89			ANI11

Notes 1. μPD703252Y only

<R> **2.** μPD703253Y only

<R> **3.** In the flash memory versions (V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)), this pin alternately functions as the on-chip debug pin.

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(3/4)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
P90	45	43	I/O	Port 9	A0/KR6/TXDA1
P91	46	44		16-bit I/O port	A1/KR7/RXDA1
P92	47	45		Input/output can be specified in 1-bit units.	A2/TIP41/TOP41
P93	48	46		N-ch open-drain output can be specified in 1-bit units.	A3/TIP40/TOP40
P94	49	47		5 V tolerant.	A4/TIP31/TOP31
P95	50	48			A5/TIP30/TOP30
P96	51	49			A6/TIP21/TOP21
P97	52	50			A7/SIB1/TIP20/TOP20
P98	53	51			A8/SOB1
P99	54	52			A9/$\overline{\text{SCKB1}}$
P910	55	53			A10/SIB3
P911	56	54			A11/SOB3
P912	57	55			A12/$\overline{\text{SCKB3}}$
P913	58	56			A13/INTP4
P914	59	57			A14/INTP5
<R>	–	57			A14/INTP5/TIP51Note/TOP51Note
P915	60	58			A15/INTP6
<R>	–	58			A15/INTP6/TIP50Note/TOP50Note
PCM0	63	61	I/O	Port CM	$\overline{\text{WAIT}}$
PCM1	64	62		4-bit I/O port	CLKOUT
PCM2	65	63		Input/output can be specified in 1-bit units.	$\overline{\text{HLDAK}}$
PCM3	66	64			$\overline{\text{HLDRQ}}$
PCT0	67	65	I/O	Port CT	$\overline{\text{WR0}}$
PCT1	68	66		4-bit I/O port	$\overline{\text{WR1}}$
PCT4	69	67		Input/output can be specified in 1-bit units.	$\overline{\text{RD}}$
PCT6	70	68			ASTB
PDH0	89	87	I/O	Port DH	A16
PDH1	90	88		6-bit I/O port	A17
PDH2	61	59		Input/output can be specified in 1-bit units.	A18
PDH3	62	60			A19
PDH4	8	6			A20
PDH5	9	7			A21

<R> **Note** μPD703253Y only

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(4/4)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
PDL0	73	71	I/O	Port DL	AD0
PDL1	74	72		16-bit I/O port	AD1
PDL2	75	73		Input/output can be specified in 1-bit units.	AD2
PDL3	76	74			AD3
PDL4	77	75			AD4
PDL5	78	76			AD5
PDL6	79	77			AD6
PDL7	80	78			AD7
PDL8	81	79			AD8
PDL9	82	80			AD9
PDL10	83	81			AD10
PDL11	84	82			AD11
PDL12	85	83			AD12
PDL13	86	84			AD13
PDL14	87	85			AD14
PDL15	88	86			AD15

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(2) Non-port pins

(1/5)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
<R> A0	45	43	Output	Address bus for external memory (when using separate bus) Port 9 cannot be used as port pins or other alternate function pins when the A0 to A15 pins are used in the separate bus mode. N-ch open-drain output selectable. 5 V tolerant.	P90/KR6/TXDA1
<R> A1	46	44			P91/KR7/RXDA1
<R> A2	47	45			P92/TIP41/TOP41
<R> A3	48	46			P93/TIP40/TOP40
<R> A4	49	47			P94/TIP31/TOP31
<R> A5	50	48			P95/TIP30/TOP30
<R> A6	51	49			P96/TIP21/TOP21
<R> A7	52	50			P97/SIB1/TIP20/TOP20
<R> A8	53	51			P98/SOB1
<R> A9	54	52			P99/$\overline{\text{SCKB1}}$
<R> A10	55	53			P910/SIB3
<R> A11	56	54			P911/SOB3
<R> A12	57	55			P912/$\overline{\text{SCKB3}}$
<R> A13	58	56			P913/INTP4
<R> A14	59	57			P914/INTP5
<R>	–	57			P914/INTP5/TIP51[Note]/TOP51[Note]
<R> A15	60	58			P915/INTP6
<R>	–	58			P915/INTP6/TIP50[Note]/TOP50[Note]
A16	89	87	Output	Address bus for external memory	PDH0
A17	90	88			PDH1
A18	61	59			PDH2
A19	62	60			PDH3
A20	8	6			PDH4
A21	9	7			PDH5
AD0	73	71	I/O	Address bus/data bus for external memory	PDL0
AD1	74	72			PDL1
AD2	75	73			PDL2
AD3	76	74			PDL3
AD4	77	75			PDL4
AD5	78	76			PDL5
AD6	79	77			PDL6
AD7	80	78			PDL7
AD8	81	79			PDL8
AD9	82	80			PDL9
AD10	83	81			PDL10

<R> **Note** μPD703253Y only

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(2/5)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
AD11	84	82	I/O	Address bus/data bus for external memory	PDL11
AD12	85	83			PDL12
AD13	86	84			PDL13
AD14	87	85			PDL14
AD15	88	86			PDL15
ADTRG	20	18	Input	A/D converter external trigger input. 5 V tolerant.	P03/INTP0
ANI0	2	100	Input	Analog voltage input for A/D converter	P70
ANI1	1	99			P71
ANI2	100	98			P72
ANI3	99	97			P73
ANI4	98	96			P74
ANI5	97	95			P75
ANI6	96	94			P76
ANI7	95	93			P77
ANI8	94	92			P78
ANI9	93	91			P79
ANI10	92	90			P710
ANI11	91	89			P711
<R> ANO0[Note]	–	3	Output	Analog voltage output for D/A converter	P10
<R> ANO1[Note]	–	4			P11
ASCKA0	29	27	Input	UARTA0 baud rate clock input. 5 V tolerant.	P32/TIP00/TOP00
ASTB	70	68	Output	Address strobe signal output for external memory	PCT6
AV_{REF0}	3	1	–	Reference voltage input for A/D converter/positive power supply for port 7	–
<R> AV_{REF1}	7	5		Reference voltage input for D/A converter[Note]/positive power supply for port 1	–
<R> AV_{SS}	4	2	–	Ground potential for A/D and D/A[Note] converters (same potential as V_{SS})	–
BV_{DD}	72	70	–	Positive power supply pin for bus interface and alternate-function ports	–
BV_{SS}	71	69	–	Ground potential for bus interface and alternate-function ports	–
CLKOUT	64	62	Output	Internal system clock output	PCM1
<R> CRXD0[Note]	–	32	Input	CAN receive data input. 5 V tolerant.	P37
<R> CTXD0[Note]	–	31	Output	CAN transmit data output. N-ch open-drain output selectable. 5 V tolerant.	P36
EV_{DD}	36	34	–	Positive power supply for external (same potential as V_{DD})	–
EV_{SS}	35	33	–	Ground potential for external (same potential as V_{SS})	–

<R> **Note** μPD703253Y only

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(3/5)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
$\overline{\text{HLDAK}}$	65	63	Output	Bus hold acknowledge output	PCM2
$\overline{\text{HLDRQ}}$	66	64	Input	Bus hold request input	PCM3
IC	10	8	–	Internally connected	–
$\overline{\text{IERX0}}$ Note 1	34	32	Input	IEBus receive data input. 5 V tolerant.	P37
$\overline{\text{IETX0}}$ Note 1	33	31	Output	IEBus transmit data output. N-ch open-drain output selectable. 5 V tolerant.	P36
INTP0	20	18	Input	External interrupt request input (maskable, analog noise elimination). Analog noise elimination or digital noise elimination selectable for INTP3 pin. 5 V tolerant.	P03/ADTRG
INTP1	21	19			P04
INTP2	22	20			P05
INTP3	23	21			P06
INTP4	58	56			P913/A13
INTP5	59	57			P914/A14
<R>	–	57			P914/A14/TIP51 Note 2/TOP51 Note 2
INTP6	60	58			P915/A15
<R>	–	58			P915/A15/TIP50 Note 2/TOP50 Note 2
INTP7	28	26			P31/RXDA0
KR0 Note 3	39	37	Input	Key interrupt input (on-chip analog noise eliminator). 5 V tolerant.	P50
KR1 Note 3	40	38			P51
KR2 Note 3	41	39			P52
KR3 Note 3	42	40			P53
KR4 Note 3	43	41			P54
KR5 Note 3	44	42			P55
KR6 Note 3	45	43			P90/A0/TXDA1
KR7 Note 3	46	44			P91/A1/RXDA1
NMI Note 4	19	17	Input	External interrupt input (non-maskable, analog noise elimination). 5 V tolerant.	P02
$\overline{\text{RD}}$	69	67	Output	Read strobe signal output for external memory	PCT4
REGC	12	10	–	Connection of regulator output stabilization capacitance (4.7 μF)	–
$\overline{\text{RESET}}$	16	14	Input	System reset input	–
RXDA0	28	26	Input	Serial receive data input (UARTA0, UARTA1) 5 V tolerant.	P31/INTP7
RXDA1	38	36			P91/A1/KR7

Notes 1. μPD703252Y only

<R> **2.** μPD703253Y only

3. Pull this pin up externally.

4. The NMI pin alternately functions as the P02 pin. It functions as the P02 pin after reset. To enable the NMI pin, set the PMC0.PMC02 bit to 1. The initial setting of the NMI pin is “No edge detected”. Select the NMI pin valid edge using INTF0 and INTR0 registers.

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(4/5)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
$\overline{SCKB0}$	26	24	I/O	Serial clock I/O (CSIB0, CSIB1, CSIB3). N-ch open-drain output selectable. 5 V tolerant.	P42
$\overline{SCKB1}$	54	52			P99/A9
$\overline{SCKB3}$	57	55			P912/A12
SCL00	38	36	I/O	Serial clock I/O (I^2C00, I^2C01) N-ch open-drain output selectable. 5 V tolerant.	P39
SCL01	25	23			P41/SOB0
SDA00	37	35	I/O	Serial transmit/receive data I/O (I^2C00, I^2C01) N-ch open-drain output selectable. 5 V tolerant.	P38
SDA01	24	22			P40/SIB0
SIB0	24	22	Input	Serial receive data input (CSIB0, CSIB1, CSIB3) 5 V tolerant.	P40/SDA01
SIB1	52	50			P97/A7/TIP20/TOP20
SIB3	55	53			P910/A10
SOB0	25	23	Output	Serial transmit data output (CSIB0, CSIB1, CSIB3) N-ch open-drain output selectable. 5 V tolerant.	P41/SCL01
SOB1	53	51			P98/A8
SOB3	56	54			P911/A11
TIP00	29	27	Input	External event count input/capture trigger input/external trigger input (TMP0). 5 V tolerant.	P32/ASCKA0/TOP00
TIP01	30	28		Capture trigger input (TMP0). 5 V tolerant.	P33/TOP01
TIP10	31	29		External event count input/capture trigger input/external trigger input (TMP1). 5 V tolerant.	P34/TOP10
TIP11	32	30		Capture trigger input (TMP1). 5 V tolerant.	P35/TOP11
TIP20	52	50		External event count input/capture trigger input/external trigger input (TMP2). 5 V tolerant.	P97/A7/SIB1/TOP20
TIP21	51	49		Capture trigger input (TMP2). 5 V tolerant.	P96/A6/TOP21
TIP30	50	48		External event count input/capture trigger input/external trigger input (TMP3). 5 V tolerant.	P95/A5/TOP30
TIP31	49	47		Capture trigger input (TMP3). 5 V tolerant.	P94/A4/TOP31
TIP40	48	46		External event count input/capture trigger input/external trigger input (TMP4). 5 V tolerant	P93/A3/TOP40
TIP41	47	45		Capture trigger input (TMP4). 5 V tolerant.	P92/A2/TOP41
<R> TIP50[Note]	–	58		External event count input/capture trigger input/external trigger input (TMP5). 5 V tolerant.	P915/A15/INTP6/TOP50[Note]
<R> TIP51[Note]	–	57		Capture trigger input (TMP5). 5 V tolerant.	P914/A14/INTP5/TOP51[Note]

<R> **Note** μPD703253Y only

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(5/5)

Pin Name	Pin No.		I/O	Function	Alternate Function
	GF	GC			
TOP00	29	27	Output	Timer output (TMP0)	P32/ASCKA0/TIP00
TOP01	30	28		N-ch open-drain output selectable. 5 V tolerant.	P33/TIP01
TOP10	31	29		Timer output (TMP1)	P34/TIP10
TOP11	32	30		N-ch open-drain output selectable. 5 V tolerant.	P35/TIP11
TOP20	52	50	Output	Timer output (TMP2)	P97/A7/SIB1/TIP20
TOP21	51	49		N-ch open-drain output selectable. 5 V tolerant.	P96/A6/TIP21
TOP30	50	48		Timer output (TMP3)	P95/A5/TIP30
TOP31	49	47		N-ch open-drain output selectable. 5 V tolerant.	P94/A4/TIP31
TOP40	48	46		Timer output (TMP4)	P93/A3/TIP40
TOP41	47	45		N-ch open-drain output selectable. 5 V tolerant.	P92/A2/TIP41
<R> TOP50[Note]	–	58		Timer output (TMP5)	P915/A15/INTP6/TIP50[Note]
<R> TOP51[Note]	–	57		N-ch open-drain output selectable. 5 V tolerant.	P914/A14/INTP5/TIP51[Note]
TXDA0	27	25	Output	Serial transmit data output (UARTA0, UARTA1)	P30
TXDA1	45	43		N-ch open-drain output selectable. 5 V tolerant.	P90/A0/KR6
V_{DD}	11	9	–	Positive power supply pin for internal	–
V_{SS}	13	11	–	Ground potential for internal	–
$\overline{WAIT}$	63	61	Input	External wait input	PCM0
$\overline{WR0}$	67	65	Output	Write strobe for external memory (lower 8 bits)	PCT0
$\overline{WR1}$	68	66		Write strove for external memory (higher 8 bits)	PCT1
X1	14	12	Input	Connection of resonator for main clock	–
X2	15	13	–		–
XT1	17	15	Input	Connection of resonator for subclock	–
XT2	18	16	–		–

<R> **Note** μPD703253Y only

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

2.2 Pin States

The operation states of pins in the various modes are described below.

<R> **Table 2-2. Pin Operation States in Various Modes**

Pin Name	During Reset (Immediately After Power Is Turned On)	During Reset (Except Immediately After Power Is Turned On)	HALT Mode[Note 2]	IDLE1, IDLE2, Sub-IDLE Mode[Note 2]	STOP Mode[Note 2]	Idle State[Note 3]	Bus Hold
P05	Pulled down	Pulled down[Note 4]	Held	Held	Held	Held	Held
P10/ANO0[Note 5], P11/ANO1[Note 5]	Undefined[Note 1]	Hi-Z	Held	Held	**Note 10**	Held	Held
P53		Hi-Z	Held	Held	Held	Held	Held
AD0 to AD15	Hi-Z[Note 6]	Hi-Z[Note 6]	**Notes 7, 8**	Hi-Z	Hi-Z	Held	Hi-Z
A0 to A15			Undefined[Notes 7, 9]				
A16 to A21			Undefined[Note 7]				
$\overline{WAIT}$			–	–	–	–	–
CLKOUT			Operating	L	L	Operating	Operating
$\overline{WR0}$, $\overline{WR1}$			H[Note 7]	H	H	H	Hi-Z
$\overline{RD}$							
ASTB							
$\overline{HLDAK}$			Operating[Note 7]				L
$\overline{HLDRQ}$				–	–	–	Operating
Other port pins	Hi-Z	Hi-Z	Held	Held	Held	Held	Held

Notes 1. These pins may momentarily output an undefined level upon power application.

2. Operates while an alternate function is operating.

3. In separate bus mode, the state of the pins in the idle state inserted after the T2 state is shown. In multiplexed bus mode, the state of the pins in the idle state inserted after the T3 state is shown.

4. Pulled down during external reset. During internal reset by the watchdog timer, clock monitor, etc., the state of this pin differs according to the OCDM.OCDM0 bit setting.

5. μPD703253Y only

6. The bus control pins function alternately as port pins, so they are initialized to the input mode (port mode).

7. Operates even in the HALT mode, during DMA operation (μPD703253Y only).

8. In separate bus mode: Hi-Z
In multiplexed bus mode: Undefined

9. In separate bus mode

10. In port mode: Held
When alternate function is used: Hi-Z

Remark Hi-Z: High impedance
Held: The state during the immediately preceding external bus cycle is held.
L: Low-level output
H: High-level output
–: Input without sampling (not acknowledged)

2.3 Pin I/O Circuit Types, I/O Buffer Power Supplies, and Connection of Unused Pins

(1/3)

Pin	Alternate Function	Pin No.		I/O Circuit Type	Recommended Connection
		GF	GC		
P02	NMI	19	17	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P03	INTP0/ADTRG	20	18		
P04	INTP1	21	19		
P05	INTP2	22	20	10-N	Input: Independently connect to EV_{SS} via a resistor. Fixing to V_{DD} level is prohibited. Output: Leave open. Internally pull-down after reset by $\overline{\text{RESET}}$ pin.
P06	INTP3	23	21	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P10, P11	–	5, 6	3, 4	5	Input: Independently connect to AV_{REF1} or AV_{SS} via a resistor. Output: Leave open.
<R> P10, P11	ANO0, ANO1 Note 1	–	3, 4	12-D	
<R> P30	TXDA0	27	25	10-G	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
P31	RXDA0/INTP7	28	26	10-D	
P32	ASCKA0/TIP00/TOP00	29	27		
P33	TIP01/TOP01	30	28		
P34	TIP10/TOP10	31	29		
P35	TIP11/TOP11	32	30		
<R> P36	$\overline{\text{IETX0}}$ Note 2	33	31	10-G	
<R>	CTXD0 Note 1	–	31		
P37	$\overline{\text{IERX0}}$ Note 2	34	32	10-D	
<R>	CRXD0 Note 1	–	32		
P38	SDA00	37	35		
P39	SCL00	38	36		
P40	SIB0/SDA01	24	22		
P41	SOB0/SCL01	25	23		
P42	$\overline{\text{SCKB0}}$	26	24		
P50 to P55	KR0 to KR5	39 to 44	37 to 42		
P70 to P711	ANI0 to ANI11	2, 1, 100 to 91	100 to 89	11-G	Input: Independently connect to AV_{REF0} or AV_{SS} via a resistor. Output: Leave open.

<R> **Notes 1.** μPD703253Y only

2. μPD703252Y only

Remark GF: 100-pin plastic QFP (14 × 20)

GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(2/3)

	Pin	Alternate Function	Pin No.		I/O Circuit Type	Recommended Connection
			GF	GC		
	P90	A0/KR6/TDXA1	45	43	10-D	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
	P91	A1/KR7/RXDA1	46	44		
	P92	A2/TIP41/TOP41	47	45		
	P93	A3/TIP40/TOP40	48	46		
	P94	A4/TIP31/TOP31	49	47		
	P95	A5/TIP30/TOP30	50	48		
	P96	A6/TIP21/TOP21	51	49		
	P97	A7/SIB1/TIP20/TOP20	52	50		
<R>	P98	A8/SOB1	53	51	10-G	
	P99	A9/$\overline{\text{SCKB1}}$	54	52	10-D	
	P910	A10/SIB3	55	53		
<R>	P911	A11/SOB3	56	54	10-G	
	P912	A12/$\overline{\text{SCKB3}}$	57	55	10-D	
	P913	A13/INTP4	58	56		
	P914	A14/INTP5	59	57		
<R>		A14/INTP5/TIP51[Note]/TOP51[Note]	–	57		
	P915	A15/INTP6	60	58		
<R>		A15/INTP6/TIP50[Note]/TOP50[Note]	–	58		
	PCM0	$\overline{\text{WAIT}}$	63	61	5	Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.
	PCM1	CLKOUT	64	62		
	PCM2	$\overline{\text{HLDAK}}$	65	63		
	PCM3	$\overline{\text{HLDRQ}}$	66	64		
	PCT0, PCT1	$\overline{\text{WR0}}$, $\overline{\text{WR1}}$	67, 68	65, 66		
	PCT4	$\overline{\text{RD}}$	69	67		
	PCT6	ASTB	70	68		
	PDH0 to PDH3	A16 to A19	89, 90, 61, 62	87, 88, 59, 60		
	PDH4, PDH5	A20, A21	8, 9	6, 7		Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
	PDL0 to PDL15	AD0 to AD15	73 to 88	71 to 86		Input: Independently connect to BV_{DD} or BV_{SS} via a resistor. Output: Leave open.

<R> **Note** μPD703253Y only

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

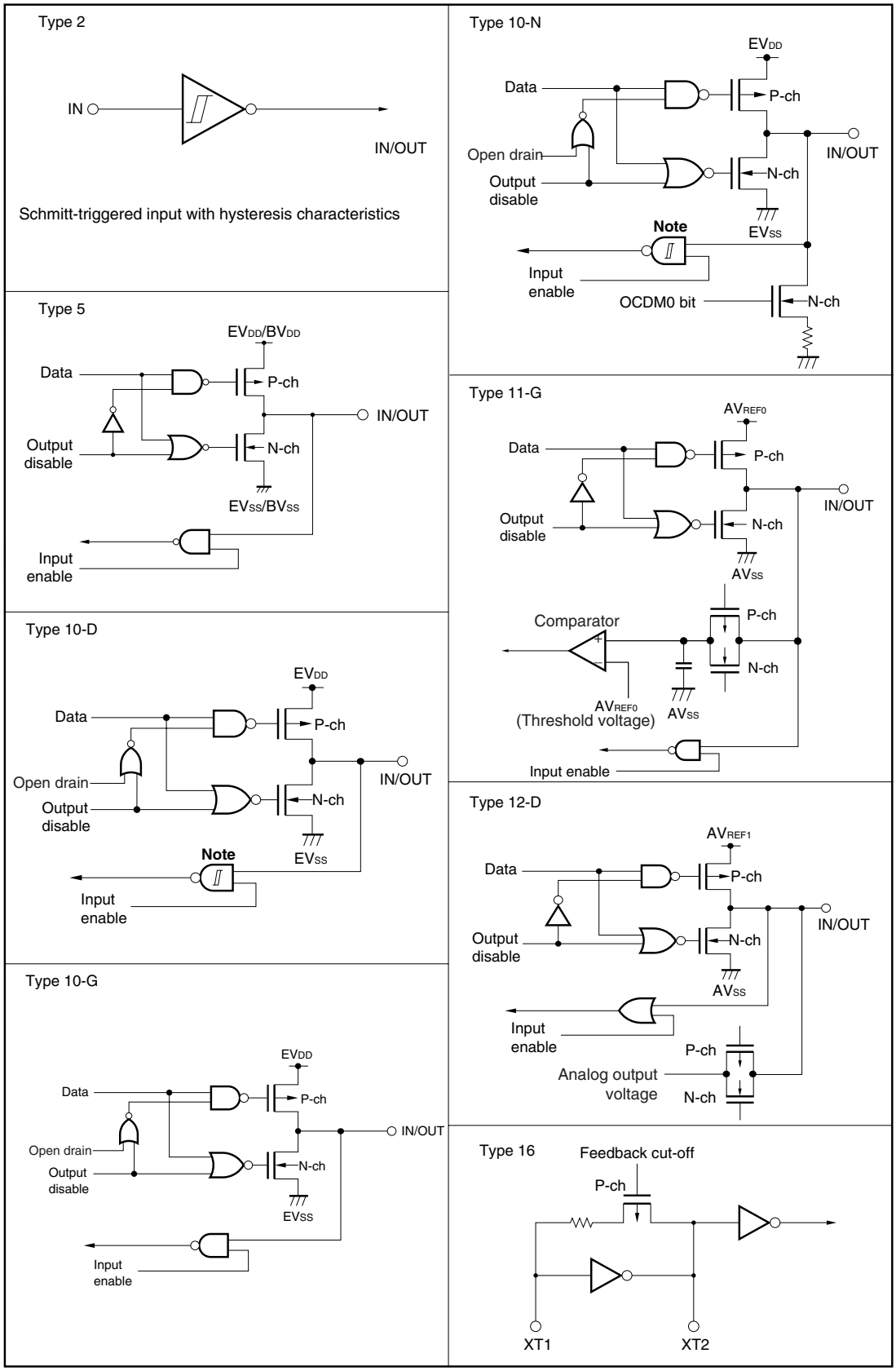
(3/3)

Pin	Alternate Function	Pin No.		I/O Circuit Type	Recommended Connection
		GF	GC		
AV_{REF0}	–	3	1	–	Directly connect to V_{DD} and always supply power.
AV_{REF1}	–	7	5	–	Directly connect to V_{DD} and always supply power.
AV_{SS}	–	4	2	–	Directly connect to V_{SS} and always supply power.
BV_{DD}	–	72	70	–	Directly connect to V_{DD} and always supply power.
BV_{SS}	–	71	69	–	Directly connect to V_{SS} and always supply power.
EV_{DD}	–	36	34	–	–
EV_{SS}	–	35	33	–	–
IC	–	10	8	–	Directly connect to V_{SS}.
REGC	–	12	10	–	Connect regulator output stabilization capacitance (4.7 μF).
$\overline{RESET}$	–	16	14	2	–
V_{DD}	–	11	9	–	–
V_{SS}	–	13	11	–	–
X1	–	14	12	–	–
X2	–	15	13	–	–
XT1	–	17	15	16	Connect to V_{SS}.
XT2	–	18	16	16	Leave open.

Remark GF: 100-pin plastic QFP (14 × 20)

GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

Figure 2-1. Pin I/O Circuits



Note Hysteresis characteristics are not available in port mode.

2.4 Cautions

<R> The following pins may momentarily output an undefined level upon power application.

<R> • P10/ANO0[Note] pin
<R> • P11/ANO1[Note] pin
• P53/KR3 pin

<R> **Note** μPD703253Y only

CHAPTER 3 CPU FUNCTION

The CPU of the V850ES/SG1 is based on RISC architecture and executes almost all instructions with one clock by using a 5-stage pipeline.

3.1 Features

○ Minimum instruction execution time: 50 ns (operating with main clock (f_{XX}) = 20 MHz)
30.5 μs (with subclock (f_{XT}) = 32.768 kHz operation)

○ Memory space Program (physical address) space: 64 MB linear
Data (logical address) space: 4 GB linear

○ General-purpose registers: 32 bits × 32 registers

○ Internal 32-bit architecture

○ 5-stage pipeline control

○ Multiplication/division instruction

○ Saturation operation instruction

○ 32-bit shift instruction: 1 clock

○ Load/store instruction with long/short format

○ Four types of bit manipulation instructions
- SET1
- CLR1
- NOT1
- TST1

3.2 CPU Register Set

The registers of the V850ES/SG1 can be classified into two types: general-purpose program registers and dedicated system registers. All the registers are 32 bits wide.

For details, refer to the **V850ES Architecture User's Manual**.

(1) Program register set

31	0
r0	(Zero register)
r1	(Assembler-reserved register)
r2	
r3	(Stack pointer (SP))
r4	(Global pointer (GP))
r5	(Text pointer (TP))
r6	
r7	
r8	
r9	
r10	
r11	
r12	
r13	
r14	
r15	
r16	
r17	
r18	
r19	
r20	
r21	
r22	
r23	
r24	
r25	
r26	
r27	
r28	
r29	
r30	(Element pointer (EP))
r31	(Link pointer (LP))

31	0
PC	(Program counter)

(2) System register set

31	0
EIPC	(Interrupt status saving register)
EIPSW	(Interrupt status saving register)

FEPC	(NMI status saving register)
FEPSW	(NMI status saving register)

ECR	(Interrupt source register)

PSW	(Program status word)

CTPC	(CALLT execution status saving register)
CTPSW	(CALLT execution status saving register)

DBPC	(Exception/debug trap status saving register)
DBPSW	(Exception/debug trap status saving register)

CTBP	(CALLT base pointer)

3.2.1 Program register set

The program registers include general-purpose registers and a program counter.

(1) General-purpose registers (r0 to r31)

Thirty-two general-purpose registers, r0 to r31, are available. Any of these registers can be used to store a data variable or an address variable.

However, r0 and r30 are implicitly used by instructions and care must be exercised when these registers are used. r0 always holds 0 and is used for an operation that uses 0 or addressing of offset 0. r30 is used by the SLD and SST instructions as a base pointer when these instructions access the memory. r1, r3 to r5, and r31 are implicitly used by the assembler and C compiler. When using these registers, save their contents for protection, and then restore the contents after using the registers. r2 is sometimes used by the real-time OS. If the real-time OS does not use r2, it can be used as a register for variables.

Table 3-1. Program Registers

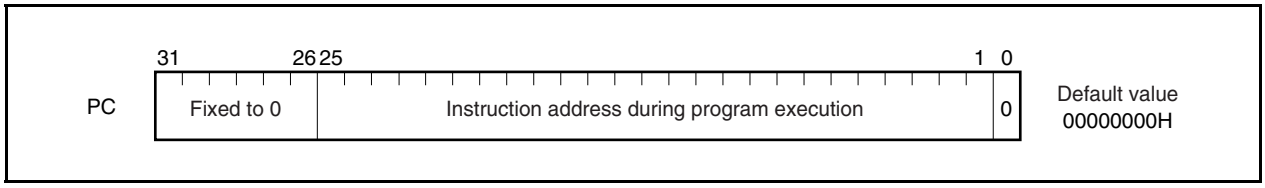
Name	Usage	Operation
r0	Zero register	Always holds 0.
r1	Assembler-reserved register	Used as working register to create 32-bit immediate data
r2	Register for address/data variable (if real-time OS does not use r2)	
r3	Stack pointer	Used to create a stack frame when a function is called
r4	Global pointer	Used to access a global variable in the data area
r5	Text pointer	Used as register that indicates the beginning of a text area (area where program codes are located)
r6 to r29	Register for address/data variable	
r30	Element pointer	Used as base pointer to access memory
r31	Link pointer	Used when the compiler calls a function
PC	Program counter	Holds the instruction address during program execution

Remark For further details on the r1, r3 to r5, and r31 that are used in the assembler and C compiler, refer to the **CA850 (C Compiler Package) Assembly Language User's Manual**.

(2) Program counter (PC)

The program counter holds the instruction address during program execution. The lower 32 bits of this register are valid. Bits 31 to 26 are fixed to 0. A carry from bit 25 to 26 is ignored even if it occurs.

Bit 0 is fixed to 0. This means that execution cannot branch to an odd address.



3.2.2 System register set

The system registers control the status of the CPU and hold interrupt information.

These registers can be read or written by using system register load/store instructions (LDSR and STSR), using the system register numbers listed below.

Table 3-2. System Register Numbers

System Register Number	System Register Name	Operand Specification	
		LDSR Instruction	STSR Instruction
0	Interrupt status saving register (EIPC)[Note 1]	√	√
1	Interrupt status saving register (EIPSW)[Note 1]	√	√
2	NMI status saving register (FEPC)[Note 1]	√	√
3	NMI status saving register (FEPSW)[Note 1]	√	√
4	Interrupt source register (ECR)	×	√
5	Program status word (PSW)	√	√
6 to 15	Reserved for future function expansion (operation is not guaranteed if these registers are accessed)	×	×
16	CALLT execution status saving register (CTPC)	√	√
17	CALLT execution status saving register (CTPSW)	√	√
18	Exception/debug trap status saving register (DBPC)	√[Note 2]	√[Note 2]
19	Exception/debug trap status saving register (DBPSW)	√[Note 2]	√[Note 2]
20	CALLT base pointer (CTBP)	√	√
21 to 31	Reserved for future function expansion (operation is not guaranteed if these registers are accessed)	×	×

Notes 1. Because only one set of these registers is available, the contents of these registers must be saved by program if multiple interrupts are enabled.

<R> **2.** These registers can be accessed only during the interval between the execution of the DBTRAP instruction or illegal instruction code and DBRET instruction execution.

Caution Even if EIPC or FEPC, or bit 0 of CTPC is set to 1 by the LDSR instruction, bit 0 is ignored when execution is returned to the main routine by the RETI instruction after interrupt servicing (this is because bit 0 of the PC is fixed to 0). Set an even value to EIPC, FEPC, and CTPC (bit 0 = 0).

Remark √: Can be accessed
×: Access prohibited

(1) Interrupt status saving registers (EIPC and EIPSW)

EIPC and EIPSW are used to save the status when an interrupt occurs.

If a software exception or a maskable interrupt occurs, the contents of the program counter (PC) are saved to EIPC, and the contents of the program status word (PSW) are saved to EIPSW (these contents are saved to the NMI status saving registers (FEPC and FEPSW) if a non-maskable interrupt occurs).

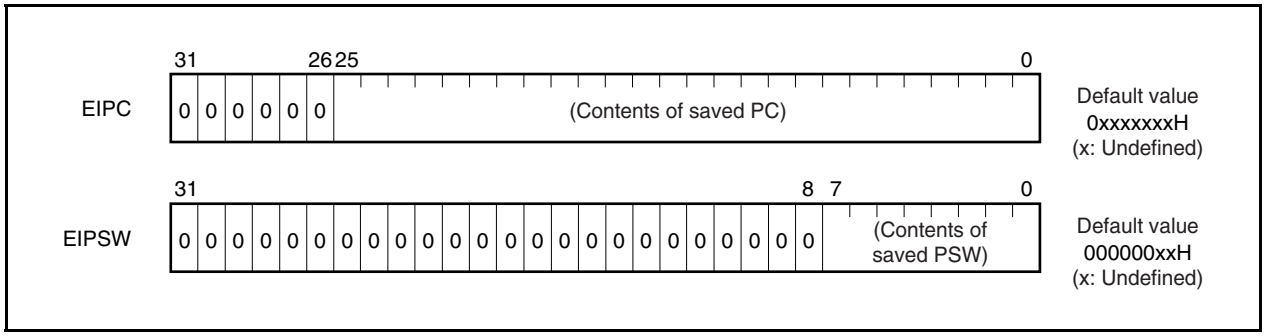
The address of the instruction next to the instruction under execution, except some instructions (see **19.8 Periods in Which Interrupts Are Not Acknowledged by CPU**), is saved to EIPC when a software exception or a maskable interrupt occurs.

The current contents of the PSW are saved to EIPSW.

Because only one set of interrupt status saving registers is available, the contents of these registers must be saved by program when multiple interrupts are enabled.

Bits 31 to 26 of EIPC and bits 31 to 8 of EIPSW are reserved for future function expansion (these bits are always fixed to 0).

The value of EIPC is restored to the PC and the value of EIPSW to the PSW by the RETI instruction.



(2) NMI status saving registers (FEPC and FEPSW)

FEPC and FEPSW are used to save the status when a non-maskable interrupt (NMI) occurs.

If an NMI occurs, the contents of the program counter (PC) are saved to FEPC, and those of the program status word (PSW) are saved to FEPSW.

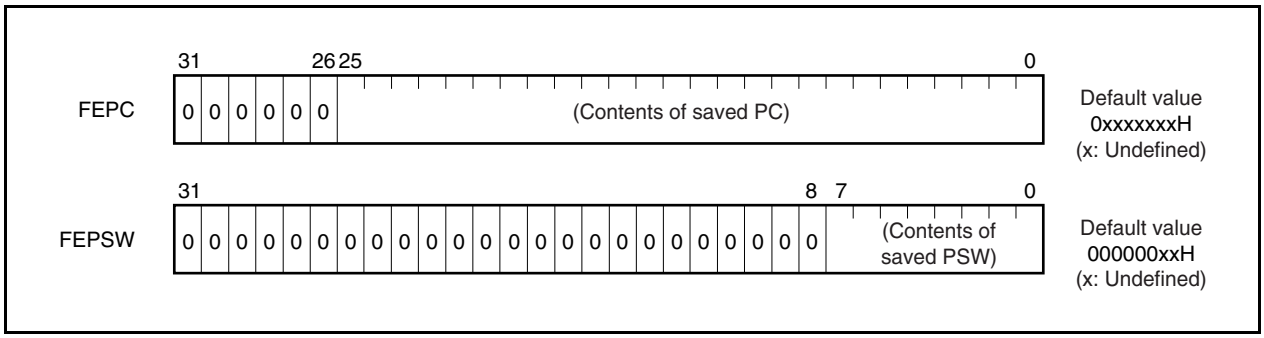
The address of the instruction next to the one of the instruction under execution, except some instructions, is saved to FEPC when an NMI occurs.

The current contents of the PSW are saved to FEPSW.

Because only one set of NMI status saving registers is available, the contents of these registers must be saved by program when multiple interrupts are enabled.

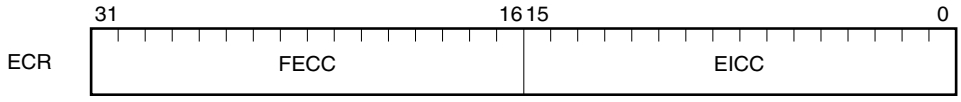
Bits 31 to 26 of FEPC and bits 31 to 8 of FEPSW are reserved for future function expansion (these bits are always fixed to 0).

The value of FEPC is restored to the PC and the value of FEPSW to the PSW by the RETI instruction.



(3) Interrupt source register (ECR)

The interrupt source register (ECR) holds the source of an exception or interrupt if an exception or interrupt occurs. This register holds the exception code of each interrupt source. Because this register is a read-only register, data cannot be written to this register using the LDSR instruction.



Bit position	Bit name	Meaning
31 to 16	FECC	Exception code of non-maskable interrupt (NMI)
15 to 0	EICC	Exception code of exception or maskable interrupt

(4) Program status word (PSW)

The program status word (PSW) is a collection of flags that indicate the status of the program (result of instruction execution) and the status of the CPU.

If the contents of a bit of this register are changed by using the LDSR instruction, the new contents are validated immediately after completion of LDSR instruction execution. However if the ID flag is set to 1, interrupt requests will not be acknowledged while the LDSR instruction is being executed.

Bits 31 to 8 of this register are reserved for future function expansion (these bits are fixed to 0).

(1/2)

Bit position	Flag name	Meaning
31 to 8	RFU	Reserved field. Fixed to 0.
7	NP	Indicates that a non-maskable interrupt (NMI) is being serviced. This bit is set to 1 when an NMI request is acknowledged, disabling multiple interrupts. 0: NMI is not being serviced. 1: NMI is being serviced.
6	EP	Indicates that an exception is being processed. This bit is set to 1 when an exception occurs. Even if this bit is set, interrupt requests are acknowledged. 0: Exception is not being processed. 1: Exception is being processed.
5	ID	Indicates whether a maskable interrupt can be acknowledged. 0: Interrupt enabled 1: Interrupt disabled
4	SAT[Note]	Indicates that the result of a saturation operation has overflowed and is saturated. Because this is a cumulative flag, it is set to 1 when the result of a saturation operation instruction is saturated, and is not cleared to 0 even if the subsequent operation result is not saturated. Use the LDSR instruction to clear this bit. This flag is neither set to 1 nor cleared to 0 by execution of an arithmetic operation instruction. 0: Not saturated 1: Saturated
3	CY	Indicates whether a carry or a borrow occurs as a result of an operation. 0: Carry or borrow does not occur. 1: Carry or borrow occurs.
2	OV[Note]	Indicates whether an overflow occurs during operation. 0: Overflow does not occur. 1: Overflow occurs.
1	S[Note]	Indicates whether the result of an operation is negative. 0: The result is positive or 0. 1: The result is negative.
0	Z	Indicates whether the result of an operation is 0. 0: The result is not 0. 1: The result is 0.

Remark Also read **Note** on the next page.

(2/2)

Note The result of the operation that has performed saturation processing is determined by the contents of the OV and S flags. The SAT flag is set to 1 only when the OV flag is set to 1 when a saturation operation is performed.

Status of operation result	Flag status			Result of operation of saturation processing
	SAT	OV	S	
Maximum positive value is exceeded	1	1	0	7FFFFFFFH
Maximum negative value is exceeded	1	1	1	80000000H
Positive (maximum value is not exceeded)	Holds value before operation	0	0	Operation result itself
Negative (maximum value is not exceeded)			1	

(5) CALLT execution status saving registers (CTPC and CTPSW)

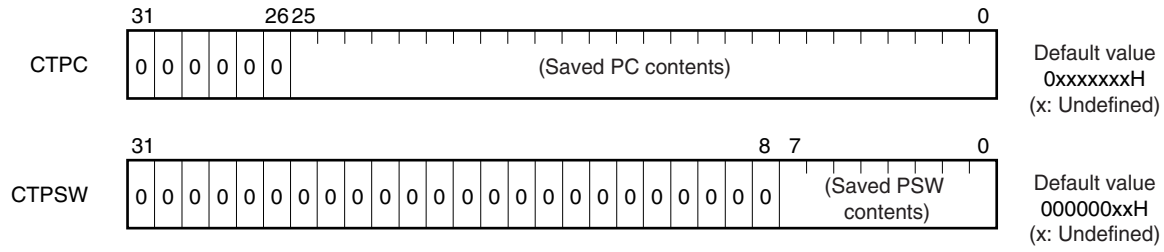
CTPC and CTPSW are CALLT execution status saving registers.

When the CALLT instruction is executed, the contents of the program counter (PC) are saved to CTPC, and those of the program status word (PSW) are saved to CTPSW.

The contents saved to CTPC are the address of the instruction next to CALLT.

The current contents of the PSW are saved to CTPSW.

Bits 31 to 26 of CTPC and bits 31 to 8 of CTPSW are reserved for future function expansion (fixed to 0).



(6) Exception/debug trap status saving registers (DBPC and DBPSW)

DBPC and DBPSW are exception/debug trap status registers.

If an exception trap or debug trap occurs, the contents of the program counter (PC) are saved to DBPC, and those of the program status word (PSW) are saved to DBPSW.

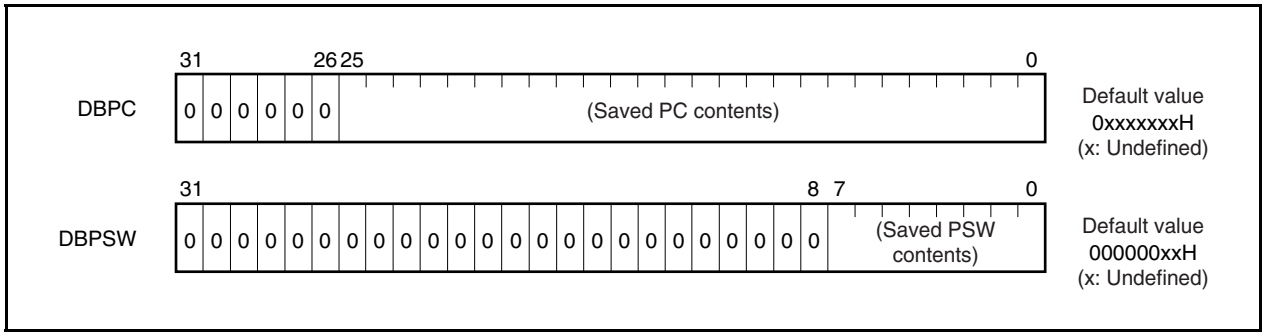
The contents to be saved to DBPC are the address of the instruction next to the one that is being executed when an exception trap or debug trap occurs.

The current contents of the PSW are saved to DBPSW.

<R> This register can be read or written only during the interval between the execution of the DBTRAP instruction or illegal instruction code and DBRET instruction.

Bits 31 to 26 of DBPC and bits 31 to 8 of DBPSW are reserved for future function expansion (fixed to 0).

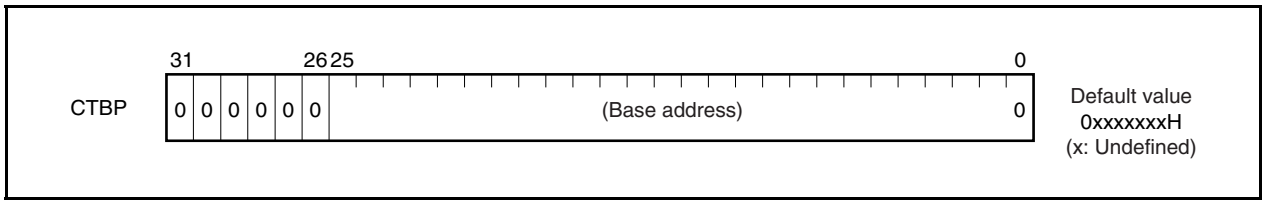
The value of DBPC is restored to the PC and the value of DBPSW to the PSW by the DBRET instruction.



(7) CALLT base pointer (CTBP)

The CALLT base pointer (CTBP) is used to specify a table address or generate a target address (bit 0 is fixed to 0).

Bits 31 to 26 of this register are reserved for future function expansion (fixed to 0).



3.3 Operation Modes

In the normal operation mode of the V850ES/SG1, each pin related to the bus interface is set to the port mode after system reset has been released. Execution branches to the reset entry address of the internal ROM, and then instruction processing is started.

3.3.1 Specifying operation mode

Specify the operation mode by using the IC pin.

Always input a low level to the IC pin.

IC Pin	Operation Mode
L	Normal operation mode
H	Setting prohibited

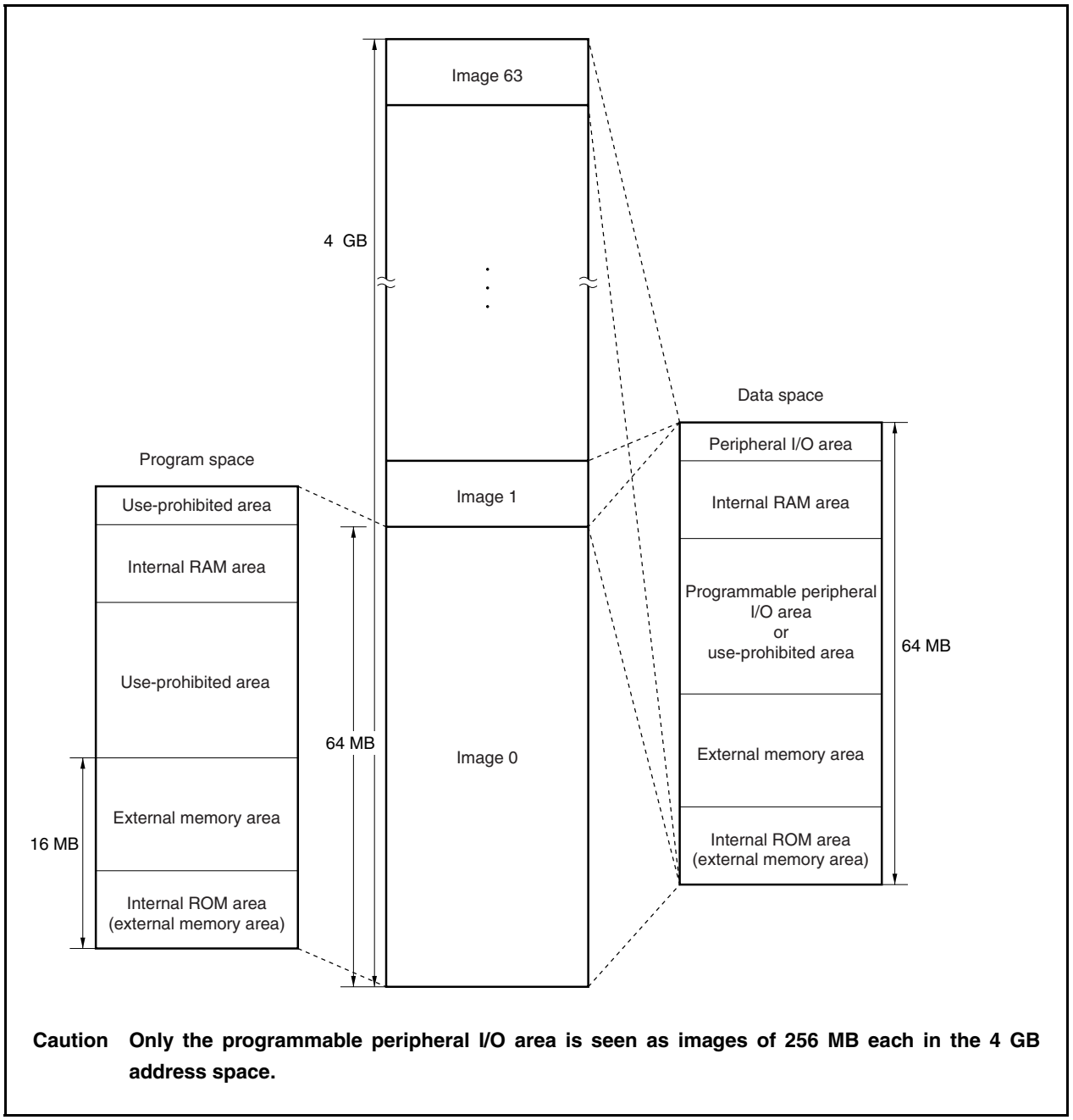
Remark L: Low-level input
H: High-level input

3.4 Address Space

3.4.1 CPU address space

For instruction addressing, up to a combined total of 16 MB of external memory area and internal ROM area, plus an internal RAM area, are supported in a linear address space (program space) of up to 64 MB. For operand addressing (data access), up to 4 GB of a linear address space (data space) is supported. The 4 GB address space, however, is viewed as 64 images of a 64 MB physical address space. This means that the same 64 MB physical address space is accessed regardless of the value of bits 31 to 26.

Figure 3-1. Image on Address Space



<R> **Caution Only the programmable peripheral I/O area is seen as images of 256 MB each in the 4 GB address space.**

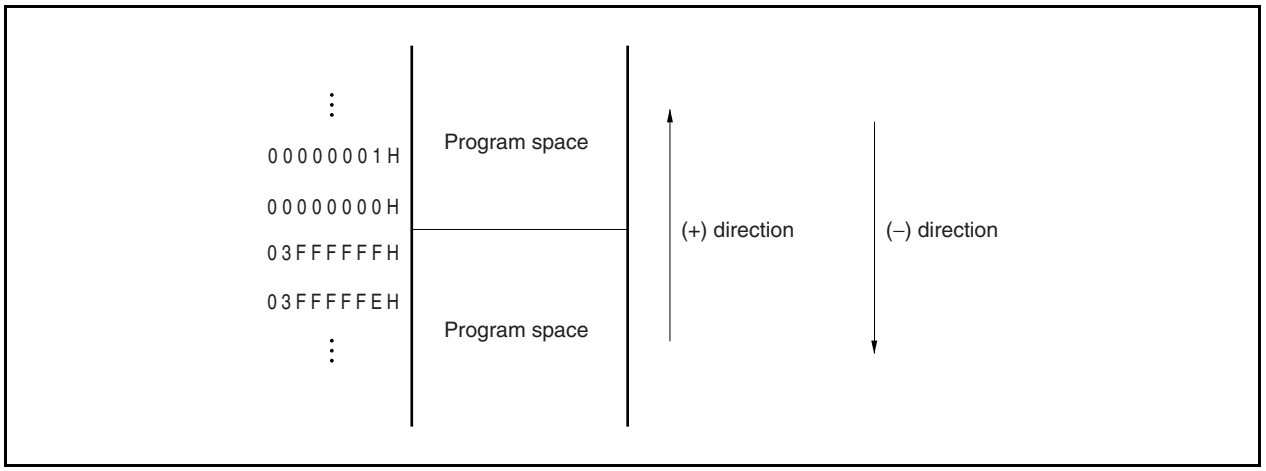
3.4.2 Wraparound of CPU address space

(1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed to 0 and only the lower 26 bits are valid. The higher 6 bits ignore a carry or borrow from bit 25 to 26 during branch address calculation.

Therefore, the highest address of the program space, 03FFFFFFH, and the lowest address, 00000000H, are contiguous addresses. That the highest address and the lowest address of the program space are contiguous in this way is called wraparound.

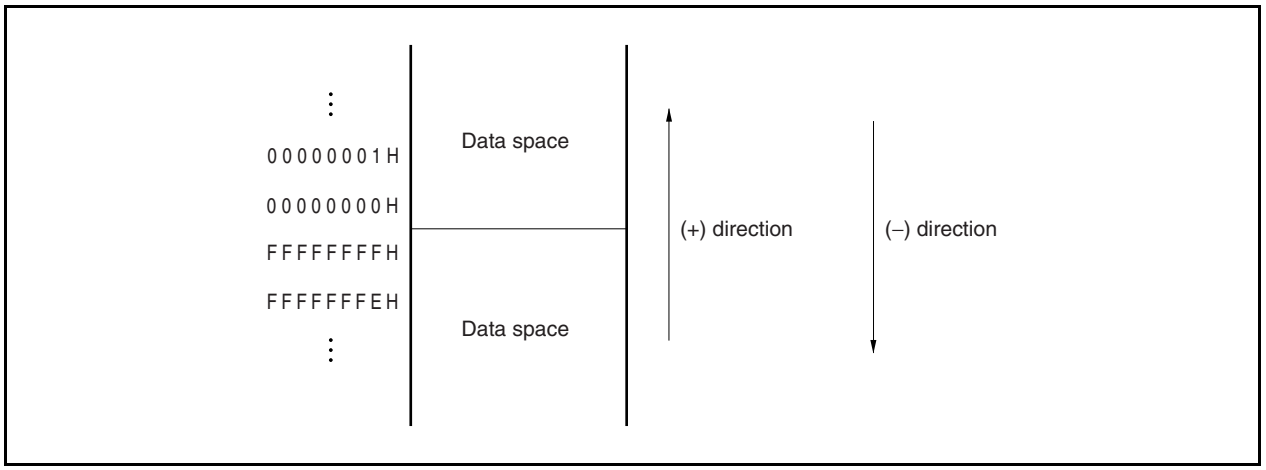
Caution Because the 4 KB area of addresses 03FFF000H to 03FFFFFFH is an on-chip peripheral I/O area, instructions cannot be fetched from this area. Therefore, do not execute an operation in which the result of a branch address calculation affects this area.



(2) Data space

The result of an operand address calculation operation that exceeds 32 bits is ignored.

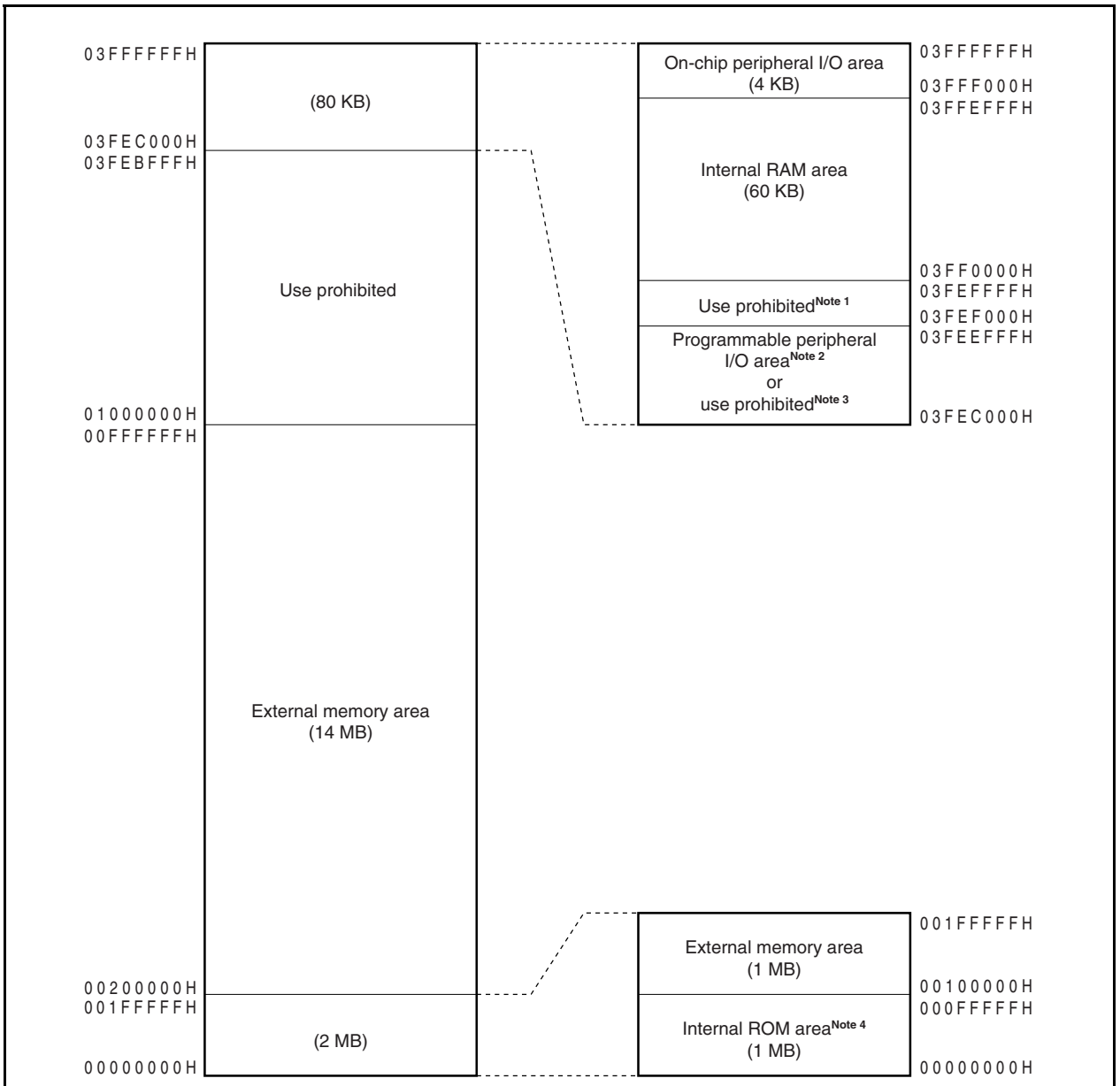
Therefore, the highest address of the data space, FFFFFFFFH, and the lowest address, 00000000H, are contiguous, and wraparound occurs at the boundary of these addresses.



3.4.3 Memory map

The areas shown below are reserved in the V850ES/SG1.

Figure 3-2. Data Memory Map (Physical Addresses)



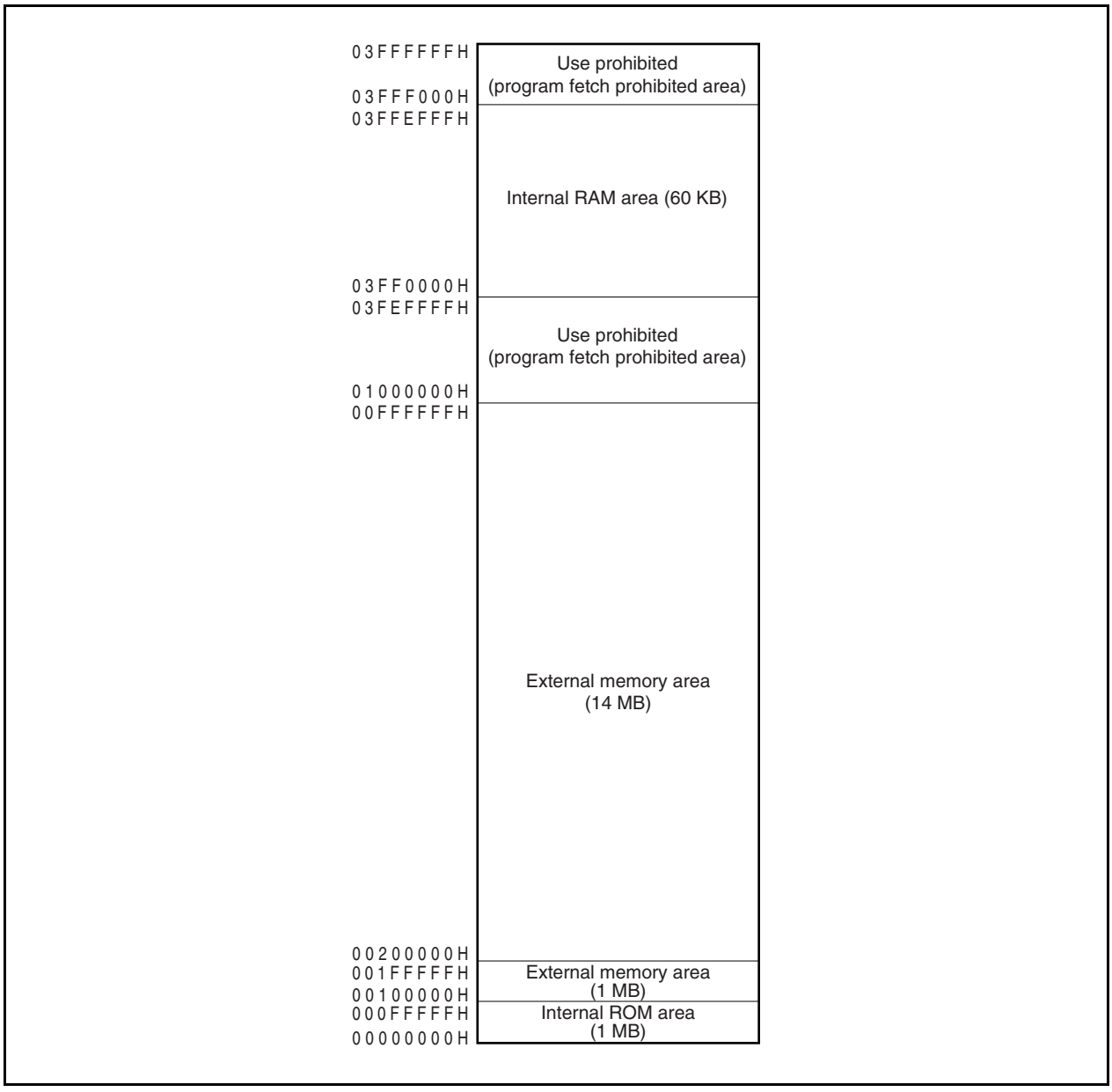
Notes 1. Use of addresses 03FEF000H to 03FEFFFFH is prohibited because these addresses are in the same area as the on-chip peripheral I/O area.

<R> **2.** The programmable peripheral I/O area only can be viewed in the 4 GB address space as the image in 256 MB unit.

<R> **3.** Addresses 03FEC000H to 03FEC5FFH are allocated to addresses 03FEC000H to 03FEEFFFH of the μPD703253Y as a programmable peripheral I/O area. Use of these addresses in the μPD703249Y and 703252Y is prohibited.

4. Fetch access and read access to addresses 00000000H to 000FFFFFH is made to the internal ROM area. However, data write access to these addresses is made to the external memory area.

Figure 3-3. Program Memory Map



3.4.4 Areas

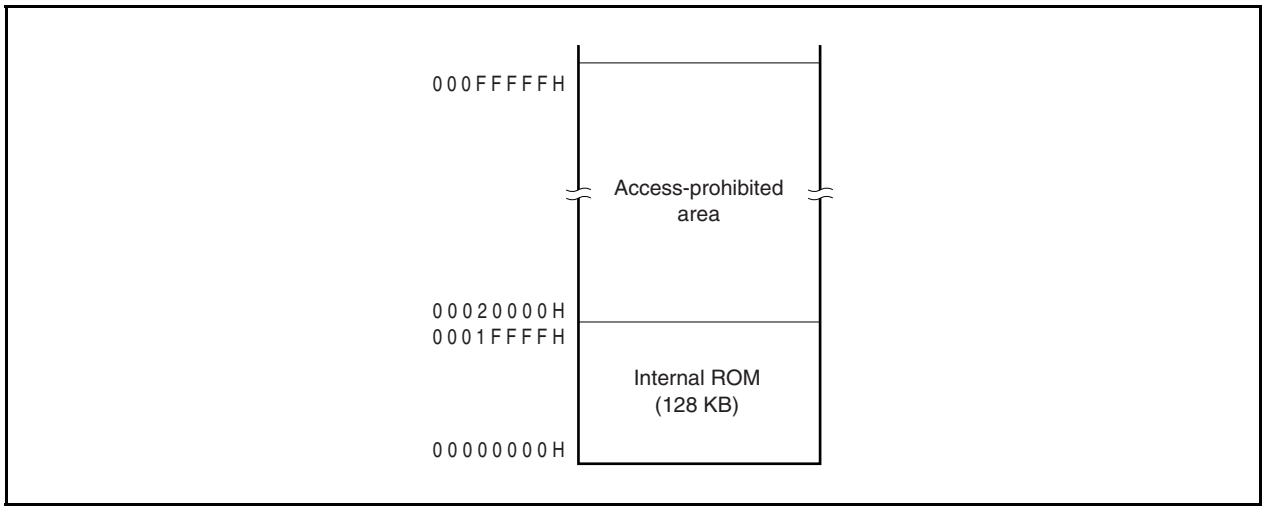
(1) Internal ROM area

Up to 1 MB is reserved as an internal ROM area.

<R> **(a) Internal ROM (128 KB)**

128 KB are allocated to addresses 00000000H to 0001FFFFH in the μPD703253Y.
Accessing addresses 00020000H to 000FFFFFH is prohibited.

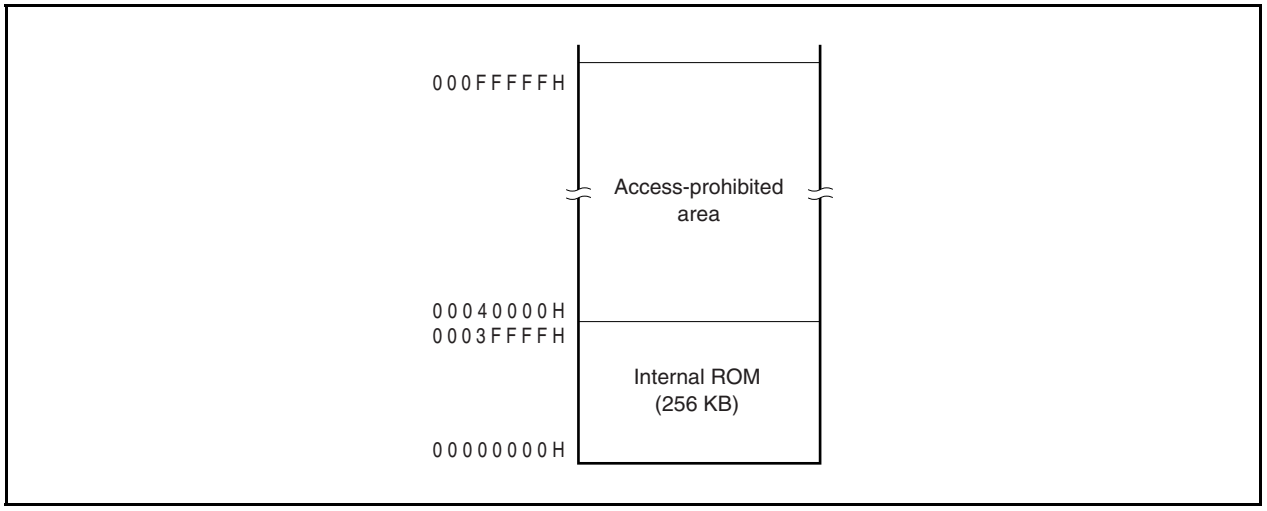
Figure 3-4. Internal ROM Area (128 KB)



(b) Internal ROM (256 KB)

256 KB are allocated to addresses 00000000H to 0003FFFFH in the μPD703249Y and 703252Y.
Accessing addresses 00040000H to 000FFFFFH is prohibited.

Figure 3-5. Internal ROM Area (256 KB)



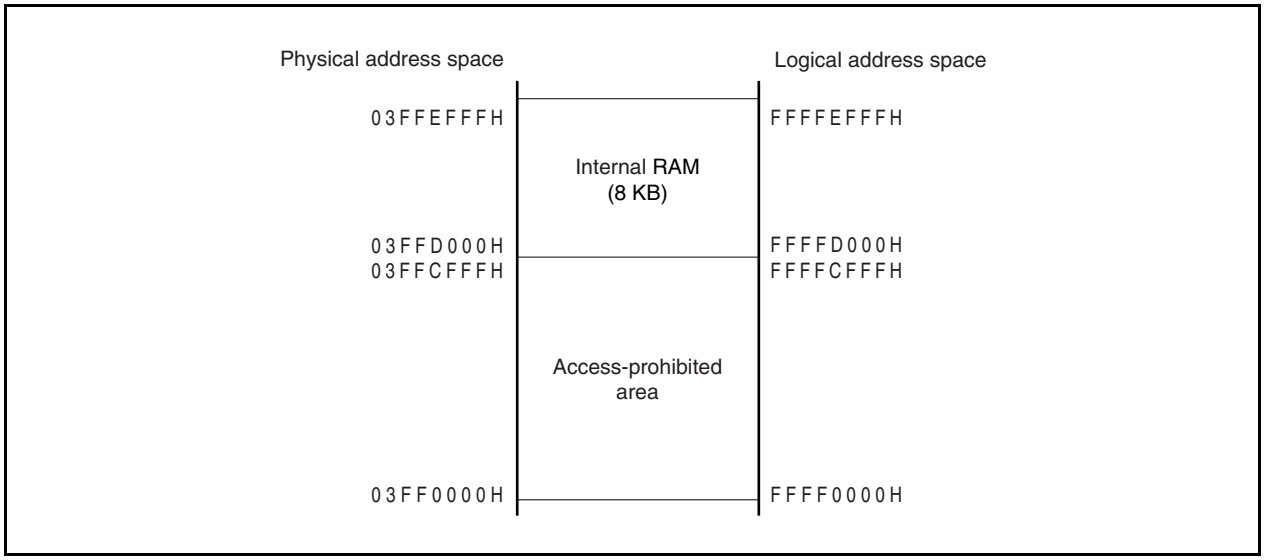
(2) Internal RAM area

Up to 60 KB are reserved as the internal RAM area.

<R> **(a) Internal RAM (8 KB)**

8 KB are allocated to addresses 03FFD000H to 03FFEFFFH of the μPD703253Y.
Accessing addresses 03FF0000H to 03FFCFFFH is prohibited.

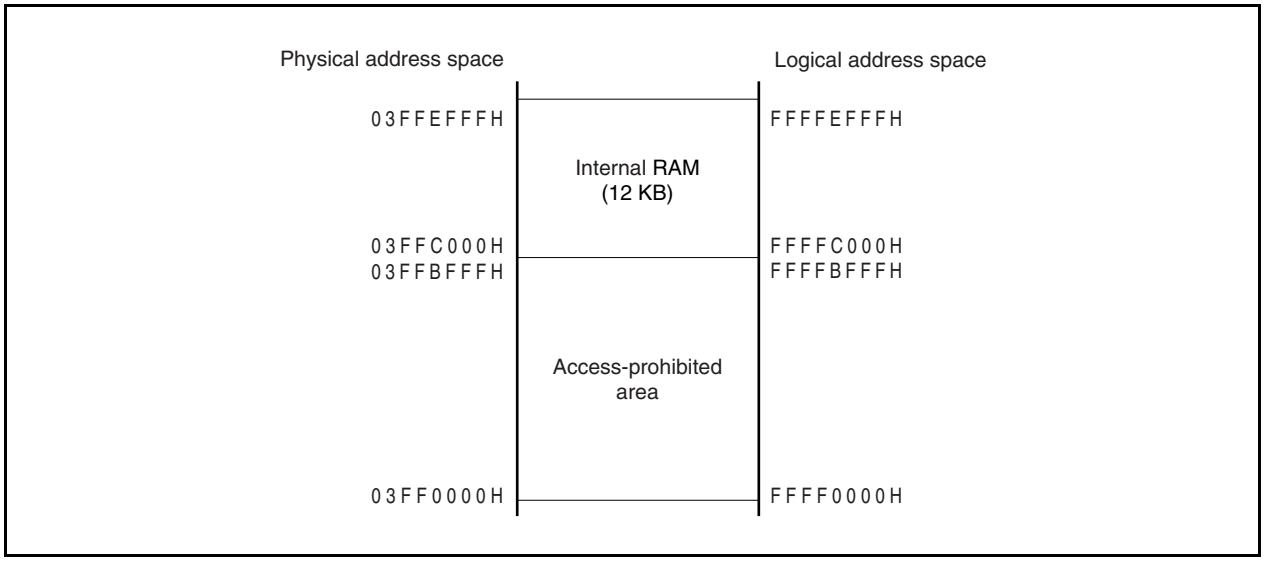
Figure 3-6. Internal RAM Area (8 KB)



(b) Internal RAM (12 KB)

12 KB are allocated to addresses 03FFC000H to 03FFEFFFH in the μPD703249Y and 703252Y.
Accessing addresses 03FF0000H to 03FFBFFFH is prohibited.

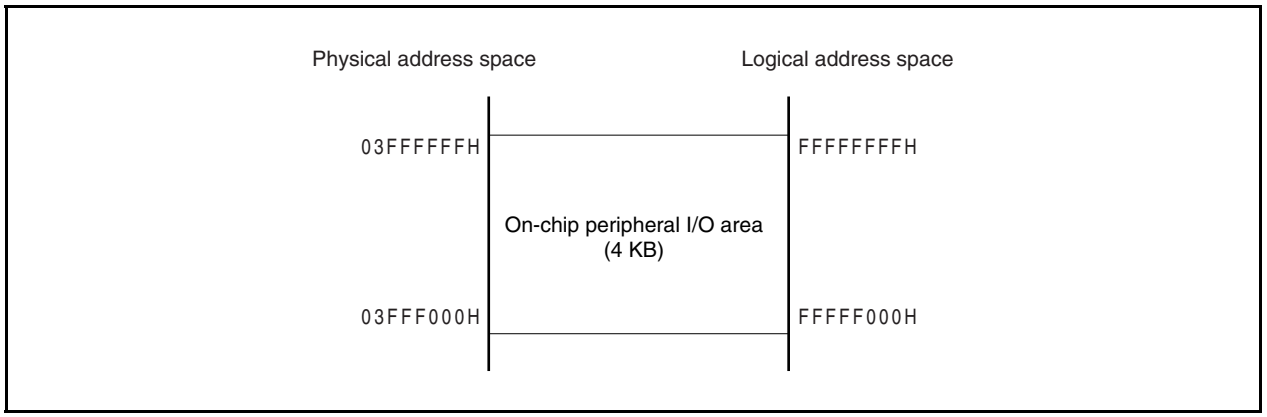
Figure 3-7. Internal RAM Area (12 KB)



(3) On-chip peripheral I/O area

4 KB of addresses 03FFF000H to 03FFFFFFH are reserved as the on-chip peripheral I/O area.

Figure 3-8. On-Chip Peripheral I/O Area



Peripheral I/O registers that have functions to specify the operation mode for and monitor the status of the on-chip peripheral I/O are mapped to the on-chip peripheral I/O area. Program cannot be fetched from this area.

Cautions 1. When a register is accessed in word units, a word area is accessed twice in halfword units in the order of lower area and higher area, with the lower 2 bits of the address ignored.

2. If a register that can be accessed in byte units is accessed in halfword units, the higher 8 bits are undefined when the register is read, and data is written to the lower 8 bits.

3. Addresses not defined as registers are reserved for future expansion. The operation is undefined and not guaranteed when these addresses are accessed.

<R> **4. The internal ROM/RAM area and on-chip peripheral I/O area are assigned to successive addresses. When accessing the internal ROM/RAM area by incrementing or decrementing addresses using pointer operations and such, therefore, be careful not to access the onchip peripheral I/O area by mistakenly extending over the internal ROM/RAM area boundary.**

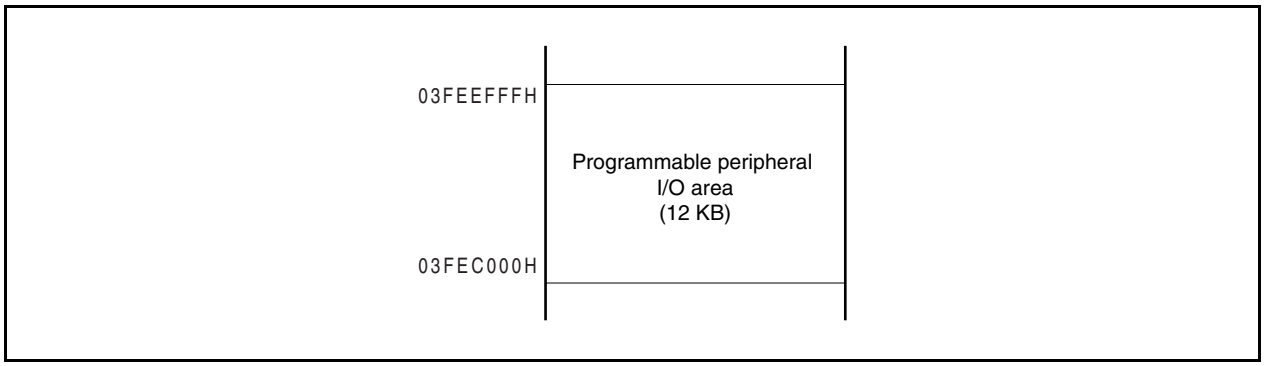
<R> **(4) Programmable peripheral I/O area**

Cautions 1. The programmable peripheral I/O area exists only in the μPD703253Y. This area cannot be used in the μPD703249Y and 703252Y.

2. The programmable peripheral I/O area is seen as images of 256 MB each in the 4 GB address space.

12 KB of addresses 03FEC000H to 03FEEFFFH are reserved as the programmable peripheral I/O area.

Figure 3-9. Programmable Peripheral I/O Area



(5) External memory area

15 MB (00100000H to 00FFFFFFH) are allocated as the external memory area. For details, see **CHAPTER 5 BUS CONTROL FUNCTION**.

<R> **Caution The V850ES/SG1 has 22 address pins (A0 to A21), so the external memory area appears as a repeated 4 MB image. When A20 and A21 are used, it is necessary that $EV_{DD} = BV_{DD} = V_{DD}$.**

3.4.5 Recommended use of address space

The architecture of the V850ES/SG1 requires that a register that serves as a pointer be secured for address generation when operand data in the data space is accessed. The address stored in this pointer ±32 KB can be directly accessed by an instruction for operand data. Because the number of general-purpose registers that can be used as a pointer is limited, however, by keeping the performance from dropping during address calculation when a pointer value is changed, as many general-purpose registers as possible can be secured for variables, and the program size can be reduced.

(1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed to 0, and only the lower 26 bits are valid. Regarding the program space, therefore, a 64 MB space of contiguous addresses starting from 00000000H unconditionally corresponds to the memory map.

To use the internal RAM area as the program space, access the following addresses.

Caution If a branch instruction is at the upper limit of the internal RAM area, a prefetch operation (invalid fetch) straddling the on-chip peripheral I/O area does not occur.

<R>

RAM Size	Access Address
12 KB	03FFC000H to 03FFEFFFH
8 KB	03FFD000H to 03FFEFFFH

(2) Data space

With the V850ES/SG1, it seems that there are sixty-four 64 MB address spaces on the 4 GB CPU address space. Therefore, the least significant bit (bit 25) of a 26-bit address is sign-extended to 32 bits and allocated as an address.

(a) Application example of wraparound

If R = r0 (zero register) is specified for the LD/ST disp16 [R] instruction, a range of addresses 00000000H ±32 KB can be addressed by sign-extended disp16. All the resources, including the internal hardware, can be addressed by one pointer.

The zero register (r0) is a register fixed to 0 by hardware, and practically eliminates the need for registers dedicated to pointers.

Example μPD703253Y

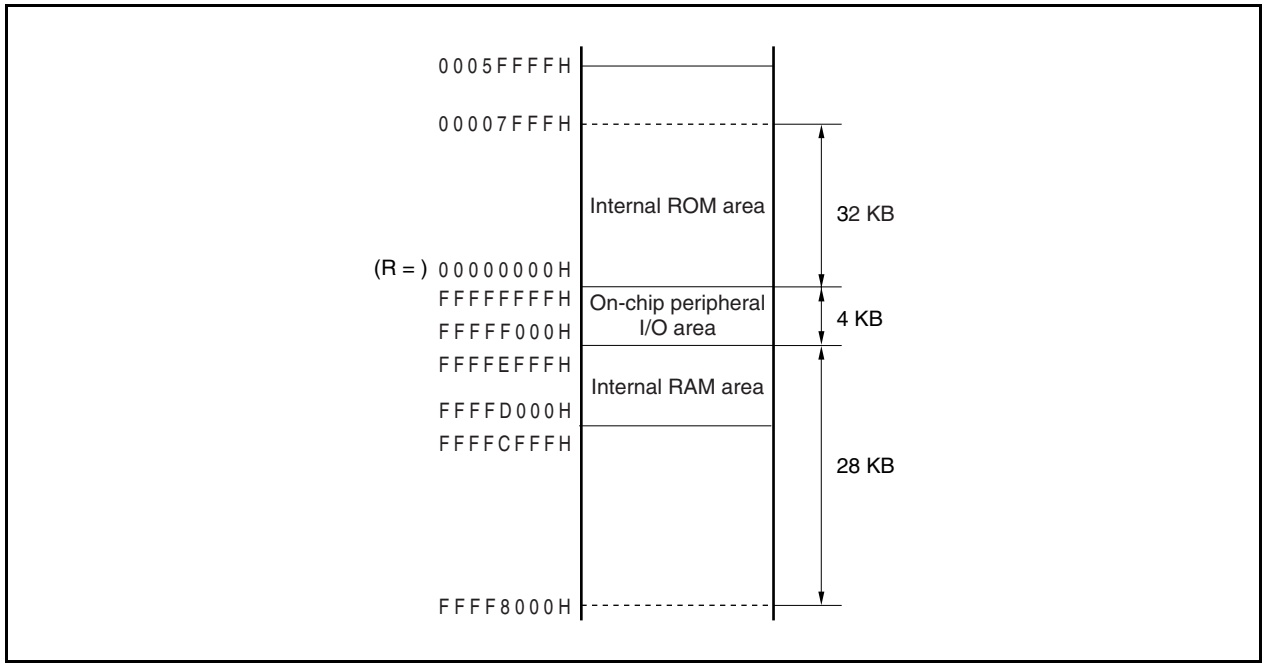
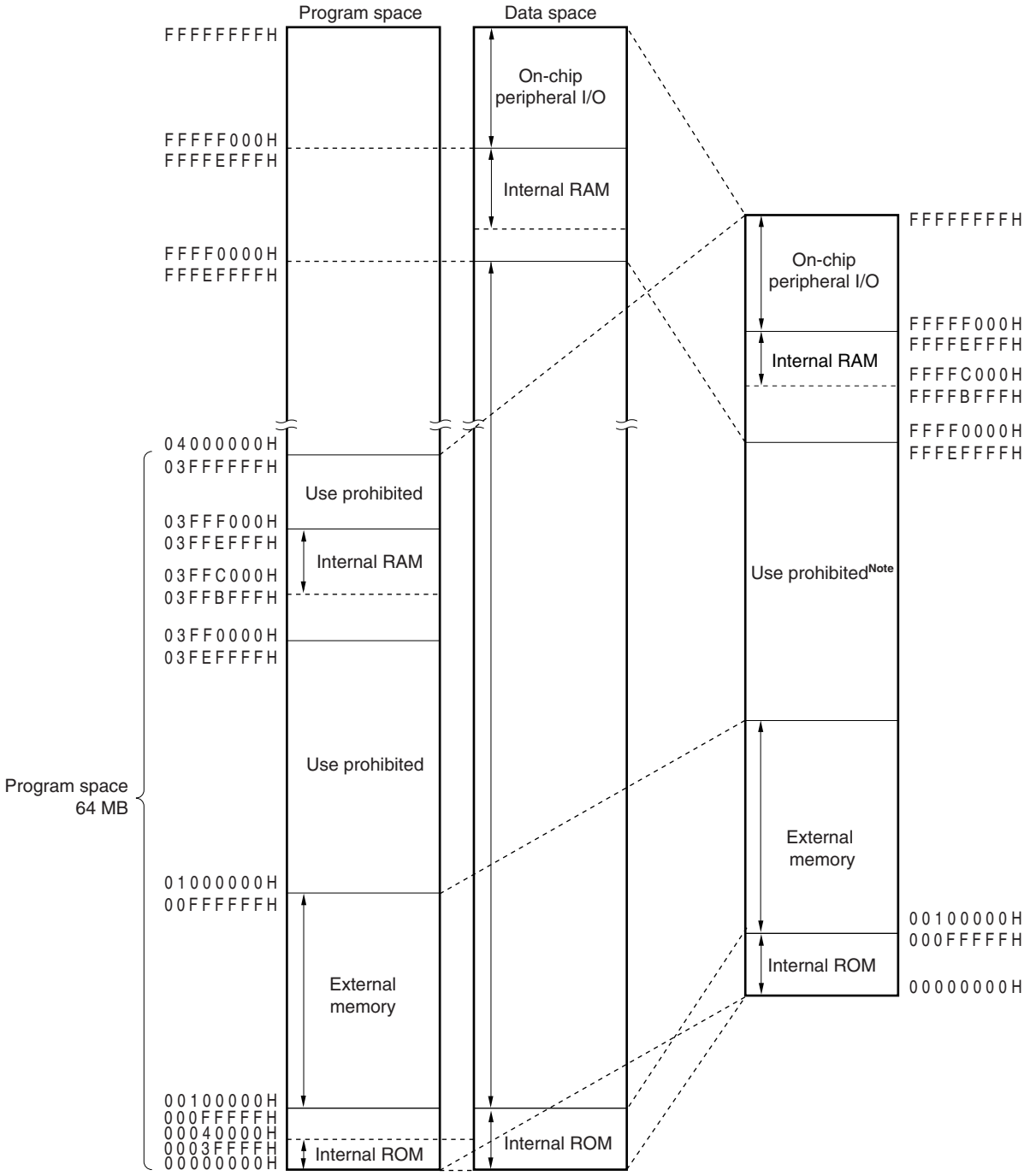


Figure 3-10. Recommended Memory Map



<R>

<R>

Note In the μPD703253Y, the data space of addresses 03FEC000H to 03FEEFFFH is assigned as the programmable peripheral I/O area. Only the programmable peripheral I/O area is seen as images of 256 MB each in the 4 GB address space.

Remarks 1. ↕ indicates the recommended area.

<R>

2. This figure is the recommended memory map of the μPD703249Y.

3.4.6 Peripheral I/O registers

(1/10)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF004H	Port DL register	PDL	R/W			√	0000H[Note 1]
FFFFF004H	Port DL register L	PDLL		√	√		00H[Note 1]
FFFFF005H	Port DL register H	PDLH		√	√		00H[Note 1]
FFFFF006H	Port DH register	PDH		√	√		00H[Note 1]
FFFFF00AH	Port CT register	PCT		√	√		00H[Note 1]
FFFFF00CH	Port CM register	PCM		√	√		00H[Note 1]
FFFFF024H	Port DL mode register	PMDL				√	FFFFH
FFFFF024H	Port DL mode register L	PMDLL		√	√		FFH
FFFFF025H	Port DL mode register H	PMDLH		√	√		FFH
FFFFF026H	Port DH mode register	PMDH		√	√		FFH
FFFFF02AH	Port CT mode register	PMCT		√	√		FFH
FFFFF02CH	Port CM mode register	PMCM		√	√		FFH
FFFFF044H	Port DL mode control register	PMCDL				√	0000H
FFFFF044H	Port DL mode control register L	PMCDLL		√	√		00H
FFFFF045H	Port DL mode control register H	PMCDLH		√	√		00H
FFFFF046H	Port DH mode control register	PMCDH		√	√		00H
FFFFF04AH	Port CT mode control register	PMCCT		√	√		00H
FFFFF04CH	Port CM mode control register	PMCCM		√	√		00H
<R> FFFFF064H	Peripheral I/O area select control register	BPC[Note 2]				√	0000H
FFFFF066H	Bus size configuration register	BSC				√	5555H
FFFFF06EH	System wait control register	VSWC			√		77H
<R> FFFFF080H	DMA source address register 0L	DSA0L[Note 2]				√	Undefined
<R> FFFFF082H	DMA source address register 0H	DSA0H[Note 2]				√	Undefined
<R> FFFFF084H	DMA destination address register 0L	DDA0L[Note 2]				√	Undefined
<R> FFFFF086H	DMA destination address register 0H	DDA0H[Note 2]				√	Undefined
<R> FFFFF088H	DMA source address register 1L	DSA1L[Note 2]				√	Undefined
<R> FFFFF08AH	DMA source address register 1H	DSA1H[Note 2]				√	Undefined
<R> FFFFF08CH	DMA destination address register 1L	DDA1L[Note 2]				√	Undefined
<R> FFFFF08EH	DMA destination address register 1H	DDA1H[Note 2]				√	Undefined
<R> FFFFF090H	DMA source address register 2L	DSA2L[Note 2]				√	Undefined
<R> FFFFF092H	DMA source address register 2H	DSA2H[Note 2]				√	Undefined
<R> FFFFF094H	DMA destination address register 2L	DDA2L[Note 2]				√	Undefined
<R> FFFFF096H	DMA destination address register 2H	DDA2H[Note 2]				√	Undefined
<R> FFFFF098H	DMA source address register 3L	DSA3L[Note 2]				√	Undefined
<R> FFFFF09AH	DMA source address register 3H	DSA3H[Note 2]				√	Undefined
<R> FFFFF09CH	DMA destination address register 3L	DDA3L[Note 2]				√	Undefined
<R> FFFFF09EH	DMA destination address register 3H	DDA3H[Note 2]				√	Undefined
<R> FFFFF0C0H	DMA transfer count register 0	DBC0[Note 2]				√	Undefined
<R> FFFFF0C2H	DMA transfer count register 1	DBC1[Note 2]				√	Undefined
<R> FFFFF0C4H	DMA transfer count register 2	DBC2[Note 2]				√	Undefined

Notes 1. The output latch is 00H or 0000H. When these registers are in the input mode, the pin statuses are read.

<R> **2.** μPD703253Y only

(2/10)

	Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
					1	8	16	
<R>	FFFFF0C6H	DMA transfer count register 3	DBC3[Note]	R/W			√	Undefined
<R>	FFFFF0D0H	DMA addressing control register 0	DADC0[Note]				√	0000H
<R>	FFFFF0D2H	DMA addressing control register 1	DADC1[Note]				√	0000H
<R>	FFFFF0D4H	DMA addressing control register 2	DADC2[Note]				√	0000H
<R>	FFFFF0D6H	DMA addressing control register 3	DADC3[Note]				√	0000H
<R>	FFFFF0E0H	DMA channel control register 0	DCHC0[Note]		√	√		00H
<R>	FFFFF0E2H	DMA channel control register 1	DCHC1[Note]		√	√		00H
<R>	FFFFF0E4H	DMA channel control register 2	DCHC2[Note]		√	√		00H
<R>	FFFFF0E6H	DMA channel control register 3	DCHC3[Note]		√	√		00H
	FFFFF100H	Interrupt mask register 0	IMR0				√	FFFFH
	FFFFF100H	Interrupt mask register 0L	IMR0L		√	√		FFH
	FFFFF101H	Interrupt mask register 0H	IMR0H		√	√		FFH
	FFFFF102H	Interrupt mask register 1	IMR1				√	FFFFH
	FFFFF102H	Interrupt mask register 1L	IMR1L		√	√		FFH
	FFFFF103H	Interrupt mask register 1H	IMR1H		√	√		FFH
	FFFFF104H	Interrupt mask register 2	IMR2				√	FFFFH
	FFFFF104H	Interrupt mask register 2L	IMR2L		√	√		FFH
	FFFFF105H	Interrupt mask register 2H	IMR2H		√	√		FFH
	FFFFF106H	Interrupt mask register 3	IMR3				√	FFFFH
	FFFFF106H	Interrupt mask register 3L	IMR3L		√	√		FFH
	FFFFF107H	Interrupt mask register 3H	IMR3H		√	√		FFH
	FFFFF112H	Interrupt control register	PIC0		√	√		47H
	FFFFF114H	Interrupt control register	PIC1		√	√		47H
	FFFFF116H	Interrupt control register	PIC2		√	√		47H
	FFFFF118H	Interrupt control register	PIC3		√	√		47H
	FFFFF11AH	Interrupt control register	PIC4		√	√		47H
	FFFFF11CH	Interrupt control register	PIC5		√	√		47H
	FFFFF11EH	Interrupt control register	PIC6		√	√		47H
	FFFFF120H	Interrupt control register	PIC7		√	√		47H
	FFFFF12CH	Interrupt control register	TP0OVIC		√	√		47H
	FFFFF12EH	Interrupt control register	TP0CCIC0		√	√		47H
	FFFFF130H	Interrupt control register	TP0CCIC1		√	√		47H
	FFFFF132H	Interrupt control register	TP1OVIC		√	√		47H
	FFFFF134H	Interrupt control register	TP1CCIC0		√	√		47H
	FFFFF136H	Interrupt control register	TP1CCIC1		√	√		47H
	FFFFF138H	Interrupt control register	TP2OVIC		√	√		47H
	FFFFF13AH	Interrupt control register	TP2CCIC0		√	√		47H
	FFFFF13CH	Interrupt control register	TP2CCIC1		√	√		47H
	FFFFF13EH	Interrupt control register	TP3OVIC		√	√		47H
	FFFFF140H	Interrupt control register	TP3CCIC0		√	√		47H
	FFFFF142H	Interrupt control register	TP3CCIC1		√	√		47H
	FFFFF144H	Interrupt control register	TP4OVIC		√	√		47H

<R> **Note** μPD703253Y only

(3/10)

	Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
					1	8	16	
	FFFFF146H	Interrupt control register	TP4CCIC0	R/W	√	√		47H
	FFFFF148H	Interrupt control register	TP4CCIC1		√	√		47H
<R>	FFFFF14AH	Interrupt control register	TP5OVIC[Note 1]		√	√		47H
<R>	FFFFF14CH	Interrupt control register	TP5CCIC0[Note 1]		√	√		47H
<R>	FFFFF14EH	Interrupt control register	TP5CCIC1[Note 1]		√	√		47H
	FFFFF150H	Interrupt control register	TM0EQIC0		√	√		47H
	FFFFF152H	Interrupt control register	CB0RIC/IICIC1		√	√		47H
	FFFFF154H	Interrupt control register	CB0TIC		√	√		47H
	FFFFF156H	Interrupt control register	CB1RIC		√	√		47H
	FFFFF158H	Interrupt control register	CB1TIC		√	√		47H
	FFFFF15EH	Interrupt control register	CB3RIC		√	√		47H
	FFFFF160H	Interrupt control register	CB3TIC		√	√		47H
	FFFFF162H	Interrupt control register	UA0RIC		√	√		47H
	FFFFF164H	Interrupt control register	UA0TIC		√	√		47H
	FFFFF166H	Interrupt control register	UA1RIC		√	√		47H
	FFFFF168H	Interrupt control register	UA1TIC		√	√		47H
	FFFFF16AH	Interrupt control register	IICIC0		√	√		47H
	FFFFF16EH	Interrupt control register	ADIC		√	√		47H
<R>	FFFFF170H	Interrupt control register	DMAIC0[Note 1]		√	√		47H
<R>	FFFFF172H	Interrupt control register	DMAIC1[Note 1]		√	√		47H
<R>	FFFFF174H	Interrupt control register	DMAIC2[Note 1]		√	√		47H
<R>	FFFFF176H	Interrupt control register	DMAIC3[Note 1]		√	√		47H
	FFFFF178H	Interrupt control register	KRIC		√	√		47H
	FFFFF17AH	Interrupt control register	WTIIC		√	√		47H
	FFFFF17CH	Interrupt control register	WTIC		√	√		47H
<R>	FFFFF17EH	Interrupt control register	ERRIC0[Note 1]/ERRIC[Note 2]		√	√		47H
<R>	FFFFF180H	Interrupt control register	WUPIC0[Note 1]/STAIC[Note 2]		√	√		47H
<R>	FFFFF182H	Interrupt control register	RECIC0[Note 1]/IEIC1[Note 2]		√	√		47H
<R>	FFFFF184H	Interrupt control register	TRXIC0[Note 1]/IEIC2[Note 2]		√	√		47H
	FFFFF1FAH	In-service priority register	ISPR	R	√	√		00H
	FFFFF1FCH	Command register	PRCMD	W		√		Undefined
	FFFFF1FEH	Power save control register	PSC	R/W	√	√		00H
	FFFFF200H	A/D converter mode register 0	ADA0M0		√	√		00H
	FFFFF201H	A/D converter mode register 1	ADA0M1		√	√		00H
	FFFFF202H	A/D converter channel specification register	ADA0S		√	√		00H
	FFFFF203H	A/D converter mode register 2	ADA0M2		√	√		00H
	FFFFF204H	Power-fail compare mode register	ADA0PFM		√	√		00H
	FFFFF205H	Power-fail compare threshold value register	ADA0PFT		√	√		00H

<R> **Notes 1.** μPD703253Y only

2. μPD703252Y only

(4/10)

	Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
					1	8	16	
	FFFFF210H	A/D conversion result register 0	ADA0CR0	R			√	Undefined
	FFFFF211H	A/D conversion result register 0H	ADA0CR0H			√		Undefined
	FFFFF212H	A/D conversion result register 1	ADA0CR1				√	Undefined
	FFFFF213H	A/D conversion result register 1H	ADA0CR1H			√		Undefined
	FFFFF214H	A/D conversion result register 2	ADA0CR2				√	Undefined
	FFFFF215H	A/D conversion result register 2H	ADA0CR2H			√		Undefined
	FFFFF216H	A/D conversion result register 3	ADA0CR3				√	Undefined
	FFFFF217H	A/D conversion result register 3H	ADA0CR3H			√		Undefined
	FFFFF218H	A/D conversion result register 4	ADA0CR4				√	Undefined
	FFFFF219H	A/D conversion result register 4H	ADA0CR4H			√		Undefined
	FFFFF21AH	A/D conversion result register 5	ADA0CR5				√	Undefined
	FFFFF21BH	A/D conversion result register 5H	ADA0CR5H			√		Undefined
	FFFFF21CH	A/D conversion result register 6	ADA0CR6				√	Undefined
	FFFFF21DH	A/D conversion result register 6H	ADA0CR6H			√		Undefined
	FFFFF21EH	A/D conversion result register 7	ADA0CR7				√	Undefined
	FFFFF21FH	A/D conversion result register 7H	ADA0CR7H			√		Undefined
	FFFFF220H	A/D conversion result register 8	ADA0CR8				√	Undefined
	FFFFF221H	A/D conversion result register 8H	ADA0CR8H			√		Undefined
	FFFFF222H	A/D conversion result register 9	ADA0CR9				√	Undefined
	FFFFF223H	A/D conversion result register 9H	ADA0CR9H			√		Undefined
	FFFFF224H	A/D conversion result register 10	ADA0CR10				√	Undefined
	FFFFF225H	A/D conversion result register 10H	ADA0CR10H			√		Undefined
	FFFFF226H	A/D conversion result register 11	ADA0CR11				√	Undefined
	FFFFF227H	A/D conversion result register 11H	ADA0CR11H			√		Undefined
<R>	FFFFF280H	D/A converter conversion value setting register 0	DA0CS0[Note]	R/W		√		00H
<R>	FFFFF281H	D/A converter conversion value setting register 1	DA0CS1[Note]			√		00H
<R>	FFFFF282H	D/A converter mode register	DA0M[Note]		√	√		00H
	FFFFF300H	Key return mode register	KRM		√	√		00H
	FFFFF318H	Noise elimination control register	NFC			√		00H
	FFFFF320H	BRG1 prescaler mode register	PRSM1		√	√		00H
	FFFFF321H	BRG1 prescaler compare register	PRSCM1			√		00H
	FFFFF340H	IIC division clock select register	OCKS0			√		00H
	FFFFF344H	IIC division clock select register	OCKS1			√		00H
	FFFFF348H	IEBus clock select register	OCKS2[Note 1]			√		00H
	FFFFF360H	IEBus control register	BCR[Note 1]		√	√		00H
	FFFFF361H	IEBus power save register	PSR[Note 1]		√	√		00H
	FFFFF362H	IEBus slave status register	SSR[Note 1]	R	√	√		81H
	FFFFF363H	IEBus unit status register	USR[Note 1]		√	√		00H
	FFFFF364H	IEBus interrupt status register	ISR[Note 1]	R/W	√	√		00H
	FFFFF365H	IEBus error status register	ESR[Note 1]		√	√		00H
	FFFFF366H	IEBus unit address register	UAR[Note 1]				√	0000H

<R> **Note** μPD703253Y only

(5/10)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF368H	IEBus slave address register	SAR[Note 1]	R/W			√	0000H
FFFFF36AH	IEBus partner address register	PAR[Note 1]	R			√	0000H
FFFFF36CH	IEBus receive address register	PSR[Note 1]				√	0000H
FFFFF36EH	IEBus control data register	CDR[Note 1]	R/W		√		00H
FFFFF36FH	IEBus telegraph length register	CLR[Note 1]			√		01H
FFFFF370H	IEBus data register	DR[Note 1]			√		00H
FFFFF371H	IEBus field status register	FSR[Note 1]			√		00H
FFFFF372H	IEBus success count register	SCR[Note 1]			√		01H
FFFFF373H	IEBus communication count register	CCR[Note 1]			√		20H
FFFFF400H	Port 0 register	P0		√	√		00H[Note 2]
FFFFF402H	Port 1 register	P1		√	√		00H[Note 2]
FFFFF406H	Port 3 register	P3				√	0000H[Note 2]
FFFFF406H	Port 3 register L	P3L		√	√		00H[Note 2]
FFFFF407H	Port 3 register H	P3H		√	√		00H[Note 2]
FFFFF408H	Port 4 register	P4		√	√		00H[Note]
FFFFF40AH	Port 5 register	P5		√	√		00H[Note]
FFFFF40EH	Port 7 register L	P7L		√	√		00H[Note]
FFFFF40FH	Port 7 register H	P7H		√	√		00H[Note 2]
FFFFF412H	Port 9 register	P9				√	0000H[Note 2]
FFFFF412H	Port 9 register L	P9L		√	√		00H[Note 2]
FFFFF413H	Port 9 register H	P9H		√	√		00H[Note 2]
FFFFF420H	Port 0 mode register	PM0		√	√		FFH
FFFFF422H	Port 1 mode register	PM1		√	√		FFH
FFFFF426H	Port 3 mode register	PM3				√	FFFFH
FFFFF426H	Port 3 mode register L	PM3L		√	√		FFH
FFFFF427H	Port 3 mode register H	PM3H		√	√		FFH
FFFFF428H	Port 4 mode register	PM4		√	√		FFH
FFFFF42AH	Port 5 mode register	PM5		√	√		FFH
FFFFF42EH	Port 7 mode register L	PM7L		√	√		FFH
FFFFF42FH	Port 7 mode register H	PM7H		√	√		FFH
FFFFF432H	Port 9 mode register	PM9				√	FFFFH
FFFFF432H	Port 9 mode register L	PM9L		√	√		FFH
FFFFF433H	Port 9 mode register H	PM9H		√	√		FFH
FFFFF440H	Port 0 mode control register	PMC0		√	√		00H
FFFFF446H	Port 3 mode control register	PMC3				√	0000H
FFFFF446H	Port 3 mode control register L	PMC3L		√	√		00H
FFFFF447H	Port 3 mode control register H	PMC3H		√	√		00H
FFFFF448H	Port 4 mode control register	PMC4		√	√		00H

Notes 1. μPD703252Y only

2. The output latch is 00H or 0000H. When these registers are input, the pin statuses are read.

(6/10)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF44AH	Port 5 mode control register	PMC5	R/W	√	√		00H
FFFFF452H	Port 9 mode control register	PMC9				√	0000H
FFFFF452H	Port 9 mode control register L	PMC9L		√	√		00H
FFFFF453H	Port 9 mode control register H	PMC9H		√	√		00H
FFFFF460H	Port 0 function control register	PFC0		√	√		00H
FFFFF466H	Port 3 function control register	PFC3				√	0000H
FFFFF466H	Port 3 function control register L	PFC3L		√	√		00H
FFFFF467H	Port 3 function control register H	PFC3H		√	√		00H
FFFFF468H	Port 4 function control register	PFC4		√	√		00H
FFFFF46AH	Port 5 function control register	PFC5		√	√		00H
FFFFF472H	Port 9 function control register	PFC9				√	0000H
FFFFF472H	Port 9 function control register L	PFC9L		√	√		00H
FFFFF473H	Port 9 function control register H	PFC9H		√	√		00H
FFFFF484H	Data wait control register 0	DWC0				√	7777H
FFFFF488H	Address wait control register	AWC				√	FFFFH
FFFFF48AH	Bus cycle control register	BCC				√	AAAAH
FFFFF590H	TMP0 control register 0	TP0CTL0		√	√		00H
FFFFF591H	TMP0 control register 1	TP0CTL1		√	√		00H
FFFFF592H	TMP0 I/O control register 0	TP0IOC0		√	√		00H
FFFFF593H	TMP0 I/O control register 1	TP0IOC1		√	√		00H
FFFFF594H	TMP0 I/O control register 2	TP0IOC2		√	√		00H
FFFFF595H	TMP0 option register 0	TP0OPT0		√	√		00H
FFFFF596H	TMP0 capture/compare register 0	TP0CCR0				√	0000H
FFFFF598H	TMP0 capture/compare register 1	TP0CCR1				√	0000H
FFFFF59AH	TMP0 counter read buffer register	TP0CNT	R			√	0000H
FFFFF5A0H	TMP1 control register 0	TP1CTL0	R/W	√	√		00H
FFFFF5A1H	TMP1 control register 1	TP1CTL1		√	√		00H
FFFFF5A2H	TMP1 I/O control register 0	TP1IOC0		√	√		00H
FFFFF5A3H	TMP1 I/O control register 1	TP1IOC1		√	√		00H
FFFFF5A4H	TMP1 I/O control register 2	TP1IOC2		√	√		00H
FFFFF5A5H	TMP1 option register 0	TP1OPT0		√	√		00H
FFFFF5A6H	TMP1 capture/compare register 0	TP1CCR0				√	0000H
FFFFF5A8H	TMP1 capture/compare register 1	TP1CCR1				√	0000H
FFFFF5AAH	TMP1 counter read buffer register	TP1CNT	R			√	0000H
FFFFF5B0H	TMP2 control register 0	TP2CTL0	R/W	√	√		00H
FFFFF5B1H	TMP2 control register 1	TP2CTL1		√	√		00H
FFFFF5B2H	TMP2 I/O control register 0	TP2IOC0		√	√		00H
FFFFF5B3H	TMP2 I/O control register 1	TP2IOC1		√	√		00H
FFFFF5B4H	TMP2 I/O control register 2	TP2IOC2		√	√		00H
FFFFF5B5H	TMP2 option register 0	TP2OPT0		√	√		00H
FFFFF5B6H	TMP2 capture/compare register 0	TP2CCR0				√	0000H
FFFFF5B8H	TMP2 capture/compare register 1	TP2CCR1				√	0000H
FFFFF5BAH	TMP2 counter read buffer register	TP2CNT	R			√	0000H

(7/10)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFF5C0H	TMP3 control register 0	TP3CTL0	R/W	√	√		00H
FFFFF5C1H	TMP3 control register 1	TP3CTL1		√	√		00H
FFFFF5C2H	TMP3 I/O control register 0	TP3IOC0		√	√		00H
FFFFF5C3H	TMP3 I/O control register 1	TP3IOC1		√	√		00H
FFFFF5C4H	TMP3 I/O control register 2	TP3IOC2		√	√		00H
FFFFF5C5H	TMP3 option register 0	TP3OPT0		√	√		00H
FFFFF5C6H	TMP3 capture/compare register 0	TP3CCR0				√	0000H
FFFFF5C8H	TMP3 capture/compare register 1	TP3CCR1				√	0000H
FFFFF5CAH	TMP3 counter read buffer register	TP3CNT	R			√	0000H
FFFFF5D0H	TMP4 control register 0	TP4CTL0	R/W	√	√		00H
FFFFF5D1H	TMP4 control register 1	TP4CTL1		√	√		00H
FFFFF5D2H	TMP4 I/O control register 0	TP4IOC0		√	√		00H
FFFFF5D3H	TMP4 I/O control register 1	TP4IOC1		√	√		00H
FFFFF5D4H	TMP4 I/O control register 2	TP4IOC2		√	√		00H
FFFFF5D5H	TMP4 option register 0	TP4OPT0		√	√		00H
FFFFF5D6H	TMP4 capture/compare register 0	TP4CCR0				√	0000H
FFFFF5D8H	TMP4 capture/compare register 1	TP4CCR1				√	0000H
FFFFF5DAH	TMP4 counter read buffer register	TP4CNT	R			√	0000H
<R> FFFFF5E0H	TMP5 control register 0	TP5CTL0[Note]	R/W	√	√		00H
<R> FFFFF5E1H	TMP5 control register 1	TP5CTL1[Note]		√	√		00H
<R> FFFFF5E2H	TMP5 I/O control register 0	TP5IOC0[Note]		√	√		00H
<R> FFFFF5E3H	TMP5 I/O control register 1	TP5IOC1[Note]		√	√		00H
<R> FFFFF5E4H	TMP5 I/O control register 2	TP5IOC2[Note]		√	√		00H
<R> FFFFF5E5H	TMP5 option register 0	TP5OPT0[Note]		√	√		00H
<R> FFFFF5E6H	TMP5 capture/compare register 0	TP5CCR0[Note]				√	0000H
<R> FFFFF5E8H	TMP5 capture/compare register 1	TP5CCR1[Note]				√	0000H
<R> FFFFF5EAH	TMP5 counter read buffer register	TP5CNT[Note]	R			√	0000H
FFFFF680H	Watch timer operation mode register	WTM	R/W	√	√		00H
FFFFF690H	TMM0 control register 0	TM0CTL0		√	√		00H
FFFFF694H	TMM0 compare register 0	TM0CMP0				√	0000H
FFFFF6C0H	Oscillation stabilization time select register	OSTS			√		06H
FFFFF6C1H	PLL lockup time specification register	PLLS			√		03H
FFFFF6D0H	Watchdog timer mode register 2	WDTM2			√		67H
FFFFF6D1H	Watchdog timer enable register	WDTE			√		9AH
FFFFF706H	Port 3 function control expansion register L	PFCE3L		√	√		00H
FFFFF70AH	Port 5 function control expansion register	PFCE5		√	√		00H
FFFFF712H	Port 9 function control expansion register	PFCE9				√	0000H
FFFFF712H	Port 9 function control expansion register L	PFCE9L		√	√		00H
FFFFF713H	Port 9 function control expansion register H	PFCE9H		√	√		00H
FFFFF802H	System status register	SYS		√	√		00H
FFFFF80CH	Internal oscillation mode register	RCM		√	√		00H

<R> **Note** μPD703253Y only

(8/10)

	Address	Function Register Name	Symbol	R/W	Manipulatable Bits				Default Value
					1	8	16	32	
<R>	FFFFF810H	DMA trigger factor register 0	DTFR0[Note 1]	R/W	√	√			00H
<R>	FFFFF812H	DMA trigger factor register 1	DTFR1[Note 1]		√	√			00H
<R>	FFFFF814H	DMA trigger factor register 2	DTFR2[Note 1]		√	√			00H
<R>	FFFFF816H	DMA trigger factor register 3	DTFR3[Note 1]		√	√			00H
	FFFFF820H	Power save mode register	PSMR		√	√			00H
	FFFFF822H	Clock control register	CKC		√	√			0AH
	FFFFF824H	Lock register	LOCKR	R	√	√			00H
	FFFFF828H	Processor clock control register	PCC	R/W	√	√			03H
	FFFFF82CH	PLL control register	PLLCTL		√	√			01H
	FFFFF82EH	CPU operation clock status register	CCLS	R	√	√			00H
	FFFFF840H	Correction address register 0	CORAD0	R/W				√	00000000H
	FFFFF840H	Correction address register 0L	CORAD0L				√		0000H
	FFFFF842H	Correction address register 0H	CORAD0H				√		0000H
	FFFFF844H	Correction address register 1	CORAD1					√	00000000H
	FFFFF844H	Correction address register 1L	CORAD1L				√		0000H
	FFFFF846H	Correction address register 1H	CORAD1H				√		0000H
	FFFFF848H	Correction address register 2	CORAD2					√	00000000H
	FFFFF848H	Correction address register 2L	CORAD2L				√		0000H
	FFFFF84AH	Correction address register 2H	CORAD2H				√		0000H
	FFFFF84CH	Correction address register 3	CORAD3					√	00000000H
	FFFFF84CH	Correction address register 3L	CORAD3L				√		0000H
	FFFFF84EH	Correction address register 3H	CORAD3H				√		0000H
	FFFFF870H	Clock monitor mode register	CLM		√	√			00H
	FFFFF880H	Correction control register	CORCN		√	√			00H
	FFFFF888H	Reset source flag register	RESF		√	√			00H
<R>	FFFFF892H	Internal RAM data status register	RAMS[Note 1]		√	√			01H
	FFFFF8B0H	Prescaler mode register 0	PRSM0		√	√			00H
	FFFFF8B1H	Prescaler compare register 0	PRSCM0			√			00H
	FFFFF9FCH	On-chip debug mode register	OCDM		√	√			01H
	FFFFF9FEH	Peripheral emulation register 1	PEMU1[Notes 1, 2]		√	√			00H
	FFFFFA00H	UARTA0 control register 0	UA0CTL0		√	√			10H
	FFFFFA01H	UARTA0 control register 1	UA0CTL1			√			00H
	FFFFFA02H	UARTA0 control register 2	UA0CTL2			√			FFH
	FFFFFA03H	UARTA0 option control register 0	UA0OPT0		√	√			14H
	FFFFFA04H	UARTA0 status register	UA0STR		√	√			00H
	FFFFFA06H	UARTA0 receive data register	UA0RX	R		√			FFH
	FFFFFA07H	UARTA0 transmit data register	UA0TX	R/W		√			FFH
	FFFFFA10H	UARTA1 control register 0	UA1CTL0		√	√			10H
	FFFFFA11H	UARTA1 control register 1	UA1CTL1			√			00H
	FFFFFA12H	UARTA1 control register 2	UA1CTL2			√			FFH
	FFFFFA13H	UARTA1 option control register 0	UA1OPT0		√	√			14H

<R> **Notes 1.** μPD703253Y only

2. Only during emulation

(9/10)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFFA14H	UARTA1 status register	UA1STR	R/W	√	√		00H
FFFFFA16H	UARTA1 receive data register	UA1RX	R		√		FFH
FFFFFA17H	UARTA1 transmit data register	UA1TX	R/W		√		FFH
FFFFFC00H	External interrupt falling edge specification register 0	INTF0		√	√		00H
FFFFFC06H	External interrupt falling edge specification register 3	INTF3		√	√		00H
FFFFFC13H	External interrupt falling edge specification register 9H	INTF9H		√	√		00H
FFFFFC20H	External interrupt rising edge specification register 0	INTR0		√	√		00H
FFFFFC26H	External interrupt rising edge specification register 3	INTR3		√	√		00H
FFFFFC33H	External interrupt rising edge specification register 9H	INTR9H		√	√		00H
FFFFFC60H	Port 0 function register	PF0		√	√		00H
FFFFFC66H	Port 3 function register	PF3				√	0000H
FFFFFC66H	Port 3 function register L	PF3L		√	√		00H
FFFFFC67H	Port 3 function register H	PF3H		√	√		00H
FFFFFC68H	Port 4 function register	PF4		√	√		00H
FFFFFC6AH	Port 5 function register	PF5		√	√		00H
FFFFFC72H	Port 9 function register	PF9				√	0000H
FFFFFC72H	Port 9 function register L	PF9L		√	√		00H
FFFFFC73H	Port 9 function register H	PF9H		√	√		00H
FFFFFD00H	CSIB0 control register 0	CB0CTL0		√	√		01H
FFFFFD01H	CSIB0 control register 1	CB0CTL1		√	√		00H
FFFFFD02H	CSIB0 control register 2	CB0CTL2			√		00H
FFFFFD03H	CSIB0 status register	CB0STR		√	√		00H
FFFFFD04H	CSIB0 receive data register	CB0RX	R			√	0000H
FFFFFD04H	CSIB0 receive data register L	CB0RXL			√		00H
FFFFFD06H	CSIB0 transmit data register	CB0TX	R/W			√	0000H
FFFFFD06H	CSIB0 transmit data register L	CB0TXL			√		00H
FFFFFD03H	CSIB0 status register	CB0STR		√	√		00H
FFFFFD10H	CSIB1 control register 0	CB1CTL0		√	√		01H
FFFFFD11H	CSIB1 control register 1	CB1CTL1		√	√		00H
FFFFFD12H	CSIB1 control register 2	CB1CTL2			√		00H
FFFFFD13H	CSIB1 status register	CB1STR		√	√		00H
FFFFFD14H	CSIB1 receive data register	CB1RX	R			√	0000H
FFFFFD14H	CSIB1 receive data register L	CB1RXL			√		00H
FFFFFD16H	CSIB1 transmit data register	CB1TX	R/W			√	0000H
FFFFFD16H	CSIB1 transmit data register L	CB1TXL			√		00H
FFFFFD30H	CSIB3 control register 0	CB3CTL0		√	√		01H
FFFFFD31H	CSIB3 control register 1	CB3CTL1		√	√		00H
FFFFFD32H	CSIB3 control register 2	CB3CTL2			√		00H
FFFFFD33H	CSIB3 status register	CB3STR		√	√		00H
FFFFFD34H	CSIB3 receive data register	CB3RX	R			√	0000H
FFFFFD34H	CSIB3 receive data register L	CB3RXL			√		00H

(10/10)

Address	Function Register Name	Symbol	R/W	Manipulatable Bits			Default Value
				1	8	16	
FFFFFD36H	CSIB3 transmit data register	CB3TX	R/W			√	0000H
FFFFFD36H	CSIB3 transmit data register L	CB3TXL			√		00H
FFFFFD80H	IIC shift register 0	IIC0			√		00H
FFFFFD82H	IIC control register 0	IICC0		√	√		00H
FFFFFD83H	Slave address register 0	SVA0			√		00H
FFFFFD84H	IIC clock select register 0	IICCL0		√	√		00H
FFFFFD85H	IIC function expansion register 0	IICX0		√	√		00H
FFFFFD86H	IIC status register 0	IICS0	R	√	√		00H
FFFFFD8AH	IIC flag register 0	IICF0	R/W	√	√		00H
FFFFFD90H	IIC shift register 1	IIC1			√		00H
FFFFFD92H	IIC control register 1	IICC1		√	√		00H
FFFFFD93H	Slave address register 1	SVA1			√		00H
FFFFFD94H	IIC clock select register 1	IICCL1		√	√		00H
FFFFFD95H	IIC function expansion register 1	IICX1		√	√		00H
FFFFFD96H	IIC status register 1	IICS1	R	√	√		00H
FFFFFD9AH	IIC flag register 1	IICF1	R/W	√	√		00H
FFFFFDBEH	External bus interface mode control register	EXIMC		√	√		00H

<R> **3.4.7 Programmable peripheral I/O registers**

The BPC register is used for programmable peripheral I/O register area selection.

(1) Peripheral I/O area select control register (BPC)

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

Reset sets this register to 0000H.

The BPC register is provided only in the μPD703253Y.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Default value
BPC	PA15	0	PA13	PA12	PA11	PA10	PA09	PA08	PA07	PA06	PA05	PA04	PA03	PA02	PA01	PA00	FFFFF064H	0000H

Bit position	Bit name	Function
15	PA15	Enables/disables usage of programmable peripheral I/O area. PA15: 0 — Usage of programmable peripheral I/O area disabled PA15: 1 — Usage of programmable peripheral I/O area enabled
13 to 0	PA13 to PA00	Specify an address in programmable peripheral I/O area (corresponding to A27 to A14, respectively).

Caution When setting the PA15 bit to 1, be sure to set the BPC register to 8FFBH.
When clearing the PA15 bit to 0, be sure to set the BPC register to 0000H.

For a list of the programmable peripheral I/O register areas, see **Table 17-16 Register Access Types**.

3.4.8 Special registers

Special registers are registers that are protected from being written with illegal data due to a program loop. Six
<R> special registers are provided in the μPD703249Y and 703252Y, and seven special registers in the μPD703253Y.

- Power save control register (PSC)
- Clock control register (CKC)
- Processor clock control register (PCC)
- Clock monitor mode register (CLM)
- Reset source flag register (RESF)
<R> • Internal RAM data status register (RAMS)[Note]
- On-chip debug mode register (OCDM)

<R> **Note** μPD703253Y only

In addition, the PRCDM register is provided to protect against a write access to the special registers so that the application system does not inadvertently stop due to a program loop. A write access to the special registers is made in a specific sequence, and an illegal store operation is reported to the SYS register.

(1) Setting data to special registers

Set data to the special registers in the following sequence.

<R> <1> Disable DMA operation.[Note 1]

<2> Prepare data to be set to the special register in a general-purpose register.

<3> Write the data prepared in <2> to the PRCMD register.

<4> Write the setting data to the special register (by using the following instructions).

- Store instruction (ST/SST instruction)
- Bit manipulation instruction (SET1/CLR1/NOT1 instruction)

(<5> to <9> Insert NOP instructions (5 instructions).)[Note 2]

<R> <10> Enable DMA operation if necessary.[Note 1]

[Example] With PSC register (setting standby mode)

```
    ST.B r11, PSMR[r0]    ; Set PSMR register (setting IDLE1, IDLE2, and STOP modes).
<R> <1>CLR1 0, DCHCn[r0]  ; Disable DMA operation.  n = 0 to 3[Note 1]
    <2>MOV0x02, r10
    <3>ST.B r10, PRCMD[r0] ; Write PRCMD register.
    <4>ST.B r10, PSC[r0]  ; Set PSC register.
    <5>NOP[Note 2]        ; Dummy instruction
    <6>NOP[Note 2]        ; Dummy instruction
    <7>NOP[Note 2]        ; Dummy instruction
    <8>NOP[Note 2]        ; Dummy instruction
    <9>NOP[Note 2]        ; Dummy instruction
<R> <10>SET1 0, DCHCn[r0] ; Enable DMA operation.  n = 0 to 3[Note 1]
    (next instruction)
```

There is no special sequence to read a special register.

<R> **Notes 1.** μPD703253Y only

2. Five NOP instructions or more must be inserted immediately after setting the IDLE1 mode, IDLE2 mode, or STOP mode (by setting the PSC.STP bit to 1).

Cautions 1. When a store instruction is executed to store data in the command register, interrupts are not acknowledged. This is because it is assumed that steps <3> and <4> above are performed by successive store instructions. If another instruction is placed between <3> and <4>, and if an interrupt is acknowledged by that instruction, the above sequence may not be established, causing malfunction.

2. Although dummy data is written to the PRCMD register, use the same general-purpose register used to set the special register (<4> in Example) to write data to the PRCMD register (<3> in Example). The same applies when a general-purpose register is used for addressing.

(2) Command register (PRCMD)

The PRCMD register is an 8-bit register that protects the registers that may seriously affect the application system from being written, so that the system does not inadvertently stop due to a program hang-up. The first write access to a special register is valid after data has been written in advance to the PRCMD register. In this way, the value of the special register can be rewritten only in a specific sequence, so as to protect the register from an illegal write access.

The PRCMD register is write-only, in 8-bit units (undefined data is read when this register is read).

Reset makes this register undefined.

After reset: Undefined W Address: FFFFF1FCH

	7	6	5	4	3	2	1	0
PRCMD	REG7	REG6	REG5	REG4	REG3	REG2	REG1	REG0

(3) System status register (SYS)

Status flags that indicate the operation status of the overall system are allocated to this register.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF802H

	7	6	5	4	3	2	1	<0>
SYS	0	0	0	0	0	0	0	PRERR

PRERR	Detects protection error
0	Protection error did not occur
1	Protection error occurred

The PRERR flag operates under the following conditions.

(a) Set condition (PRERR flag = 1)

(i) When data is written to a special register without writing anything to the PRCMD register (when <4> is executed without executing <3> in **3.4.8 (1) Setting data to special registers**)

(ii) When data is written to an on-chip peripheral I/O register other than a special register (including execution of a bit manipulation instruction) after writing data to the PRCMD register (if <4> in **3.4.8 (1) Setting data to special registers** is not the setting of a special register) <R>

Remark Even if an on-chip peripheral I/O register is read (except by a bit manipulation instruction) between an operation to write the PRCMD register and an operation to write a special register, the PRERR flag is not set, and the set data can be written to the special register.

(b) Clear condition (PRERR flag = 0)

(i) When 0 is written to the PRERR flag

(ii) When the system is reset

Cautions 1. If 0 is written to the PRERR bit of the SYS register, which is not a special register, immediately after a write access to the PRCMD register, the PRERR bit is cleared to 0 (the write access takes precedence).

2. If data is written to the PRCMD register, which is not a special register, immediately after a write access to the PRCMD register, the PRERR bit is set to 1.

3.4.9 Cautions

(1) Registers to be set first

Be sure to set the following registers first when using the V850ES/SG1.

- System wait control register (VSWC)
- On-chip debug mode register (OCDM)
- Watchdog timer mode register 2 (WDTM2)

After setting the VSWC, OCDM, and WDTM2 registers, set the other registers as necessary.
When using the external bus, set each pin to the alternate-function bus control pin mode by using the port-related registers after setting the above registers.

(a) System wait control register (VSWC)

The VSWC register controls wait of bus access to the on-chip peripheral I/O registers.
Three clocks are required to access an on-chip peripheral I/O register (without a wait cycle). The V850ES/SG1 requires wait cycles according to the operating frequency. Set the following value to the VSWC register in accordance with the frequency used.
The VSWC register can be read or written in 8-bit units (address: FFFFF06EH, default value: 77H).

Operating Frequency (f_{CLK})	Set Value of VSWC	Number of Waits
32 kHz ≤ f_{CLK} < 16.6 MHz	00H	0 (no waits)
16.6 MHz ≤ f_{CLK} ≤ 20 MHz	01H	1

(b) On-chip debug mode register (OCDM)

For details, see **4.6.3 Cautions on on-chip debug pins**.

(c) Watchdog timer mode register 2 (WDTM2)

The WDTM2 register sets the overflow time and the operation clock of watchdog timer 2.
Watchdog timer 2 automatically starts in the reset mode after reset is released. Write the WDTM2 register to activate this operation.
For details, see **CHAPTER 10 FUNCTIONS OF WATCHDOG TIMER 2**.

(2) Accessing specific on-chip peripheral I/O registers

This product has two types of internal system buses.

One is a CPU bus and the other is a peripheral bus that interfaces with low-speed peripheral hardware.

The clock of the CPU bus and the clock of the peripheral bus are asynchronous. If an access to the CPU and an access to the peripheral hardware conflict, therefore, unexpected illegal data may be transferred. If there is a possibility of a conflict, the number of cycles for accessing the CPU changes when the peripheral hardware is accessed, so that correct data is transferred. As a result, the CPU does not start processing of the next instruction but enters the wait status. If this wait status occurs, the number of clocks required to execute an instruction increases by the number of wait clocks shown below.

This must be taken into consideration if real-time processing is required.

When specific on-chip peripheral I/O registers are accessed, more wait states may be required in addition to the wait states set by the VSWC register.

The access conditions and how to calculate the number of wait states to be inserted (number of CPU clocks) at this time are shown below.

(1/2)

Peripheral Function	Register Name	Access	k
16-bit timer/event counter P (TMP) (n = 0 to 5)	TPnCNT	Read	1 or 2
	TPnCCR0, TPnCCR1	Write	• 1st access: No wait • Continuous write: 3 or 4
		Read	1 or 2
Watchdog timer 2 (WDT2)	WDTM2	Write (when WDT2 operating)	3
A/D converter	ADA0M0	Read	1 or 2
	ADA0CR0 to ADA0CR11	Read	1 or 2
	ADA0CR0H to ADA0CR11H	Read	1 or 2
I^2C00, I^2C01	IICS0, IICS1	Read	1
<R> CAN controller[Note 1] (m = 0 to 31, a = 1 to 4)	C0GMABT, C0GMABTD, C0MASKaL, C0MASKaH, C0LEC, C0INFO, C0ERC, C0IE, C0INTS, C0BRP, C0BTR	Read/write	$(f_{XX}/f_{CANMOD} + 1)/(2 + j)$ (MIN.)[Note 2] $(2 \times f_{XX}/f_{CANMOD} + 1)/(2 + j)$ (MAX.)[Note 2]
<R>	C0GMCTRL, C0GMCS, C0CTRL	Read/write	$(f_{XX}/f_{CAN} + 1)/(2 + j)$ (MIN.)[Note 2] $(2 \times f_{XX}/f_{CAN} + 1)/(2 + j)$ (MAX.)[Note 2]
	C0RGPT, C0TGPT	Write	$(f_{XX}/f_{CANMOD} + 1)/(2 + j)$ (MIN.)[Note 2] $(2 \times f_{XX}/f_{CANMOD} + 1)/(2 + j)$ (MAX.)[Note 2]
		Read	$(3 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MIN.)[Note 2] $(4 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MAX.)[Note 2]

(2/2)

	Peripheral Function	Register Name	Access	k
<R>	CAN controller[Note 1] (m = 0 to 31, a = 1 to 4)	C0LIPT, C0LOPT	Read	$(3 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MIN.)[Note 2] $(4 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MAX.)[Note 2]
<R>		C0MCTRLm	Write	$(4 \times f_{XX}/f_{CAN} + 1)/(2 + j)$ (MIN.)[Note 2] $(5 \times f_{XX}/f_{CAN} + 1)/(2 + j)$ (MAX.)[Note 2]
			Read	$(3 \times f_{XX}/f_{CAN} + 1)/(2 + j)$ (MIN.)[Note 2] $(4 \times f_{XX}/f_{CAN} + 1)/(2 + j)$ (MAX.)[Note 2]
<R>		C0MDATA01m, C0MDATA0m, C0MDATA1m, C0MDATA23m, C0MDATA2m, C0MDATA3m, C0MDATA45m, C0MDATA4m, C0MDATA5m, C0MDATA67m, C0MDATA6m, C0MDATA7m, C0MDLCm, C0MCONFm, C0MIDLm, C0MIDHm	Write (8 bits)	$(4 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MIN.)[Note 2] $(5 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MAX.)[Note 2]
			Write (16 bits)	$(2 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MIN.)[Note 2] $(3 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MAX.)[Note 2]
			Read (8/16 bits)	$(3 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MIN.)[Note 2] $(4 \times f_{XX}/f_{CANMODE} + 1)/(2 + j)$ (MAX.)[Note 2]

Number of clocks necessary for access = $3 + i + j + (2 + j) \times k$

<R> **Notes 1.** μPD703253Y only

2. Digits below the decimal point are rounded up.

Caution Accessing the above registers is prohibited in the following statuses. If a wait cycle is generated, it can only be cleared by a reset.

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

Remark f_{XX}: Main clock frequency
f_{CANMOD}: CAN module system clock
f_{CAN}: Supply clock to CAN
i: Values (0 or 1) of higher 4 bits of VSWC register
j: Values (0 or 1) of lower 4 bits of VSWC register

<R> **(3) System reserved area**

No flash memory versions are provided in the V850ES/SG1. However, the μPD70F3261Y, 70F3271Y and 70F3281Y of the V850ES/SG2 can be used as the flash memory versions for developing V850ES/SG1 software.

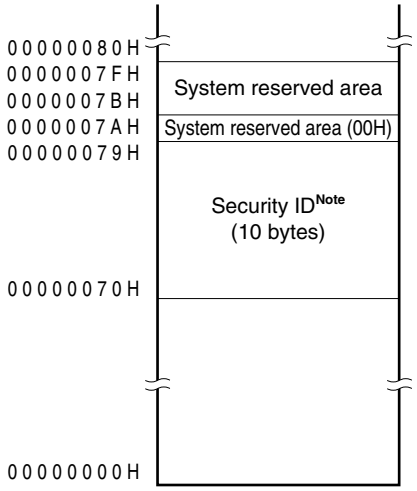
In the μPD70F3261Y, 70F3271Y, and 70F3281Y of the V850ES/SG2, the area of 0000007AH to 0000007FH is a system reserved area for function expansion, and therefore it is recommended that this area not be used. However, in the case of the special version of the following products, be sure to set 00H to 0000007AH.

- μPD70F3261Y, 70F3271Y, 70F3281Y: Ver. 1.0

Remarks 1. With products other than the above, operations are not affected even if 00H is set to 0000007AH.

2. Check the product version by the number that follows DS, ES, or CS on the third line of the package stamp. If the generic name is stamped (V850ES/SG2 etc.), the above restriction does not apply.

For enquiries, contact an NEC Electronics sales representative.



Note For the security ID, see **31.6.1 Security ID** in the **V850ES/SG2, V850ES/SG2-H Hardware User's Manual (U16541E)** .

Caution When the data in the flash memory has been deleted, all the bits are set to 1.

(4) Restriction on conflict between sld instruction and interrupt request

(a) Description

If a conflict occurs between the decode operation of an instruction in <2> immediately before the sld instruction following an instruction in <1> and an interrupt request before the instruction in <1> is complete, the execution result of the instruction in <1> may not be stored in a register.

Instruction <1>

- ld instruction: ld.b, ld.h, ld.w, ld.bu, ld.hu
- sld instruction: sld.b, sld.h, sld.w, sld.bu, sld.hu
- Multiplication instruction: mul, mulh, mulhi, mulu

Instruction <2>

mov reg1, reg2	not reg1, reg2	satsubr reg1, reg2	satsub reg1, reg2
satadd reg1, reg2	satadd imm5, reg2	or reg1, reg2	xor reg1, reg2
and reg1, reg2	tst reg1, reg2	subr reg1, reg2	sub reg1, reg2
add reg1, reg2	add imm5, reg2	cmp reg1, reg2	cmp imm5, reg2
mulh reg1, reg2	shr imm5, reg2	sar imm5, reg2	shl imm5, reg2

<Example>

<i> ld.w [r11], r10
•
•
•
<ii> mov r10, r28
<iii> sld.w 0x28, r10

If the decode operation of the mov instruction <ii> immediately before the sld instruction <iii> and an interrupt request conflict before execution of the ld instruction <i> is complete, the execution result of instruction <i> may not be stored in a register.

(b) Countermeasure

<1> When compiler (CA850) is used
Use CA850 Ver. 2.61 or later because generation of the corresponding instruction sequence can be automatically suppressed.

<2> For assembler
When executing the sld instruction immediately after instruction <ii>, avoid the above operation using either of the following methods.

- Insert a nop instruction immediately before the sld instruction.
- Do not use the same register as the sld instruction destination register in the above instruction <ii> executed immediately before the sld instruction.

CHAPTER 4 PORT FUNCTIONS

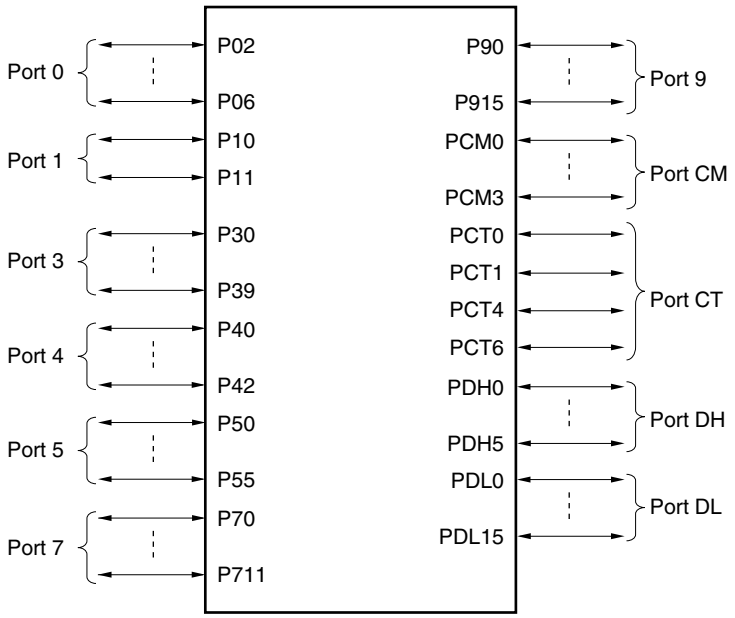
4.1 Features

- I/O ports: 84
  - 5 V tolerant/N-ch open-drain output selectable: 40 (ports 0, 3 to 5, 9)
- Input/output specifiable in 1-bit units

4.2 Basic Port Configuration

The V850ES/SG1 features a total of 84 I/O ports consisting of ports 0, 1, 3 to 5, 7, 9, CM, CT, DH, and DL. The port configuration is shown below.

Figure 4-1. Port Configuration Diagram



Caution Ports 0, 3 to 5, and 9 are 5 V tolerant.

Table 4-1. I/O Buffer Power Supplies for Pins

Power Supply	Corresponding Pins
AV_{REF0}	Port 7
AV_{REF1}	Port 1
BV_{DD}	Ports CM, CT, DH (bits 0 to 3), DL
EV_{DD}	$\overline{RESET}$, ports 0, 3 to 5, 9, DH (bits 4, 5)

4.3 Port Configuration

Table 4-2. Port Configuration

Item	Configuration
Control register	Port n mode register (PMn: n = 0, 1, 3 to 5, 7, 9, CM, CT, DH, DL) Port n mode control register (PMCn: n = 0, 3 to 5, 9, CM, CT, DH, DL) Port n function control register (PFCn: n = 0, 3 to 5, 9) Port n function control expansion register (PFCEn: n = 3, 5, 9) Port n function register (PFn: n = 0, 3 to 5, 9)
Ports	I/O: 84

(1) Port n register (Pn)

Data is input from or output to an external device by writing or reading the Pn register.

The Pn register consists of a port latch that holds output data, and a circuit that reads the status of pins.

Each bit of the Pn register corresponds to one pin of port n, and can be read or written in 1-bit units.

After reset: 00H (output latch) R/W

	7	6	5	7	3	2	1	0
Pn	Pn7	Pn6	Pn5	Pn4	Pn3	Pn2	Pn1	Pn0

Pnm	Control of output data (in output mode)
0	Output 0.
1	Output 1.

Data is written to or read from the Pn register as follows, regardless of the setting of the PMCn register.

Table 4-3. Writing/Reading Pn Register

Setting of PMn Register	Writing to Pn Register	Reading from Pn Register
Output mode (PMnm = 0)	Data is written to the output latch[Note]. In the port mode (PMCn = 0), the contents of the output latch are output from the pins.	The value of the output latch is read.
Input mode (PMnm = 1)	Data is written to the output latch. The pin status is not affected[Note].	The pin status is read.

Note The value written to the output latch is retained until a new value is written to the output latch.

(2) Port n mode register (PMn)

The PMn register specifies the input or output mode of the corresponding port pin.

Each bit of this register corresponds to one pin of port n, and the input or output mode can be specified in 1-bit units.

After reset: FFH R/W

	7	6	5	4	3	2	1	0
PMn	PMn7	PMn6	PMn5	PMn4	PMn3	PMn2	PMn1	PMn0

PMnm	Control of input/output mode
0	Output mode
1	Input mode

(3) Port n mode control register (PMCn)

The PMCn register specifies the port mode or alternate function.

Each bit of this register corresponds to one pin of port n, and the mode of the port can be specified in 1-bit units.

After reset: 00H R/W

	7	6	5	4	3	2	1	0
PMCn	PMCn7	PMCn6	PMCn5	PMCn4	PMCn3	PMCn2	PMCn1	PMCn0

PMCnm	Specification of operation mode
0	Port mode
1	Alternate function mode

(4) Port n function control register (PFCn)

The PFCn register specifies the alternate function of a port pin to be used if the pin has two alternate functions.

Each bit of this register corresponds to one pin of port n, and the alternate function of a port pin can be specified in 1-bit units.

After reset: 00H R/W

	7	6	5	4	3	2	1	0
PFCn	PFCn7	PFCn6	PFCn5	PFCn4	PFCn3	PFCn2	PFCn1	PFCn0

PFCnm	Specification of alternate function
0	Alternate function 1
1	Alternate function 2

(5) Port n function control expansion register (PFCEn)

The PFCEn register specifies the alternate function of a port pin to be used if the pin has three or more alternate functions.

Each bit of this register corresponds to one pin of port n, and the alternate function of a port pin can be specified in 1-bit units.

After reset: 00H R/W

	7	6	5	4	3	2	1	0
PFCEn	PFCEn7	PFCEn6	PFCEn5	PFCEn4	PFCEn3	PFCEn2	PFCEn1	PFCEn0
PFCn	PFCn7	PFCn6	PFCn5	PFCn4	PFCn3	PFCn2	PFCn1	PFCn0

PFCEnm	PFCnm	Specification of alternate function
0	0	Alternate function 1
0	1	Alternate function 2
1	0	Alternate function 3
1	1	Alternate function 4

(6) Port n function register (PFn)

The PFn register specifies normal output or N-ch open-drain output.

Each bit of this register corresponds to one pin of port n, and the output mode of the port pin can be specified in 1-bit units.

After reset: 00H R/W

PFn	7	6	5	4	3	2	1	0
	PFn7	PFn6	PFn5	PFn4	PFn3	PFn2	PFn1	PFn0

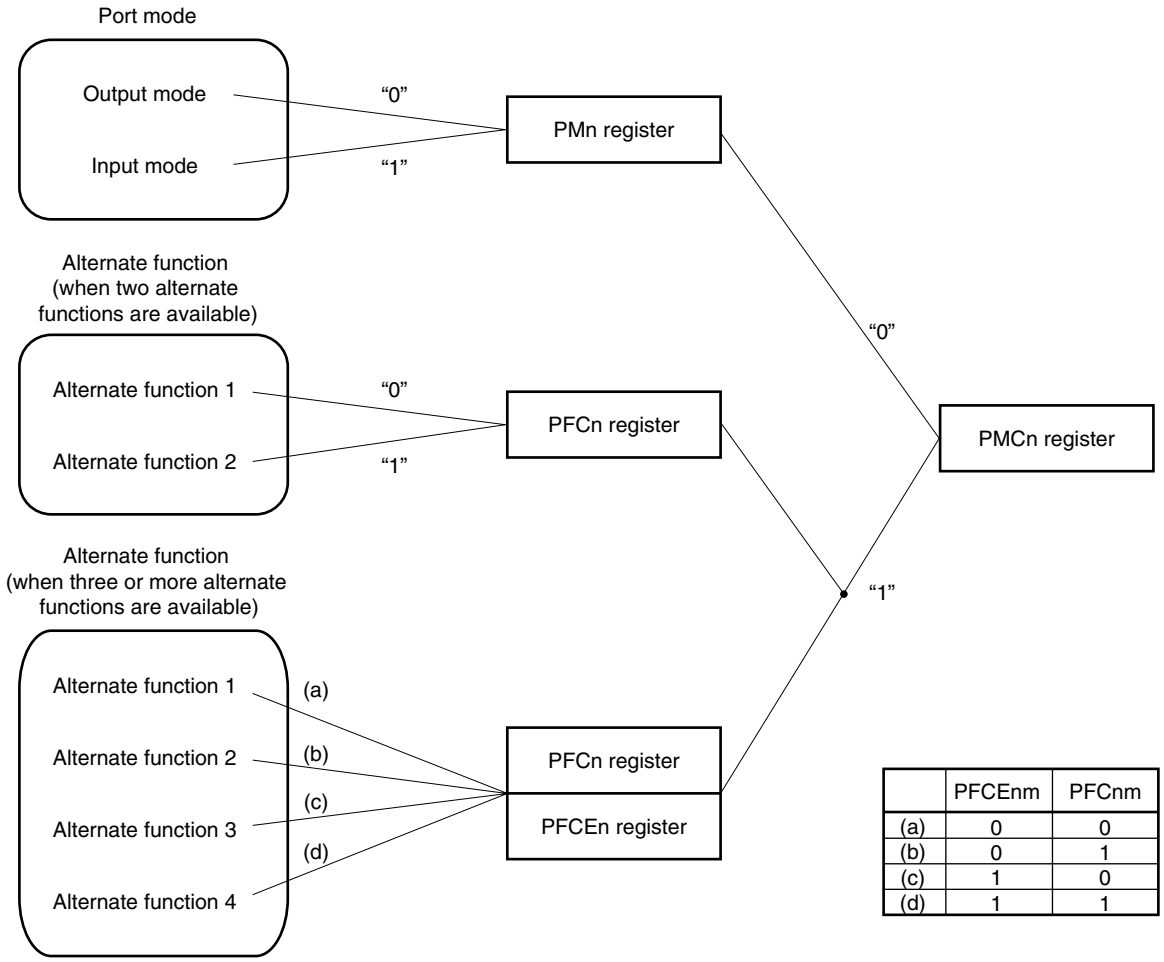
PFnm[Note]	Control of normal output/N-ch open-drain output
0	Normal output (CMOS output)
1	N-ch open-drain output

Note The PFnm bit of the PFn register is valid only when the PMnm bit of the PMn register is 0 (when the output mode is specified) in port mode (PMCnm bit = 0). When the PMnm bit is 1 (when the input mode is specified), the set value of the PFn register is invalid.

(7) Port setting

Set a port as illustrated below.

Figure 4-2. Setting of Each Register and Pin Function



	PFCEnm	PFCnm
(a)	0	0
(b)	0	1
(c)	1	0
(d)	1	1

Remark Set the alternate functions in the following sequence.

<1> Set the PFCn and PFCEn registers.
<2> Set the PFCn register.
<3> Set the INTRn or INTFn register (to specify an external interrupt pin).

If the PMCn register is set first, an unintended function may be set while the PFCn and PFCEn registers are being set.

4.3.1 Port 0

Port 0 is a 5-bit port for which I/O settings can be controlled in 1-bit units.

Port 0 includes the following alternate-function pins.

Table 4-4. Port 0 Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
P02	19	17	NMI	Input	Selectable as N-ch open-drain output	L-1
P03	20	18	INTP0/ADTRG	Input		N-1
P04	21	19	INTP1	Input		L-1
P05Note	22	20	INTP2	Input		AA-1
P06	23	21	INTP3	Input		L-1

<R> **Note** This pin functions alternately as the on-chip debug function pin ($\overline{\text{DRST}}$) in the flash memory version (V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)).

If the flash memory is used but on-chip debugging is not used, fix the P05/INTP2 pin to low level between when the reset signal of the $\overline{\text{RESET}}$ pin is released and when the OCDM.OCDM0 bit is cleared (0).

For details, see **4.6.3 Cautions on on-chip debug pins**.

Caution The P02 to P06 pins have hysteresis characteristics in the input mode of the alternate function, but do not have hysteresis characteristics in the port mode.

Remark GF: 100-pin plastic QFP (14 × 20)

GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port 0 register (P0)

After reset: 00H (output latch) R/W Address: FFFFF400H

	7	6	5	4	3	2	1	0
P0	0	P06	P05	P04	P03	P02	0	0

P0n	Output data control (in output mode) (n = 2 to 6)
0	Outputs 0
1	Outputs 1

(2) Port 0 mode register (PM0)

After reset: FFH R/W Address: FFFFF420H

	7	6	5	4	3	2	1	0
PM0	1	PM06	PM05	PM04	PM03	PM02	1	1

PM0n	I/O mode control (n = 2 to 6)
0	Output mode
1	Input mode

(3) Port 0 mode control register (PMC0)

After reset: 00H R/W Address: FFFFF440H

	7	6	5	4	3	2	1	0
PMC0	0	PMC06	PMC05	PMC04	PMC03	PMC02	0	0

PMC06	Specification of P06 pin operation mode
0	I/O port
1	INTP3 input

PMC05	Specification of P05 pin operation mode
0	I/O port
1	INTP2 input

PMC04	Specification of P04 pin operation mode
0	I/O port
1	INTP1 input

PMC03	Specification of P03 pin operation mode
0	I/O port
1	INTP0 input/ADTRG input

PMC02	Specification of P02 pin operation mode
0	I/O port
1	NMI input

(4) Port 0 function control register (PFC0)

After reset: 00H R/W Address: FFFFF460H

	7	6	5	4	3	2	1	0
PFC0	0	0	0	0	PFC03	0	0	0

PFC03	Specification of P03 pin alternate function
0	INTP0 input
1	ADTRG input

(5) Port 0 function register (PF0)

After reset: 00H R/W Address: FFFFFC60H

	7	6	5	4	3	2	1	0
PF0	0	PF06	PF05	PF04	PF03	PF02	0	0

PF0n	Control of normal output or N-ch open-drain output (n = 2 to 6)
0	Normal output (CMOS output)
1	N-ch open drain output

<R> **Caution When an output pin is pulled up at EV_{DD} or higher, be sure to set the corresponding PF0n bit to 1.**

4.3.2 Port 1

Port 1 is a 2-bit port for which I/O settings can be controlled in 1-bit units.

Port 1 includes the following alternate-function pins.

Table 4-5. Port 1 Alternate-Function Pins

	Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
		GF	GC				
	P10	5	3	–	–	–	B-1
<R>		–	3	ANO0[Note]	Output		A-2
	P11	6	4	–	–		B-1
<R>		–	4	ANO1[Note]	Output		A-2

<R> **Note** μPD703253Y only

<R> **Caution The P10 and P11 pins may momentarily output an undefined level upon power application.**

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port 1 register (P1)

After reset: 00H (output latch) R/W Address: FFFFF402H

	7	6	5	4	3	2	1	0
P1	0	0	0	0	0	0	P11	P10

P1n	Output data control (in output mode) (n = 0, 1)
0	Outputs 0
1	Outputs 1

<R> **Caution In the μPD703253Y, do not perform read from or write to the P1 register during D/A conversion (refer to 12.4.3 Cautions).**

(2) Port 1 mode register (PM1)

After reset: FFH R/W Address: FFFFF422H

	7	6	5	4	3	2	1	0
PM1	1	1	1	1	1	1	PM11	PM10

PM1n	I/O mode control (n = 0, 1)
0	Output mode
1	Input mode

<R> **Cautions 1. When using P1n as the alternate function (ANOn pin output), set the PM1n bit to 1.**

<R> **2. In the μPD703253Y, when using one of the P10 and P11 pins as an I/O port and the other as a D/A output pin, do so in an application where the port I/O level does not change during D/A output.**

4.3.3 Port 3

Port 3 is a 10-bit port for which I/O settings can be controlled in 1-bit units.

Port 3 includes the following alternate-function pins.

Table 4-6. Port 3 Alternate-Function Pins

	Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
		GF	GC				
	P30	27	25	TXDA0	Output	Selectable as N-ch open-drain output	G-7
	P31	28	26	RXDA0/INTP7	Input		N-4
	P32	29	27	ASCKA0/TIP00/TOP00	I/O		U-16
<R>	P33	30	28	TIP01/TOP01	I/O		G-1
<R>	P34	31	29	TIP10/TOP10	I/O		G-1
<R>	P35	32	30	TIP11/TOP11	I/O		G-1
	P36	33	31	$\overline{\text{IETX0}}$ Note 1	Output		G-8
<R>	P36	–	31	CTXD0 Note 2	Output		G-7
	P37	34	32	$\overline{\text{IERX0}}$ Note 1	Input		G-9
<R>	P37	–	32	CRXD0 Note 2	Input		G-14
	P38	37	35	SDA00	I/O		G-10
	P39	38	36	SCL00	I/O		G-11

Notes 1. μPD703252Y only

<R> **2.** μPD703253Y only

<R> **Caution The P31 to P35 and P37 to P39 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.**

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port 3 register (P3)

After reset: 0000H (output latch) R/W Address: P3 FFFFF406H, P3L FFFFF406H, P3H FFFFF407H

	15	14	13	12	11	10	9	8
P3 (P3H)	0	0	0	0	0	0	P39	P38
	7	6	5	4	3	2	1	0
(P3L)	P37	P36	P35	P34	P33	P32	P31	P30

P3n	Output data control (in output mode) (n = 0 to 9)
0	Outputs 0
1	Outputs 1

Remarks 1. The P3 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the P3 register as the P3H register and the lower 8 bits as the P3L register, P3 can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the P3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the P3H register.

(2) Port 3 mode register (PM3)

After reset: FFFFH R/W Address: PM3 FFFFF426H, PM3L FFFFF426H, PM3H FFFFF427H

	15	14	13	12	11	10	9	8
PM3 (PM3H)	1	1	1	1	1	1	PM39	PM38

	7	6	5	4	3	2	1	0
(PM3L)	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30

PM3n	I/O mode control (n = 0 to 9)
0	Output mode
1	Input mode

Remarks 1. The PM3 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PM3 register as the PM3H register and the lower 8 bits as the PM3L register, PM3 can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the PM3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PM3H register.

(3) Port 3 mode control register (PMC3)

After reset: 0000H R/W Address: PMC3 FFFFF446H, PMC3L FFFFF446H, PMC3H FFFFF447H

	15	14	13	12	11	10	9	8
PMC3 (PMC3H)	0	0	0	0	0	0	PMC39	PMC38

	7	6	5	4	3	2	1	0
(PMC3L)	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30

PMC39	Specification of P39 pin operation mode
0	I/O port
1	SCL00 I/O

PMC38	Specification of P38 pin operation mode
0	I/O port
1	SDA00 I/O

(2/2)

PMC37	Specification of P37 pin operation mode
0	I/O port
1	$\overline{\text{IERX0}}$ input[Note 1]/CRXD0 input[Note 2]

PMC36	Specification of P36 pin operation mode
0	I/O port
1	$\overline{\text{IETX0}}$ output[Note 1]/CTXD 0 output[Note 2]

PMC35	Specification of P35 pin operation mode
0	I/O port
1	TIP11 input/TOP11 output

PMC34	Specification of P34 pin operation mode
0	I/O port
1	TIP10 input/TOP10 output

PMC33	Specification of P33 pin operation mode
0	I/O port
1	TIP01 input/TOP01 output

PMC32	Specification of P32 pin operation mode
0	I/O port
1	ASCKA0 input/TIP00 input/TOP00 output

PMC31	Specification of P31 pin operation mode
0	I/O port
1	RXDA0 input/INTP7 input

PMC30	Specification of P30 pin operation mode
0	I/O port
1	TXDA0 output

Notes 1. μPD703252Y only

2. μPD703253Y only

Remarks 1. The PMC3 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PMC3 register as the PMC3H register and the lower 8 bits as the PMC3L register, PMC3 can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the PMC3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMC3H register.

(4) Port 3 function control register (PFC3)

After reset: 0000H R/W Address: PFC3 FFFFF466H, PFC3L FFFFF466H, PFC3L FFFFF467H

	15	14	13	12	11	10	9	8
PFC3 (PFC3H)	0	0	0	0	0	0	PFC39	PFC38
	7	6	5	4	3	2	1	0
(PFC3L)	PFC37	PFC36	PFC35	PFC34	PFC33	PFC32	PFC31	PFC30

Remarks 1. For details of alternate function specification, see **4.3.3 (6) Port 3 alternate function specifications**.

2. The PFC3 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PFC3 register as the PFC3H register and the lower 8 bits as the PFC3L register, PFC3 can be read or written in 8-bit and 1-bit units.

3. To read/write bits 8 to 15 of the PFC3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PFC3H register.

(5) Port 3 function control expansion register L (PFCE3L)

After reset: 00H R/W Address: FFFFF706H

	7	6	5	4	3	2	1	0
PFCE3L	0	0	0	0	0	PFCE32	0	0

Remark For details of alternate function specification, see **4.3.3 (6) Port 3 alternate function specifications**.

(6) Port 3 alternate function specifications

PFC39	Specification of P39 pin alternate function
0	Setting prohibited
1	SCL00 I/O

PFC38	Specification of P38 pin alternate function
0	Setting prohibited
1	SDA00 I/O

PFC37	Specification of P37 pin alternate function
0	CRXD0 input[Note 1]
1	$\overline{\text{IERX0}}$ input[Note 2]

PFC36	Specification of P36 pin alternate function
0	CTXD0 output[Note 1]
1	$\overline{\text{IETX0}}$ output[Note 2]

PFC35	Specification of P35 pin alternate function
0	TIP11 input
1	TOP11 output

PFC34	Specification of P34 pin alternate function
0	TIP10 input
1	TOP10 output

PFC33	Specification of P33 pin alternate function
0	TIP01 input
1	TOP01 output

PFCE32	PFC32	Specification of P32 pin alternate function
0	0	ASCKA0 input
0	1	Setting prohibited
1	0	TIP00 input
1	1	TOP00 output

PFC31	Specification of P31 pin alternate function
0	RXDA0 input/INTP7[Note 3] input
1	Setting prohibited

PFC30	Specification of P30 pin alternate function
0	TXDA0 output
1	Setting prohibited

Notes 1. μPD703253Y only

2. μPD703252Y only

3. The INTP7 pin and RXDA0 pin are alternate-function pins. When using the pin as the RXDA0 pin, disable edge detection for the INTP7 alternate-function pin. (Clear the INTF3.INTF31 bit and the INTR3.INTR31 bit to 0.) When using the pin as the INTP7 pin, stop UARTA0 reception. (Clear the UA0CTL0.UA0RXE bit to 0.)

(7) Port 3 function register (PF3)

After reset: 0000H R/W Address: PF3 FFFFFC66H,
PF3L FFFFFC66H, PF3H FFFFFC67H

	15	14	13	12	11	10	9	8
PF3 (PF3H)	0	0	0	0	0	0	PF39	PF38

	7	6	5	4	3	2	1	0
(PF3L)	PF37	PF36	PF35	PF34	PF33	PF32	PF31	PF30

PF3n	Control of normal output or N-ch open-drain output (n = 0 to 9)
0	Normal output (CMOS output)
1	N-ch open-drain output

<R> **Caution When an output pin is pulled up at EVDD or higher, be sure to set the corresponding PF3n bit to 1.**

Remarks 1. The PF3 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PF3 register as the PF3H register and the lower 8 bits as the PF3L register, PF3 can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the PF3 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PF3H register.

4.3.4 Port 4

Port 4 is a 3-bit port that controls I/O in 1-bit units.

Port 4 includes the following alternate-function pins.

Table 4-7. Port 4 Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
P40	24	22	SIB0/SDA01	I/O	Selectable as N-ch open-drain output	G-6
<R> P41	25	23	SOB0/SCL01	I/O		G-12
P42	26	24	$\overline{\text{SCKB0}}$	I/O		E-3

Caution The P40 to P42 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port 4 register (P4)

After reset: 00H (output latch) R/W Address: FFFFF408H

	7	6	5	4	3	2	1	0
P4	0	0	0	0	0	P42	P41	P40

P4n	Output data control (in output mode) (n = 0 to 2)
0	Outputs 0
1	Outputs 1

(2) Port 4 mode register (PM4)

After reset: FFH R/W Address: FFFFF428H

	7	6	5	4	3	2	1	0
PM4	1	1	1	1	1	PM42	PM41	PM40

PM4n	I/O mode control (n = 0 to 2)
0	Output mode
1	Input mode

(3) Port 4 mode control register (PMC4)

After reset: 00H R/W Address: FFFFF448H

	7	6	5	4	3	2	1	0
PMC4	0	0	0	0	0	PMC42	PMC41	PMC40

PMC42	Specification of P42 pin operation mode
0	I/O port
1	$\overline{\text{SCKB0}}$ I/O

PMC41	Specification of P41 pin operation mode
0	I/O port
1	SOB0 output/SCL01 I/O

PMC40	Specification of P40 pin operation mode
0	I/O port
1	SIB0 input/SDA01 I/O

(4) Port 4 function control register (PFC4)

After reset: 00H R/W Address: FFFFF468H

	7	6	5	4	3	2	1	0
PFC4	0	0	0	0	0	0	PFC41	PFC40

PFC41	Specification of P41 pin alternate function
0	SOB0 output
1	SCL01 I/O

PFC40	Specification of P40 pin alternate function
0	SIB0 input
1	SDA01 I/O

(5) Port 4 function register (PF4)

After reset: 00H R/W Address: FFFFFC68H

	7	6	5	4	3	2	1	0
PF4	0	0	0	0	0	PF42	PF41	PF40

PF4n	Control of normal output or N-ch open-drain output (n = 0 to 2)
0	Normal output (CMOS output)
1	N-ch open-drain output

<R> **Caution When an output pin is pulled up at EV_{DD} or higher, be sure to set the corresponding PF4n bit to 1.**

4.3.5 Port 5

Port 5 is a 6-bit port that controls I/O in 1-bit units.

Port 5 includes the following alternate-function pins.

Table 4-8. Port 5 Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
P50	39	37	KR0	Input	Selectable as N-ch open-drain output	U-20
P51	40	38	KR1	Input		U-20
P52Note	41	39	KR2	Input		U-21
P53Note	42	40	KR3	Input		U-22
P54Note	43	41	KR4	Input		U-23
P55Note	44	42	KR5	Input		U-24

<R> **Note** This pin functions alternately as the on-chip debug function pin (DDI, DDO, DCK, or DMS) in the flash memory version (V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)).

If the flash memory is used but on-chip debugging is not used, fix the P05/INTP2 pin to low level between when the reset signal of the $\overline{\text{RESET}}$ pin is released and when the OCDM.OCDM0 bit is cleared (0). Although the mask ROM versions do not support the on-chip debug mode, handle the P05/INTP2 pin the same as the flash memory versions.

For details, see **4.6.3 Cautions on on-chip debug pins**.

<R> **Cautions 1. The P53 pin may momentarily output an undefined level upon power application.**

2. The P50 to P55 pins have hysteresis characteristics in the input mode of the alternate function, but do not have hysteresis characteristics in the port mode.

Remark GF: 100-pin plastic QFP (14 × 20)

GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port 5 register (P5)

After reset: 00H (output latch) R/W Address: FFFFF40AH

	7	6	5	4	3	2	1	0
P5	0	0	P55	P54	P53	P52	P51	P50

P5n	Output data control (in output mode) (n = 0 to 5)
0	Outputs 0
1	Outputs 1

(2) Port 5 mode register (PM5)

After reset: FFH R/W Address: FFFFF42AH

	7	6	5	4	3	2	1	0
PM5	1	1	PM55	PM54	PM53	PM52	PM51	PM50

PM5n	I/O mode control (n = 0 to 5)
0	Output mode
1	Input mode

(3) Port 5 mode control register (PMC5)

After reset: 00H R/W Address: FFFFF44AH

	7	6	5	4	3	2	1	0
PMC5	0	0	PMC55	PMC54	PMC53	PMC52	PMC51	PMC50

PMC55	Specification of P55 pin operation mode
0	I/O port
1	KR5 input

PMC54	Specification of P54 pin operation mode
0	I/O port
1	KR4 input

PMC53	Specification of P53 pin operation mode
0	I/O port
1	KR3 input

PMC52	Specification of P52 pin operation mode
0	I/O port
1	KR2 input

PMC51	Specification of P51 pin operation mode
0	I/O port
1	KR1 input

PMC50	Specification of P50 pin operation mode
0	I/O port
1	KR0 input

(4) Port 5 function control register (PFC5)

After reset: 00H R/W Address: FFFFF46AH

	7	6	5	4	3	2	1	0
PFC5	0	0	PFC55	PFC54	PFC53	PFC52	PFC51	PFC50

Remark For details of alternate function specification, see **4.3.5 (6) Port 5 alternate function specifications**.

(5) Port 5 function control expansion register (PFCE5)

After reset: 00H R/W Address: FFFFF70AH

	7	6	5	4	3	2	1	0
PFCE5	0	0	PFCE55	PFCE54	PFCE53	PFCE52	PFCE51	PFCE50

Remark For details of alternate function specification, see **4.3.5 (6) Port 5 alternate function specifications**.

(6) Port 5 alternate function specifications

PFCE55	PFC55	Specification of P55 pin alternate function
0	1	KR5 input
Other than above		Setting prohibited

PFCE54	PFC54	Specification of P54 pin alternate function
0	1	KR4 input
Other than above		Setting prohibited

PFCE53	PFC53	Specification of P53 pin alternate function
0	1	KR3 input
Other than above		Setting prohibited

PFCE52	PFC52	Specification of P52 pin alternate function
0	1	KR2 input
Other than above		Setting prohibited

PFCE51	PFC51	Specification of P51 pin alternate function
0	1	KR1 input
Other than above		Setting prohibited

PFCE50	PFC50	Specification of P50 pin alternate function
0	1	KR0 input
Other than above		Setting prohibited

(7) Port 5 function register (PF5)

After reset: 00H R/W Address: FFFFFC6AH

	7	6	5	4	3	2	1	0
PF5	0	0	PF55	PF54	PF53	PF52	PF51	PF50

PF5n	Control of normal output or N-ch open-drain output (n = 0 to 5)
0	Normal output (CMOS output)
1	N-ch open-drain output

<R> **Caution When an output pin is pulled up at EV_{DD} or higher, be sure to set the corresponding PF5n bit to 1.**

4.3.6 Port 7

Port 7 is a 12-bit port for which I/O settings can be controlled in 1-bit units.

Port 7 includes the following alternate-function pins.

Table 4-9. Port 7 Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
P70	2	100	ANI0	Input	–	A-1
P71	1	99	ANI1	Input		A-1
P72	100	98	ANI2	Input		A-1
P73	99	97	ANI3	Input		A-1
P74	98	96	ANI4	Input		A-1
P77	97	95	ANI5	Input		A-1
P76	96	94	ANI6	Input		A-1
P77	95	93	ANI7	Input		A-1
P78	94	92	ANI8	Input		A-1
P79	93	91	ANI9	Input		A-1
P710	92	90	ANI10	Input		A-1
P711	91	89	ANI11	Input		A-1

Remark GF: 100-pin plastic QFP (14 × 20)

GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port 7 register H, port 7 register L (P7H, P7L)

<R>

After reset: 00H (output latch) R/W Address: P7L FFFFF40EH, P7H FFFFF40FH

	7	6	5	4	3	2	1	0
P7H	0	0	0	0	P711	P710	P79	P78

	7	6	5	4	3	2	1	0
P7L	P77	P76	P75	P74	P73	P72	P71	P70

P7n	Output data control (in output mode) (n = 0 to 11)
0	Outputs 0
1	Outputs 1

<R>

Caution Do not perform read or write to the P7H and P7L registers during A/D conversion (refer to 11.6 (4) Alternate I/O).

Remark These registers cannot be accessed in 16-bit units as the P7 register. They can be read or written in 8-bit or 1-bit units as the P7H and P7L registers.

(2) Port 7 mode register H, port 7 mode register L (PM7H, PM7L)

After reset: FFH R/W Address: PM7L FFFFF42EH, PM7H FFFFF42FH

	7	6	5	4	3	2	1	0
PM7H	1	1	1	1	PM711	PM710	PM79	PM78

	7	6	5	4	3	2	1	0
PM7L	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70

PM7n	I/O mode control (n = 0 to 11)
0	Output mode
1	Input mode

Caution When using the P7n pin as its alternate function (ANIn pin), set the PM7n bit to 1.

Remark These registers cannot be accessed in 16-bit units as the PM7 register. They can be read or written in 8-bit or 1-bit units as the PM7H and PM7L registers.

4.3.7 Port 9

Port 9 is a 16-bit port for which I/O settings can be controlled in 1-bit units.

Port 9 includes the following alternate-function pins.

Table 4-10. Port 9 Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
P90	45	43	A0/KR6/TXDA1	I/O	Selectable as N-ch open-drain output	U-25
P91	46	44	A1/KR7/RXDA1	I/O		U-26
P92	47	45	A2/TIP41/TOP41	I/O		U-12
P93	48	46	A3/TIP40/TOP40	I/O		U-12
P94	49	47	A4/TIP31/TOP31	I/O		U-12
P95	50	48	A5/TIP30/TOP30	I/O		U-12
P96	51	49	A6/TIP21/TOP21	I/O		U-13
P97	52	50	A7/SIB1/TIP20/TOP20	I/O		U-14
P98	53	51	A8/SOB1	Output		G-3
P99	54	52	A9/$\overline{\text{SCKB1}}$	I/O		G-5
P910	55	53	A10/SIB3	I/O		G-2
P911	56	54	A11/SOB3	Output		G-3
P912	57	55	A12/$\overline{\text{SCKB3}}$	I/O		G-5
P913	58	56	A13/INTP4	I/O		N-2
P914	59	57	A14/INTP5	I/O		U-27
<R>	–	57	A14/INTP5/TIP51[Note]/TOP51[Note]	I/O		U-15
P915	60	58	A15/INTP6	I/O		U-27
<R>	–	58	A15/INTP6/TIP50[Note]/TOP50[Note]	I/O		U-15

<R> **Note** μPD703253Y only

<R> **Caution The P90 to P97, P99, P910, and P912 to P915 pins have hysteresis characteristics in the input mode of the alternate-function pin, but do not have the hysteresis characteristics in the port mode.**

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port 9 register (P9)

After reset: 0000H (output latch) R/W Address: P9 FFFFF412H,
P9L FFFFF412H, P9H FFFFF413H

	15	14	13	12	11	10	9	8
P9 (P9H)	P915	P914	P913	P912	P911	P910	P99	P98

	7	6	5	4	3	2	1	0
(P9L)	P97	P96	P95	P94	P93	P92	P91	P90

P9n	Output data control (in output mode) (n = 0 to 15)
0	Outputs 0
1	Outputs 1

Remarks 1. The P9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the P9 register as the P9H register and the lower 8 bits as the P9L register, P9 can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the P9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the P9H register.

(2) Port 9 mode register (PM9)

After reset: FFFFH R/W Address: PM9 FFFFF432H,
PM9L FFFFF432H, PM9H FFFFF433H

	15	14	13	12	11	10	9	8
PM9 (PM9H)	PM915	PM914	PM913	PM912	PM911	PM910	PM99	PM98

	7	6	5	4	3	2	1	0
(PM9L)	PM97	PM96	PM95	PM94	PM93	PM92	PM91	PM90

PM9n	I/O mode control (n = 0 to 15)
0	Output mode
1	Input mode

Remarks 1. The PM9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PM9 register as the PM9H register and the lower 8 bits as the PM9L register, PM9 can be read or written in 8-bit and 1-bit units.

2. To read/write bits 8 to 15 of the PM9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PM9H register.

(3) Port 9 mode control register (PMC9)

After reset: 0000H R/W Address: PMC9 FFFFF452H,
PMC9L FFFFF452H, PMC9H FFFFF453H

	15	14	13	12	11	10	9	8
PMC9 (PMC9H)	PMC915	PMC914	PMC913	PMC912	PMC911	PMC910	PMC99	PMC98

	7	6	5	4	3	2	1	0
(PMC9L)	PMC97	PMC96	PMC95	PMC94	PMC93	PMC92	PMC91	PMC90

PMC915	Specification of P915 pin operation mode
0	I/O port
1	A15 output/INTP6 input/TIP50 input[Note]/TOP50 output[Note]

<R>

PMC914	Specification of P914 pin operation mode
0	I/O port
1	A14 output/INTP5 input/TIP51 input[Note]/TOP51 output[Note]

<R>

PMC913	Specification of P913 pin operation mode
0	I/O port
1	A13 output/INTP4 input

PMC912	Specification of P912 pin operation mode
0	I/O port
1	A12 output/$\overline{\text{SCKB3}}$ I/O

PMC911	Specification of P911 pin operation mode
0	I/O port
1	A11 output/SOB3 output

PMC910	Specification of P910 pin operation mode
0	I/O port
1	A10 output/SIB3 input

PMC99	Specification of P99 pin operation mode
0	I/O port
1	A9 output/$\overline{\text{SCKB1}}$ I/O

<R> **Note** μPD703253Y only

Remarks 1. The PMC9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PMC9 register as the PMC9H register and the lower 8 bits as the PMC9L register, PMC9 can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the PMC9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMC9H register.

(2/2)

PMC98	Specification of P98 pin operation mode
0	I/O port
1	A8 output/SOB1 output

PMC97	Specification of P97 pin operation mode
0	I/O port
1	A7 output/SIB1 input/TIP20 input/TOP20 output

PMC96	Specification of P96 pin operation mode
0	I/O port
1	A6 output/TIP21 input/TOP21 output

PMC95	Specification of P95 pin operation mode
0	I/O port
1	A5 output/TIP30 input/TOP30 output

PMC94	Specification of P94 pin operation mode
0	I/O port
1	A4 output/TIP31 input/TOP31 output

PMC93	Specification of P93 pin operation mode
0	I/O port
1	A3 output/TIP40 input/TOP40 output

PMC92	Specification of P92 pin operation mode
0	I/O port
1	A2 output/TIP41 input/TOP41 output

PMC91	Specification of P91 pin operation mode
0	I/O port
1	A1 output/KR7 input/RXDA1 input

PMC90	Specification of P90 pin operation mode
0	I/O port
1	A0 output/KR6 input/TXDA1 output

<R> **Caution Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once. If even one of the A0 to A15 pins is not used in the separate bus mode, port 9 pins can be used as port pins or other alternate-function pins.**

(4) Port 9 function control register (PFC9)

<R> **Caution Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once. If even one of the A0 to A15 pins is not used in the separate bus mode, port 9 pins can be used as port pins or other alternate-function pins.**

After reset: 0000H R/W Address: PFC9 FFFFF472H, PFC9L FFFFF472H, PFC9H FFFFF473H

	15	14	13	12	11	10	9	8
PFC9 (PFC9H)	PFC915	PFC914	PFC913	PFC912	PFC911	PFC910	PFC99	PFC98

	7	6	5	4	3	2	1	0
(PFC9L)	PFC97	PFC96	PFC95	PFC94	PFC93	PFC92	PFC91	PFC90

Remarks 1. For details of alternate function specification, see **4.3.7 (6) Port 9 alternate function specifications**.

2. The PFC9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PFC9 register as the PFC9H register and the lower 8 bits as the PFC9L register, PFC9 can be read or written in 8-bit or 1-bit units.

3. To read/write bits 8 to 15 of the PFC9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PFC9H register.

(5) Port 9 function control expansion register (PFCE9)

After reset: 0000H R/W Address: PFCE9 FFFFF712H, PFCE9L FFFFF712H, PFCE9H FFFFF713H

	15	14	13	12	11	10	9	8
PFCE9 (PFCE9H)	PFCE915	PFCE914	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
(PFCE9L)	PFCE97	PFCE96	PFCE95	PFCE94	PFCE93	PFCE92	PFCE91	PFCE90

Remarks 1. For details of alternate function specification, see **4.3.7 (6) Port 9 alternate function specifications**.

2. The PFCE9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PFCE9 register as the PFCE9H register and the lower 8 bits as the PFCE9L register, PFCE9 can be read or written in 8-bit or 1-bit units.

3. To read/write bits 8 to 15 of the PFCE9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PFCE9H register.

(6) Port 9 alternate function specifications

PFCE915	PFC915	Specification of P915 pin alternate function
0	0	A15 output
0	1	INTP6 input
<R> 1	0	TIP50 input[Note]
<R> 1	1	TOP50 output[Note]

PFCE914	PFC914	Specification of P914 pin alternate function
0	0	A14 output
0	1	INTP5 input
<R> 1	0	TIP51 input[Note]
<R> 1	1	TOP51 output[Note]

PFC913	Specification of P913 pin alternate function
0	A13 output
1	INTP4 input

PFC912	Specification of P912 pin alternate function
0	A12 output
1	$\overline{\text{SCKB3}}$ I/O

PFC911	Specification of P911 pin alternate function
0	A11 output
1	SOB3 output

PFC910	Specification of P910 pin alternate function
0	A10 output
1	SIB3 input

PFC99	Specification of P99 pin alternate function
0	A9 output
1	$\overline{\text{SCKB1}}$ I/O

PFC98	Specification of P98 pin alternate function
0	A8 output
1	SOB1 output

PFCE97	PFC97	Specification of P97 pin alternate function
0	0	A7 output
0	1	SIB1 input
1	0	TIP20 input
1	1	TOP20 output

<R> **Note** μPD703253Y only

PFCE96	PFC96	Specification of P96 pin alternate function
0	0	A6 output
0	1	Setting prohibited
1	0	TIP21 input
1	1	TOP21 output

PFCE95	PFC95	Specification of P95 pin alternate function
0	0	A5 output
0	1	TIP30 input
1	0	TOP30 output
1	1	Setting prohibited

PFCE94	PFC94	Specification of P94 pin alternate function
0	0	A4 output
0	1	TIP31 input
1	0	TOP31 output
1	1	Setting prohibited

PFCE93	PFC93	Specification of P93 pin alternate function
0	0	A3 output
0	1	TIP40 input
1	0	TOP40 output
1	1	Setting prohibited

PFCE92	PFC92	Specification of P92 pin alternate function
0	0	A2 output
0	1	TIP41 input
1	0	TOP41 output
1	1	Setting prohibited

PFCE91	PFC91	Specification of P91 pin alternate function
0	0	A1 output
0	1	KR7 input
1	0	RXDA1 input/KR7 input[Note]
1	1	Setting prohibited

PFCE90	PFC90	Specification of P90 pin alternate function
0	0	A0 output
0	1	KR6 input
1	0	TXDA1 output
1	1	Setting prohibited

Note The RXDA1 and KR7 pins must not be used at the same time. When using the RXDA1 pin, do not use the KR7 pin. When using the KR7 pin, do not use the RXDA1 pin (it is recommended to set the PFC91 bit to 1 and clear the PFCE91 bit to 0).

(7) Port 9 function register (PF9)

After reset: 0000H R/W Address: PF3 FFFFFC72H,
PF9L FFFFFC72H, PF9H FFFFFC73H

	15	14	13	12	11	10	9	8
PF9 (PF9H)	PF915	PF914	PF913	PF912	PF911	PF910	PF99	PF98

	7	6	5	4	3	2	1	0
(PF9L)	PF97	PF96	PF95	PF94	PF93	PF92	PF91	PF90

PF9n	Control of normal output or N-ch open-drain output (n = 0 to 15)
0	Normal output (CMOS output)
1	N-ch open-drain output

<R> **Caution When an output pin is pulled up at EVDD or higher, be sure to set the corresponding PF9n bit to 1.**

Remarks 1. The PF9 register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PF9 register as the PF9H register and the lower 8 bits as the PF9L register, PF9 can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the PF9 register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PF9H register.

4.3.8 Port CM

Port CM is a 4-bit port for which I/O settings can be controlled in 1-bit units.

Port CM includes the following alternate-function pins.

Table 4-11. Port CM Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
PCM0	63	61	$\overline{\text{WAIT}}$	Input	–	D-1
PCM1	64	62	CLKOUT	Output		D-2
PCM2	65	63	$\overline{\text{HLDAK}}$	Output		D-2
PCM3	66	64	$\overline{\text{HLDRQ}}$	Input		D-1

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port CM register (PCM)

After reset: 00H (output latch) R/W Address: FFFFF00CH

	7	6	5	4	3	2	1	0
PCM	0	0	0	0	PCM3	PCM2	PCM1	PCM0

PCMn	Output data control (in output mode) (n = 0 to 3)
0	Outputs 0
1	Outputs 1

(2) Port CM mode register (PMCM)

After reset: FFH R/W Address: FFFFF02CH

	7	6	5	4	3	2	1	0
PMCM	1	1	1	1	PMCM3	PMCM2	PMCM1	PMCM0

PMCMn	I/O mode control (n = 0 to 3)
0	Output mode
1	Input mode

(3) Port CM mode control register (PMCCM)

After reset: 00H R/W Address: FFFFF04CH

	7	6	5	4	3	2	1	0
PMCCM	0	0	0	0	PMCCM3	PMCCM2	PMCCM1	PMCCM0

PMCCM3	Specification of PCM3 pin operation mode
0	I/O port
1	$\overline{\text{HLDRQ}}$ input

PMCCM2	Specification of PCM2 pin operation mode
0	I/O port
1	$\overline{\text{HLDAK}}$ output

PMCCM1	Specification of PCM1 pin operation mode
0	I/O port
1	CLKOUT output

PMCCM0	Specification of PCM0 pin operation mode
0	I/O port
1	$\overline{\text{WAIT}}$ input

4.3.9 Port CT

Port CT is a 4-bit port for which I/O settings can be controlled in 1-bit units.

Port CT includes the following alternate-function pins.

Table 4-12. Port CT Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
PCT0	67	65	$\overline{WR0}$	Output	–	D-2
PCT1	68	66	$\overline{WR1}$	Output		D-2
PCT4	69	67	$\overline{RD}$	Output		D-2
PCT6	70	68	ASTB	Output		D-2

Remark GF: 100-pin plastic QFP (14 × 20)
GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port CT register (PCT)

After reset: 00H (output latch) R/W Address: FFFFF00AH

	7	6	5	4	3	2	1	0
PCT	0	PCT6	0	PCT4	0	0	PCT1	PCT0

PCTn	Output data control (in output mode) (n = 0, 1, 4, 6)
0	Outputs 0
1	Outputs 1

(2) Port CT mode register (PMCT)

After reset: FFH R/W Address: FFFFF02AH

	7	6	5	4	3	2	1	0
PMCT	1	PMCT6	1	PMCT4	1	1	PMCT1	PMCT0

PMCTn	I/O mode control (n = 0, 1, 4, 6)
0	Output mode
1	Input mode

(3) Port CT mode control register (PMCCT)

After reset: 00H R/W Address: FFFFF04AH

	7	6	5	4	3	2	1	0
PMCCT	0	PMCCT6	0	PMCCT4	0	0	PMCCT1	PMCCT0

PMCCT6	Specification of PCT6 pin operation mode
0	I/O port
1	ASTB output

PMCCT4	Specification of PCT4 pin operation mode
0	I/O port
1	$\overline{RD}$ output

PMCCT1	Specification of PCT1 pin operation mode
0	I/O port
1	$\overline{WR1}$ output

PMCCT0	Specification of PCT0 pin operation mode
0	I/O port
1	$\overline{WR0}$ output

4.3.10 Port DH

Port DH is a 6-bit port for which I/O settings can be controlled in 1-bit units.

Port DH includes the following alternate-function pins.

Table 4-13. Port DH Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
PDH0	89	87	A16	Output	–	D-2
PDH1	90	88	A17	Output		D-2
PDH2	61	59	A18	Output		D-2
PDH3	62	60	A19	Output		D-2
PDH4	8	6	A20	Output		D-2
PDH5	9	7	A21	Output		D-2

Remark GF: 100-pin plastic QFP (14 × 20)

GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port DH register (PDH)

After reset: 00H (output latch) R/W Address: FFFFF006H

	7	6	5	4	3	2	1	0
PDH	0	0	PDH5	PDH4	PDH3	PDH2	PDH1	PDH0

PDHn	Output data control (in output mode) (n = 0 to 5)
0	Outputs 0
1	Outputs 1

(2) Port DH mode register (PMDH)

After reset: FFH R/W Address: FFFFF026H

	7	6	5	4	3	2	1	0
PMDH	1	1	PMDH5	PMDH4	PMDH3	PMDH2	PMDH1	PMDH0

PMDHn	I/O mode control (n = 0 to 5)
0	Output mode
1	Input mode

(3) Port DH mode control register (PMCDH)

After reset: 00H R/W Address: FFFFF046H

	7	6	5	4	3	2	1	0
PMCDH	0	0	PMCDH5	PMCDH4	PMCDH3	PMCDH2	PMCDH1	PMCDH0

PMCDHn	Specification of PDHn pin operation mode (n = 0 to 5)
0	I/O port
1	Am output (address bus output) (m = 16 to 21)

4.3.11 Port DL

Port DL is a 16-bit port for which I/O settings can be controlled in 1-bit units.

Port DL includes the following alternate-function pins.

Table 4-14. Port DL Alternate-Function Pins

Pin Name	Pin No.		Alternate-Function Pin Name	I/O	Remark	Block Type
	GF	GC				
PDL0	73	71	AD0	I/O	–	D-3
PDL1	74	72	AD1	I/O		D-3
PDL2	75	73	AD2	I/O		D-3
PDL3	76	74	AD3	I/O		D-3
PDL4	77	75	AD4	I/O		D-3
PDL5	78	76	AD5	I/O		D-3
PDL6	79	77	AD6	I/O		D-3
PDL7	80	78	AD7	I/O		D-3
PDL8	81	79	AD8	I/O		D-3
PDL9	82	80	AD9	I/O		D-3
PDL10	83	81	AD10	I/O		D-3
PDL11	84	82	AD11	I/O		D-3
PDL12	85	83	AD12	I/O		D-3
PDL13	86	84	AD13	I/O		D-3
PDL14	87	85	AD14	I/O		D-3
PDL15	88	86	AD15	I/O		D-3

Remark GF: 100-pin plastic QFP (14 × 20)

GC: 100-pin plastic LQFP (fine pitch) (14 × 14)

(1) Port DL register (PDL)

After reset: 0000H (output latch) R/W Address: PDL FFFFF004H,
PDLL FFFFF004H, PDLH FFFFF005H

	15	14	13	12	11	10	9	8
PDL (PDLH)	PDL15	PDL14	PDL13	PDL12	PDL11	PDL10	PDL9	PDL8

	7	6	5	4	3	2	1	0
(PDLL)	PDL7	PDL6	PDL5	PDL4	PDL3	PDL2	PDL1	PDL0

PDLn	Output data control (in output mode) (n = 0 to 15)
0	Outputs 0
1	Outputs 1

Remarks 1. The PDL register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PDL register as the PDLH register and the lower 8 bits as the PDLL register, PDL can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the PDL register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PDLH register.

(2) Port DL mode register (PMDL)

After reset: FFFFH R/W Address: PMDL FFFFF024H,
PMDLL FFFFF024H, PMDLH FFFFF025H

	15	14	13	12	11	10	9	8
PMDL (PMDLH)	PMDL15	PMDL14	PMDL13	PMDL12	PMDL11	PMDL10	PMDL9	PMDL8

	7	6	5	4	3	2	1	0
(PMDLL)	PMDL7	PMDL6	PMDL5	PMDL4	PMDL3	PMDL2	PMDL1	PMDL0

PMDLn	I/O mode control (n = 0 to 15)
0	Output mode
1	Input mode

Remarks 1. The PMDL register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PMDL register as the PMDLH register and the lower 8 bits as the PMDLL register, PMDL can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the PMDL register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMDLH register.

(3) Port DL mode control register (PMCDL)

After reset: 0000H R/W Address: PMCDL FFFFF044H, PMCDLL FFFFF044H, PMCDLH FFFFF045H

	15	14	13	12	11	10	9	8
PMCDL (PMCDLH)	PMCDL15	PMCDL14	PMCDL13	PMCDL12	PMCDL11	PMCDL10	PMCDL9	PMCDL8

	7	6	5	4	3	2	1	0
(PMCDLL)	PMCDL7	PMCDL6	PMCDL5	PMCDL4	PMCDL3	PMCDL2	PMCDL1	PMCDL0

PMCDLn	Specification of PDLn pin operation mode (n = 0 to 15)
0	I/O port
1	ADn I/O (address/data bus I/O)

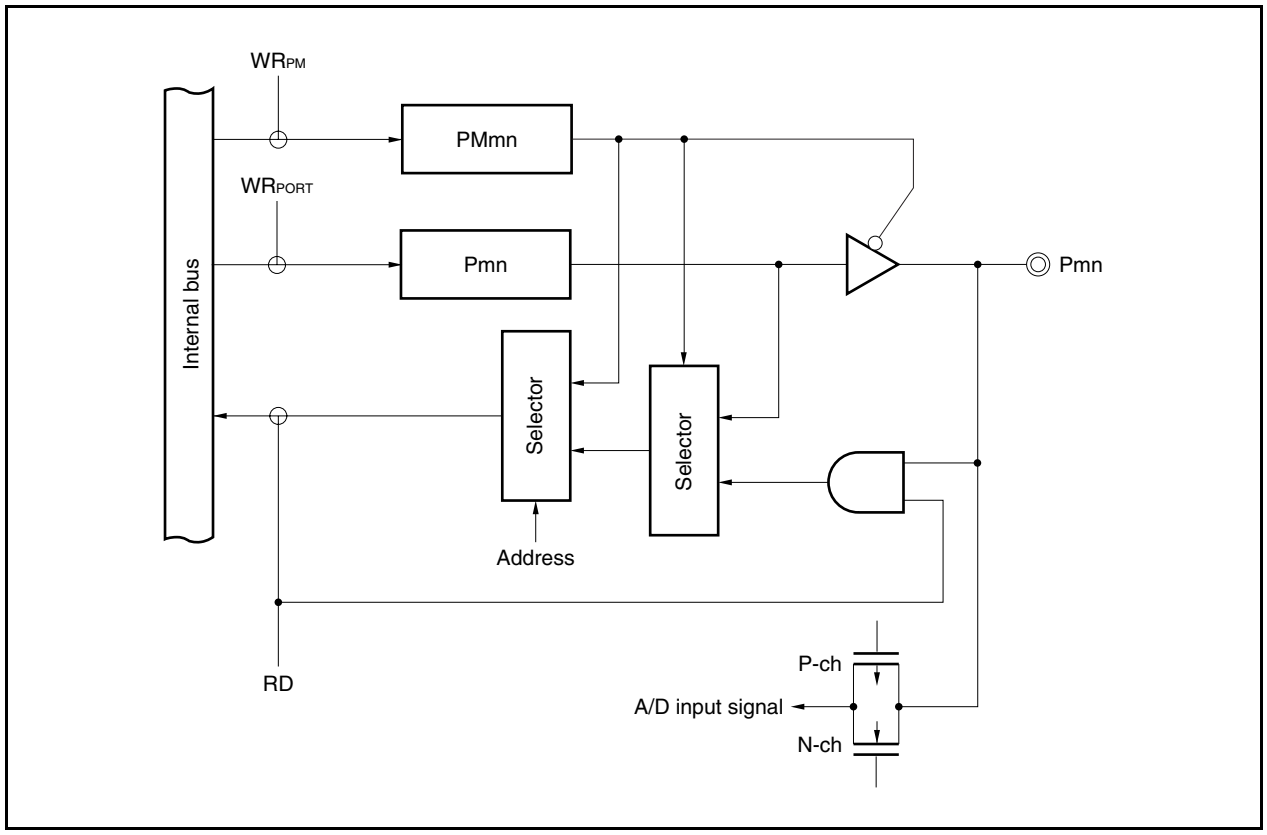
Caution When the SMSEL bit of the EXIMC register = 1 (separate mode) and the BS30 to BS00 bits of the BSC register = 0 (8-bit bus width), do not specify the AD8 to AD15 pins.

Remarks 1. The PMCDL register can be read or written in 16-bit units.
However, when using the higher 8 bits of the PMCDL register as the PMCDLH register and the lower 8 bits as the PMCDLL register, PMCDL can be read or written in 8-bit or 1-bit units.

2. To read/write bits 8 to 15 of the PMCDL register in 8-bit or 1-bit units, specify them as bits 0 to 7 of the PMCDLH register.

4.4 Block Diagrams

Figure 4-3. Block Diagram of Type A-1



<R> **Figure 4-4. Block Diagram of Type A-2**

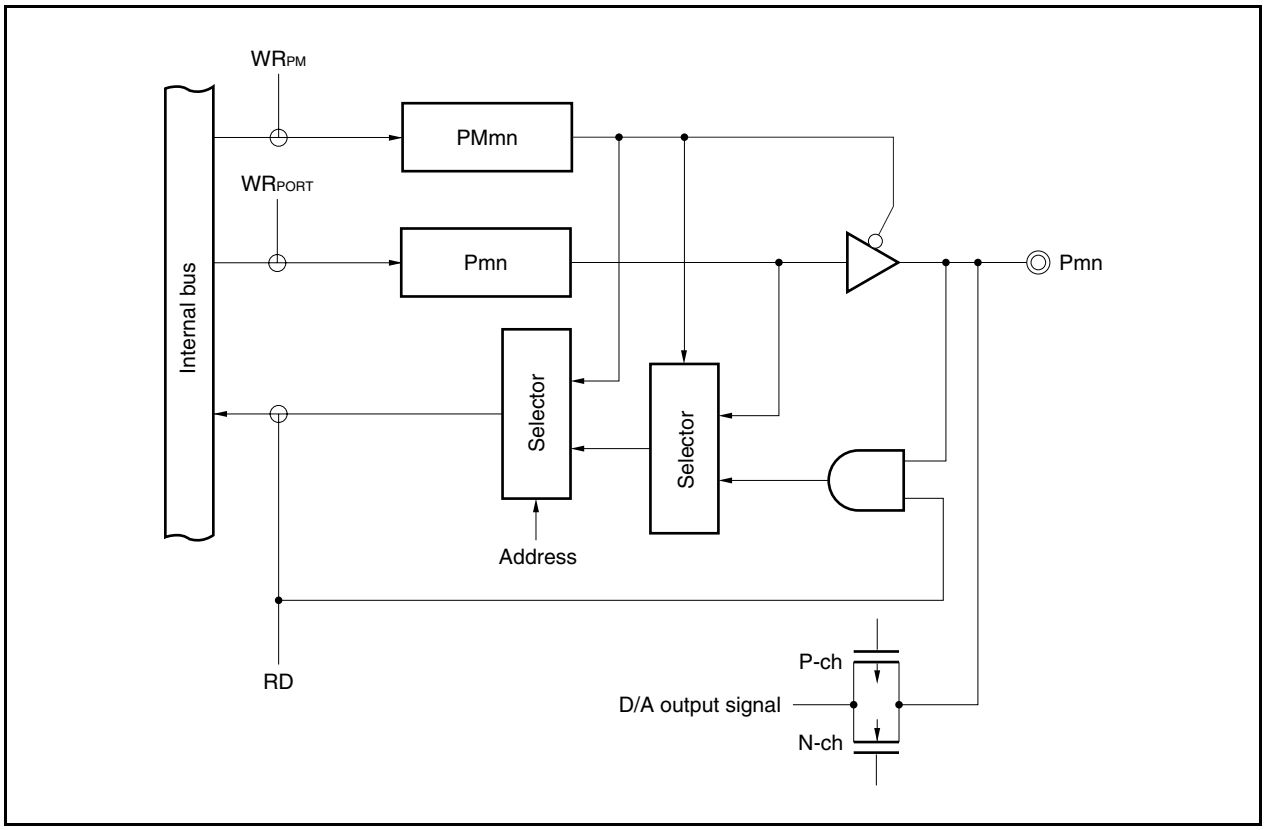


Figure 4-5. Block Diagram of Type B-1

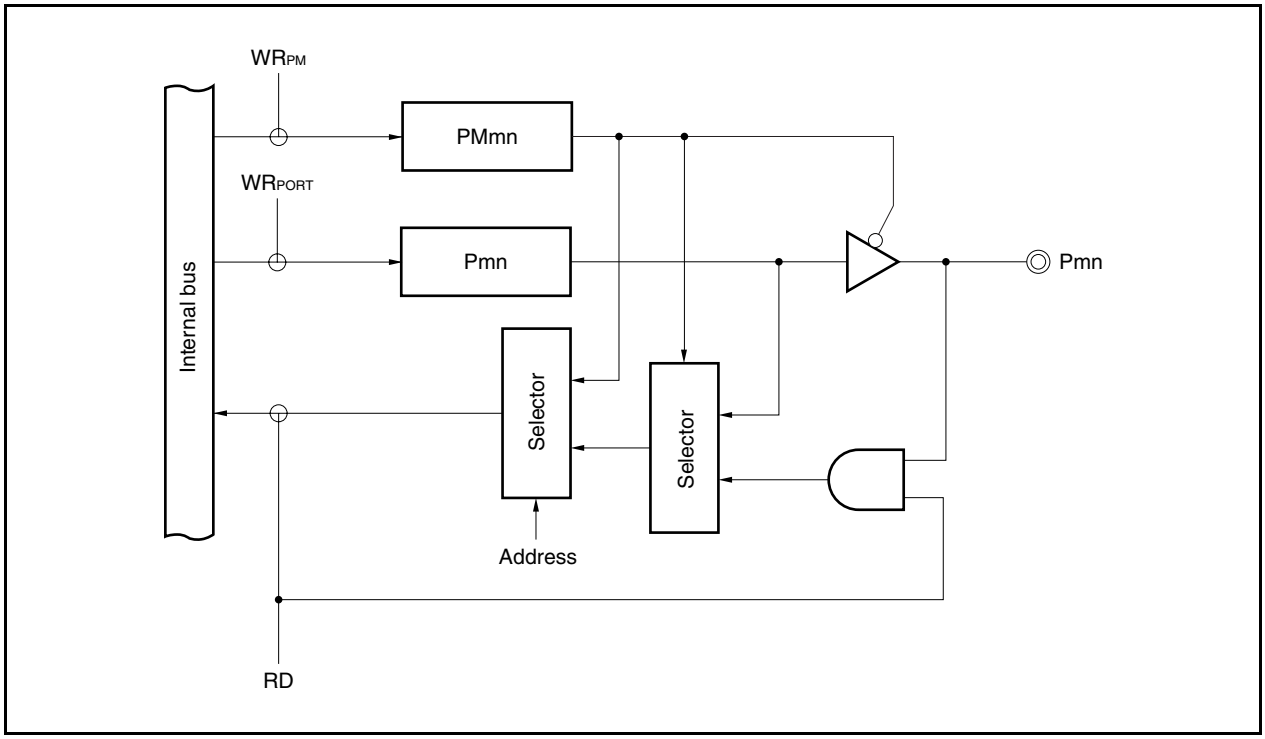


Figure 4-6. Block Diagram of Type D-1

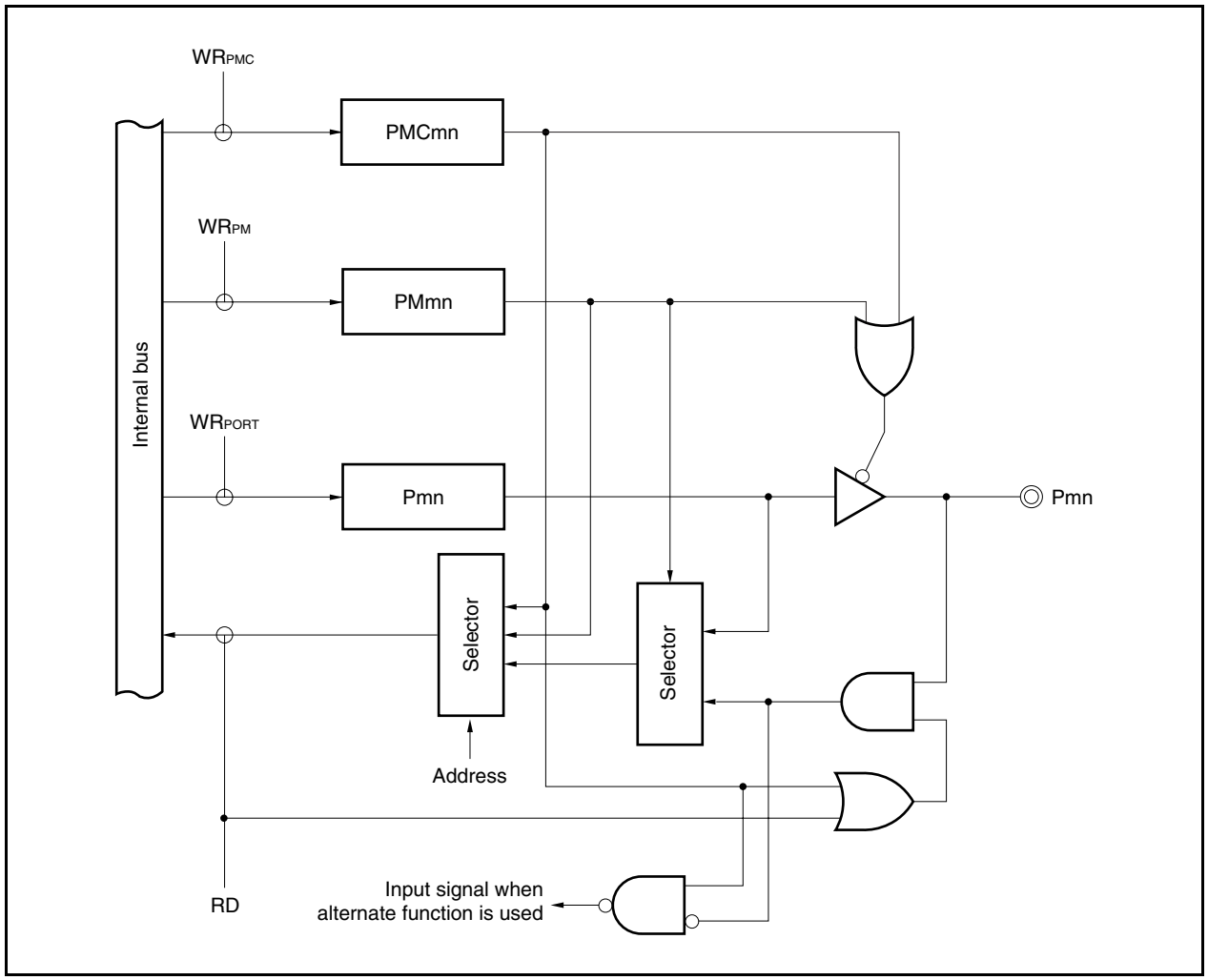


Figure 4-7. Block Diagram of Type D-2

<R>

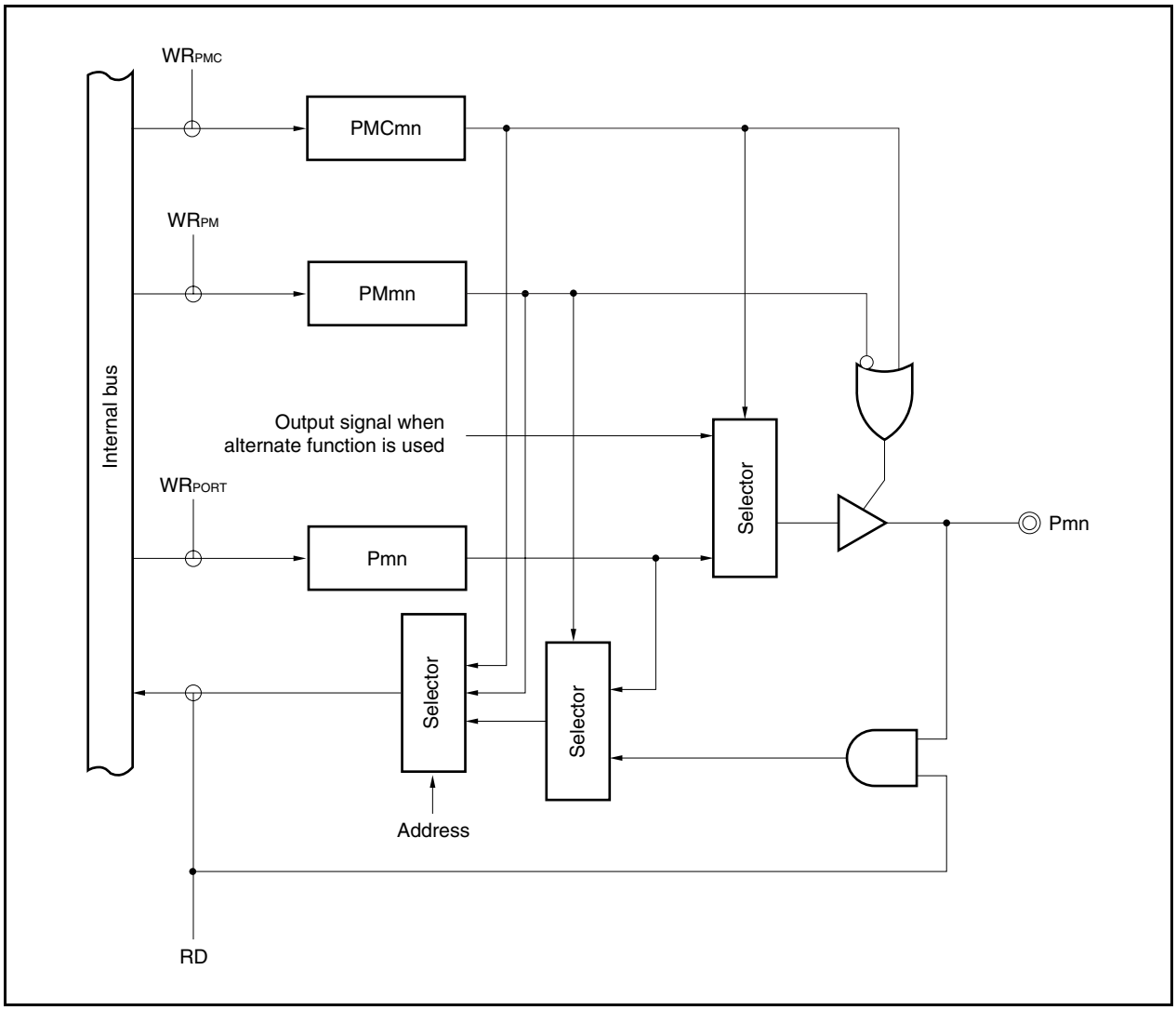


Figure 4-8. Block Diagram of Type D-3

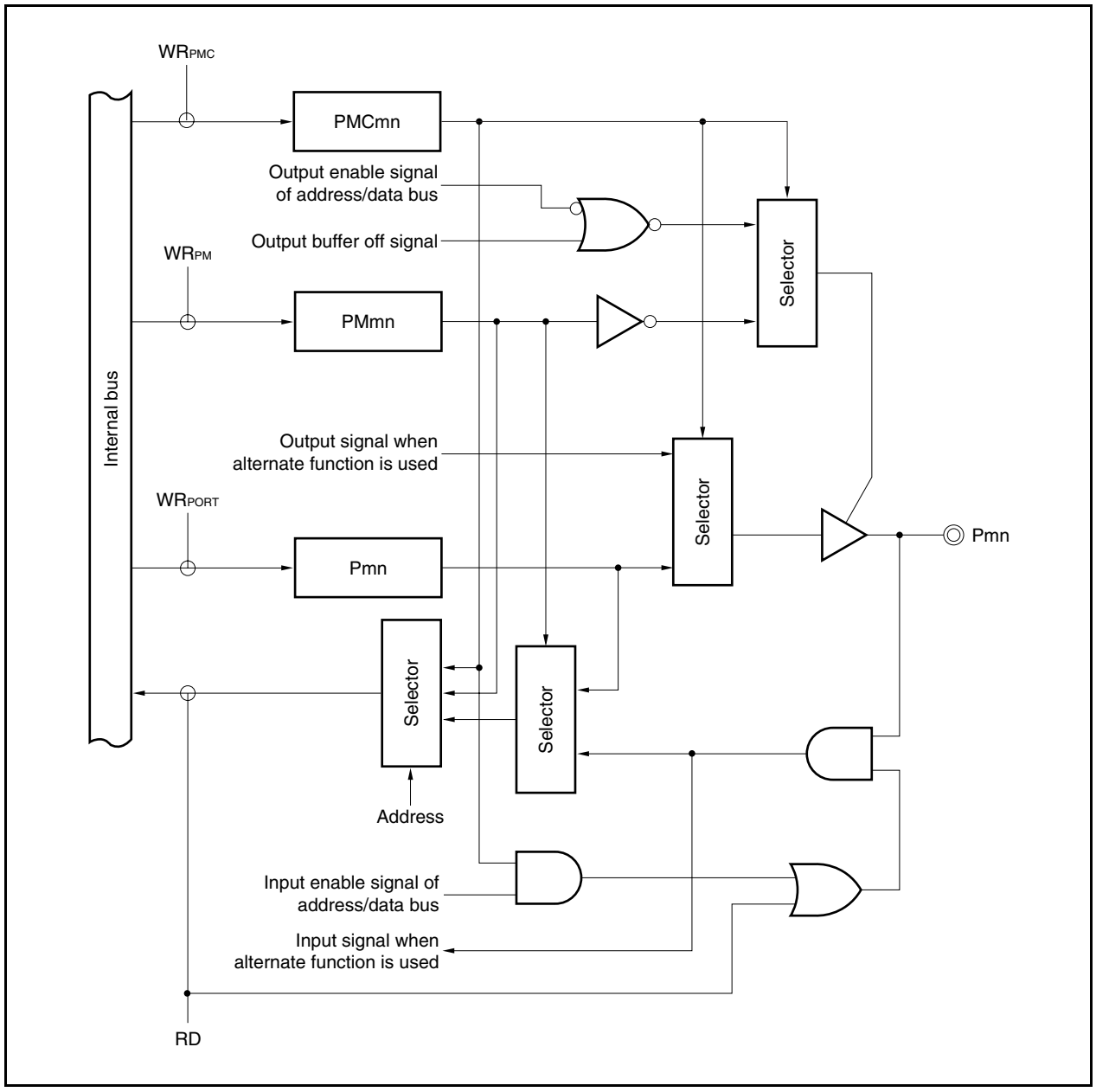
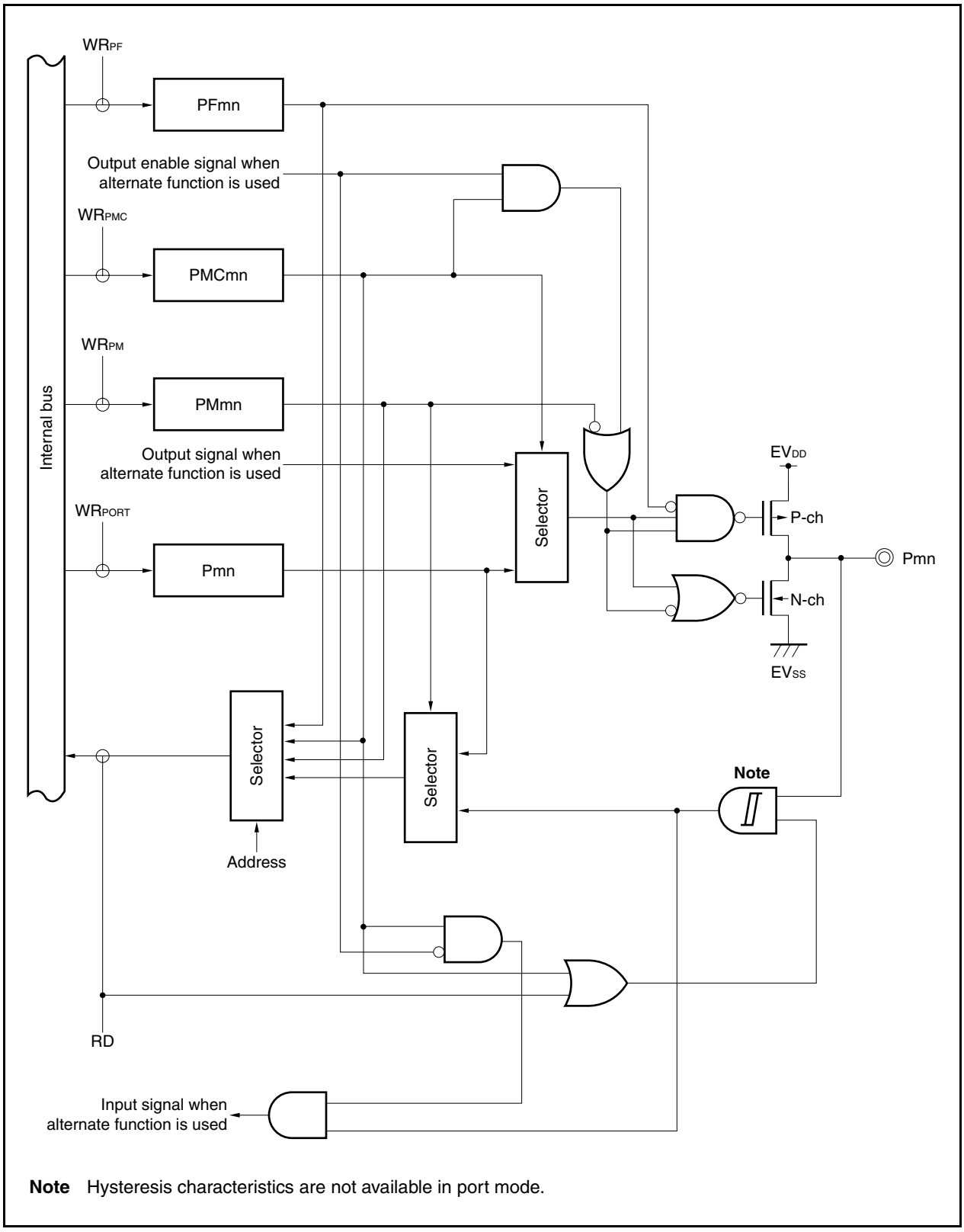


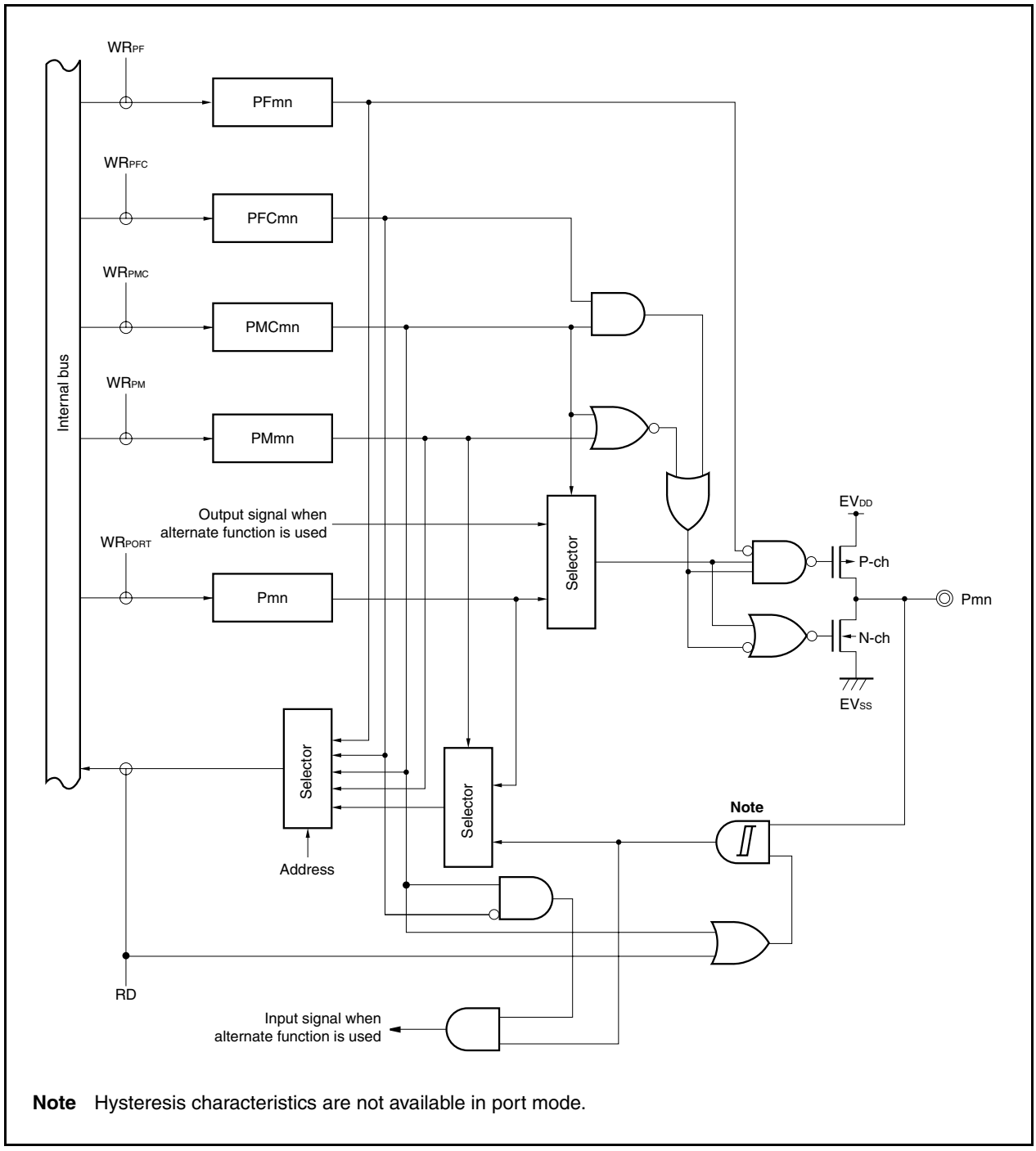
Figure 4-9. Block Diagram of Type E-3



<R>

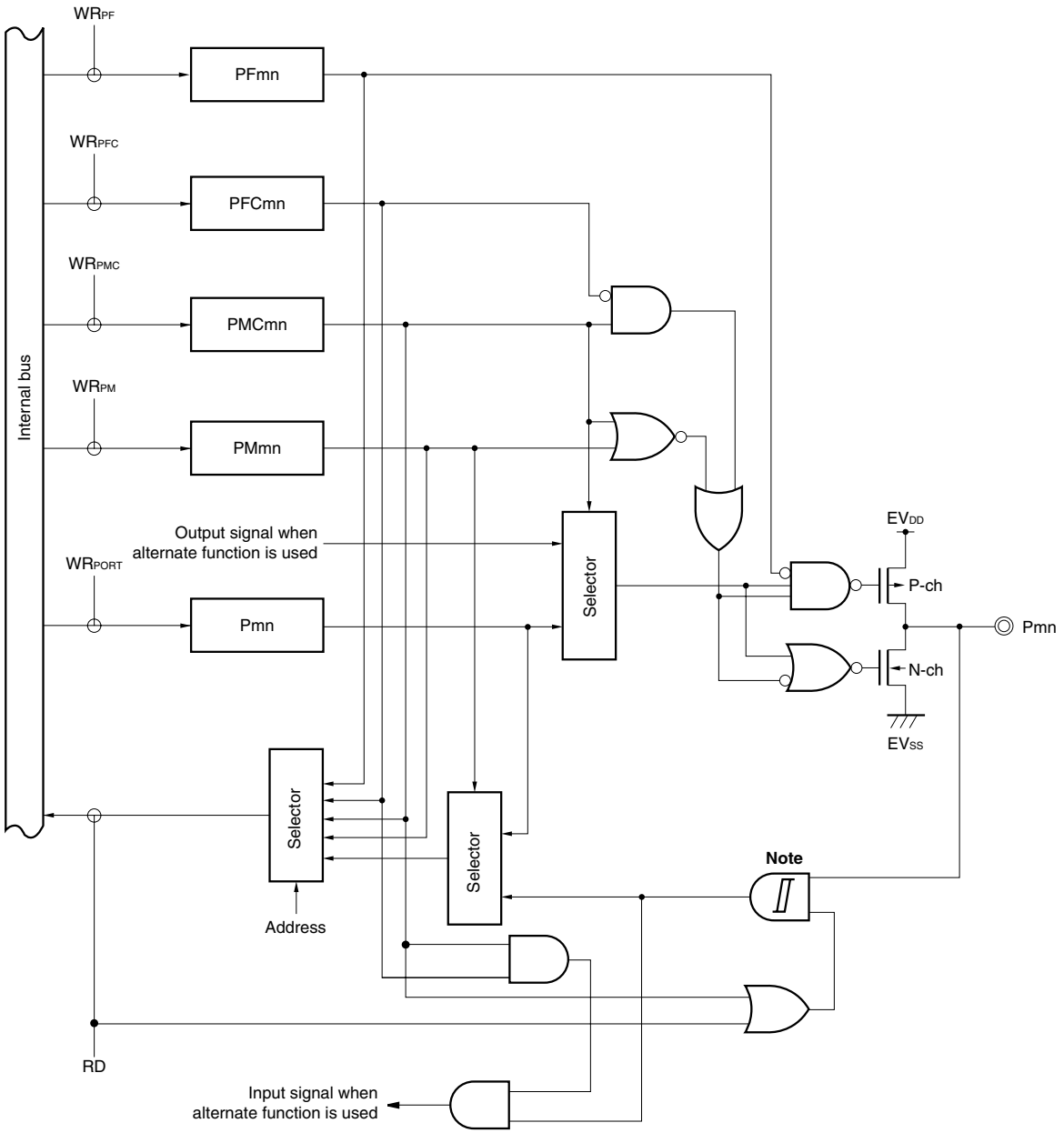
Note Hysteresis characteristics are not available in port mode.

<R> **Figure 4-10. Block Diagram of Type G-1**



Note Hysteresis characteristics are not available in port mode.

Figure 4-11. Block Diagram of Type G-2



Note Hysteresis characteristics are not available in port mode.

Figure 4-12. Block Diagram of Type G-3

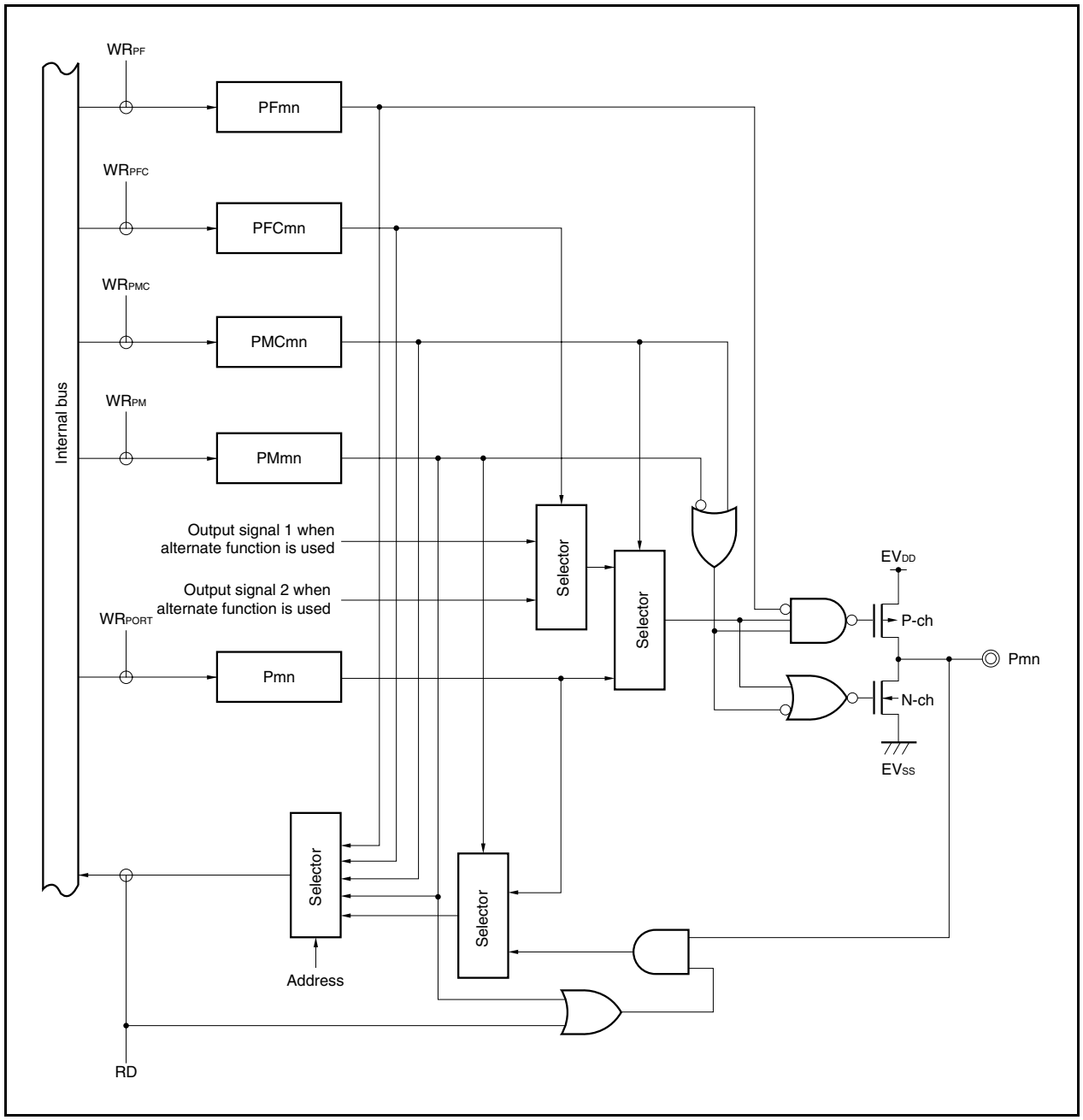
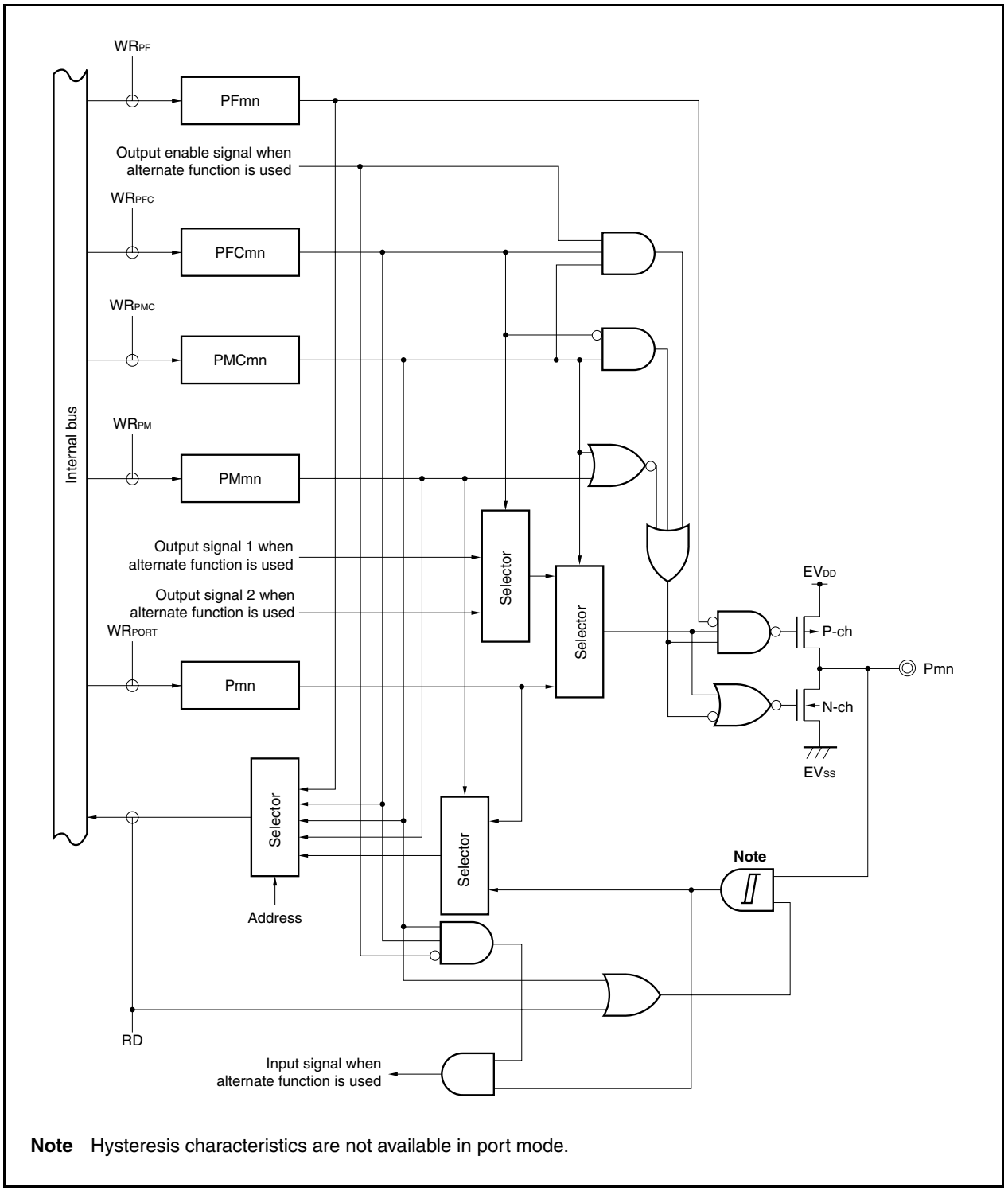
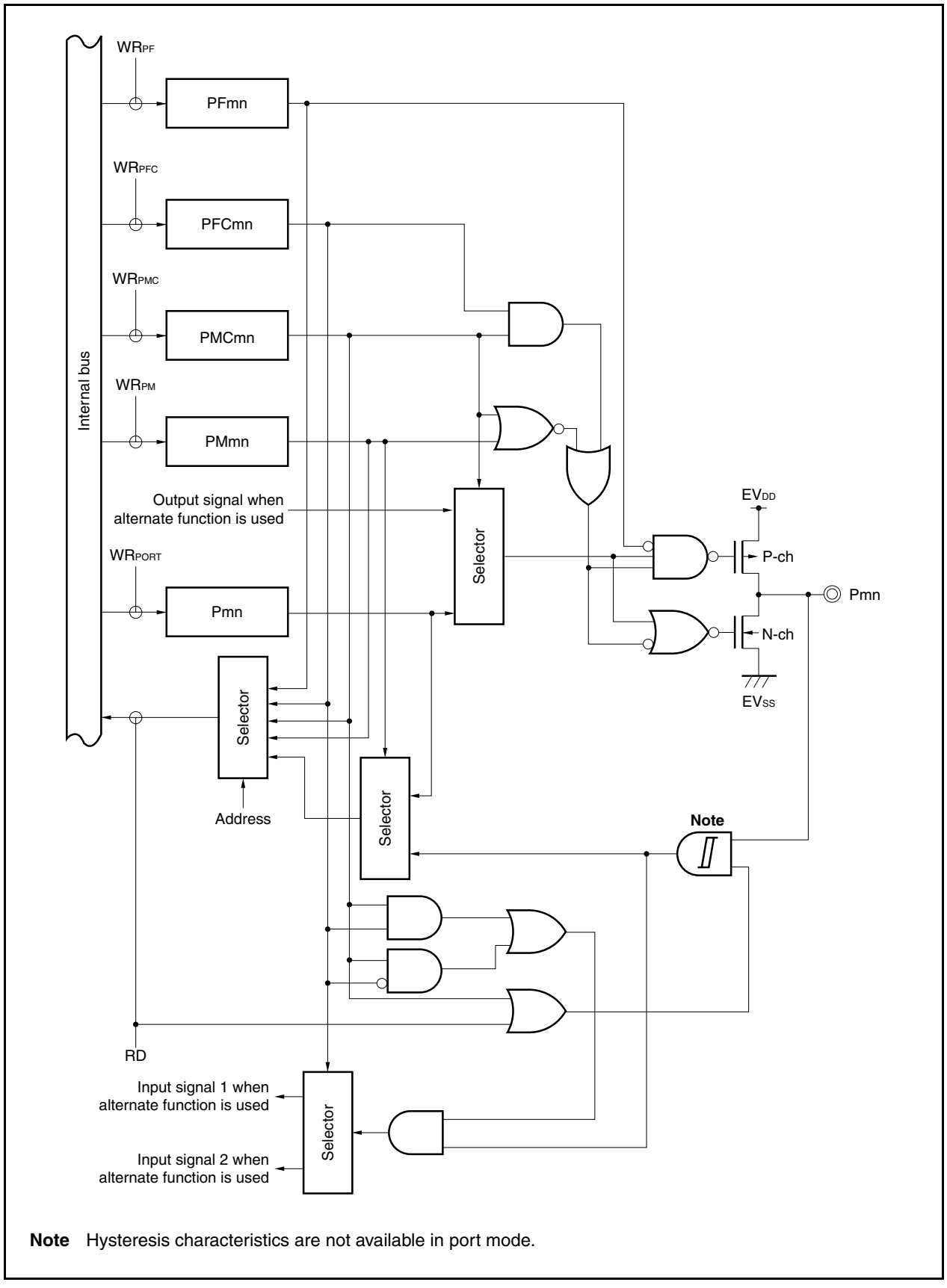


Figure 4-13. Block Diagram of Type G-5



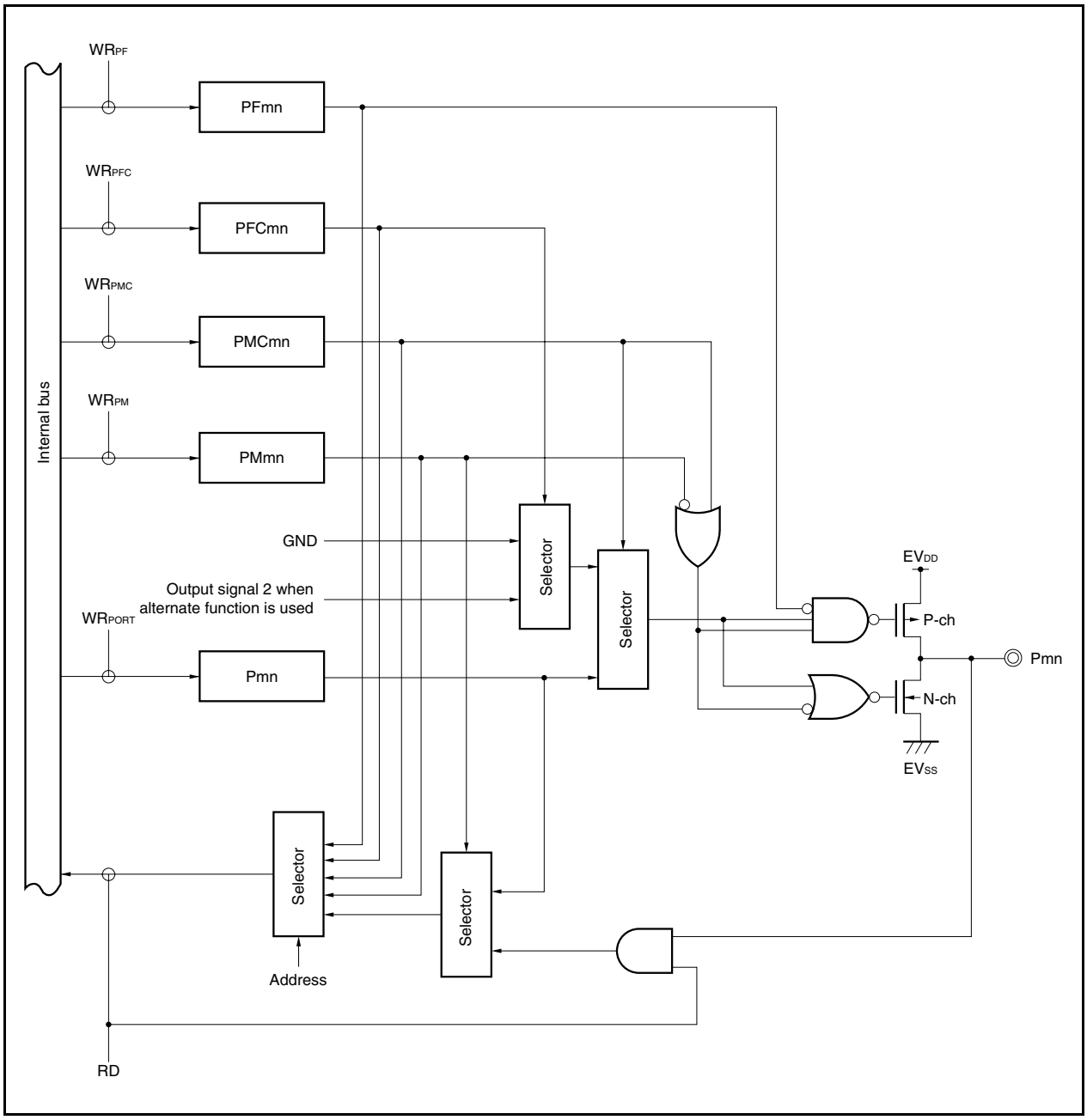
Note Hysteresis characteristics are not available in port mode.

Figure 4-14. Block Diagram of Type G-6



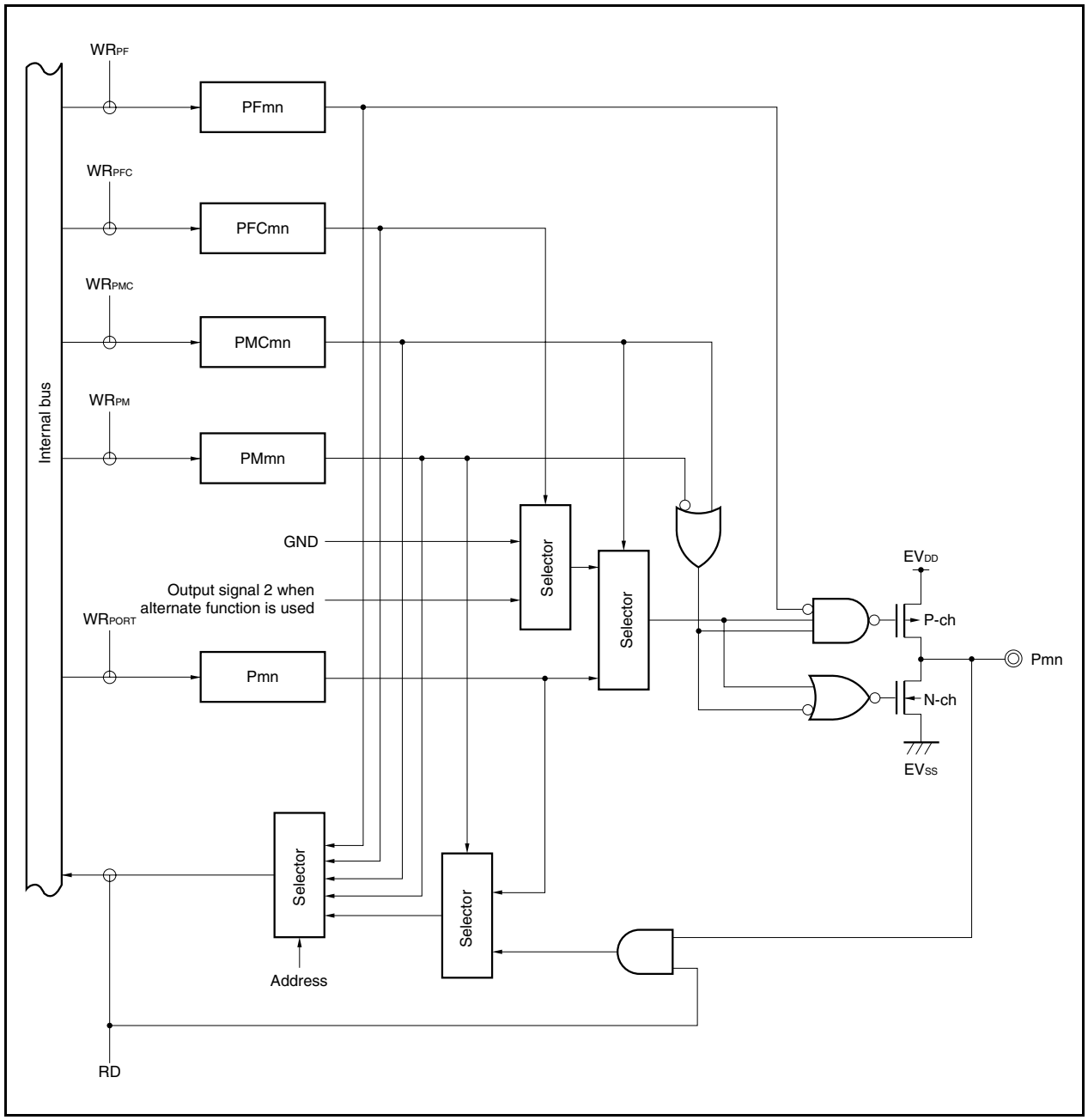
Note Hysteresis characteristics are not available in port mode.

Figure 4-15. Block Diagram of Type G-7



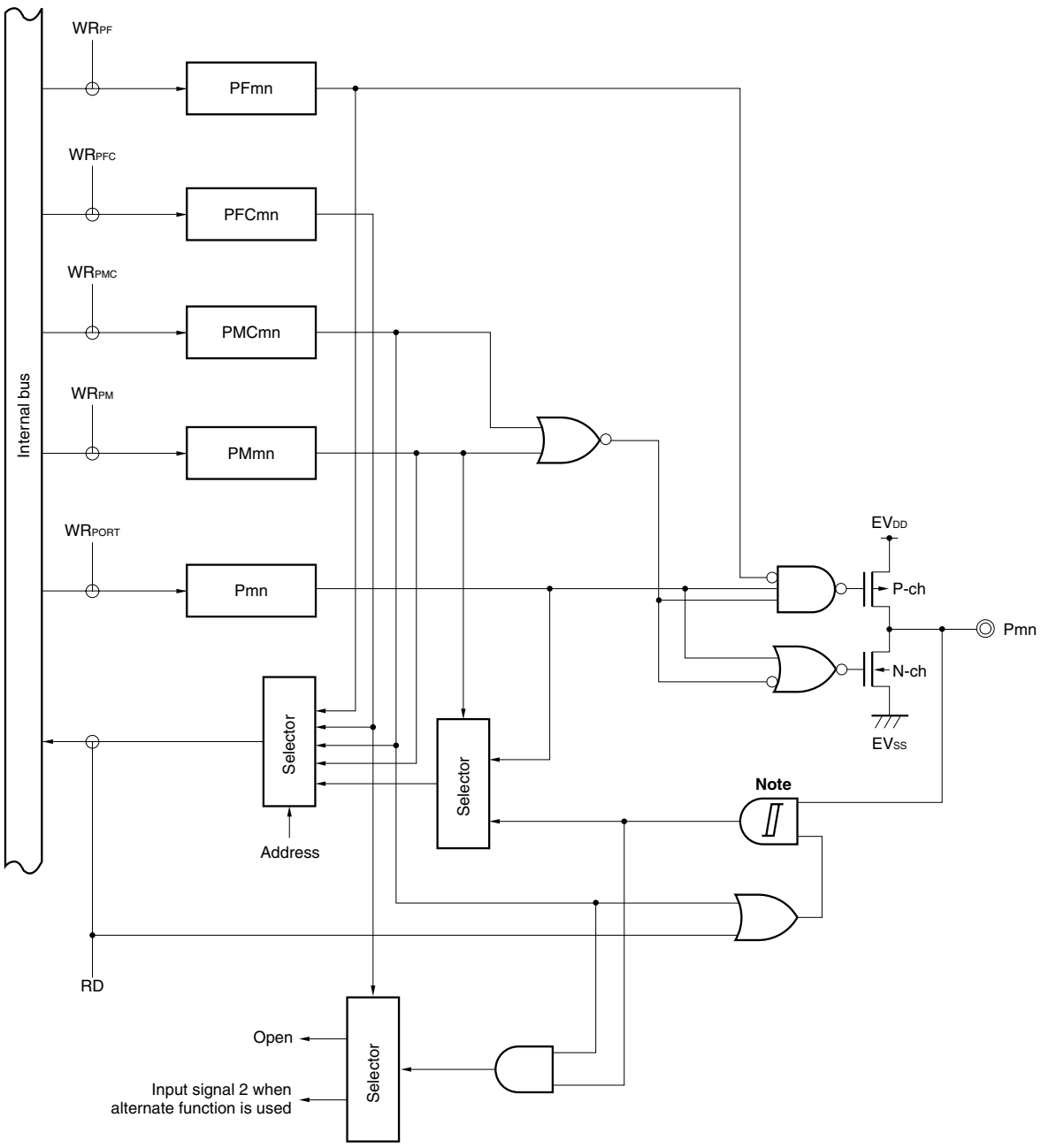
<R>

Figure 4-16. Block Diagram of Type G-8



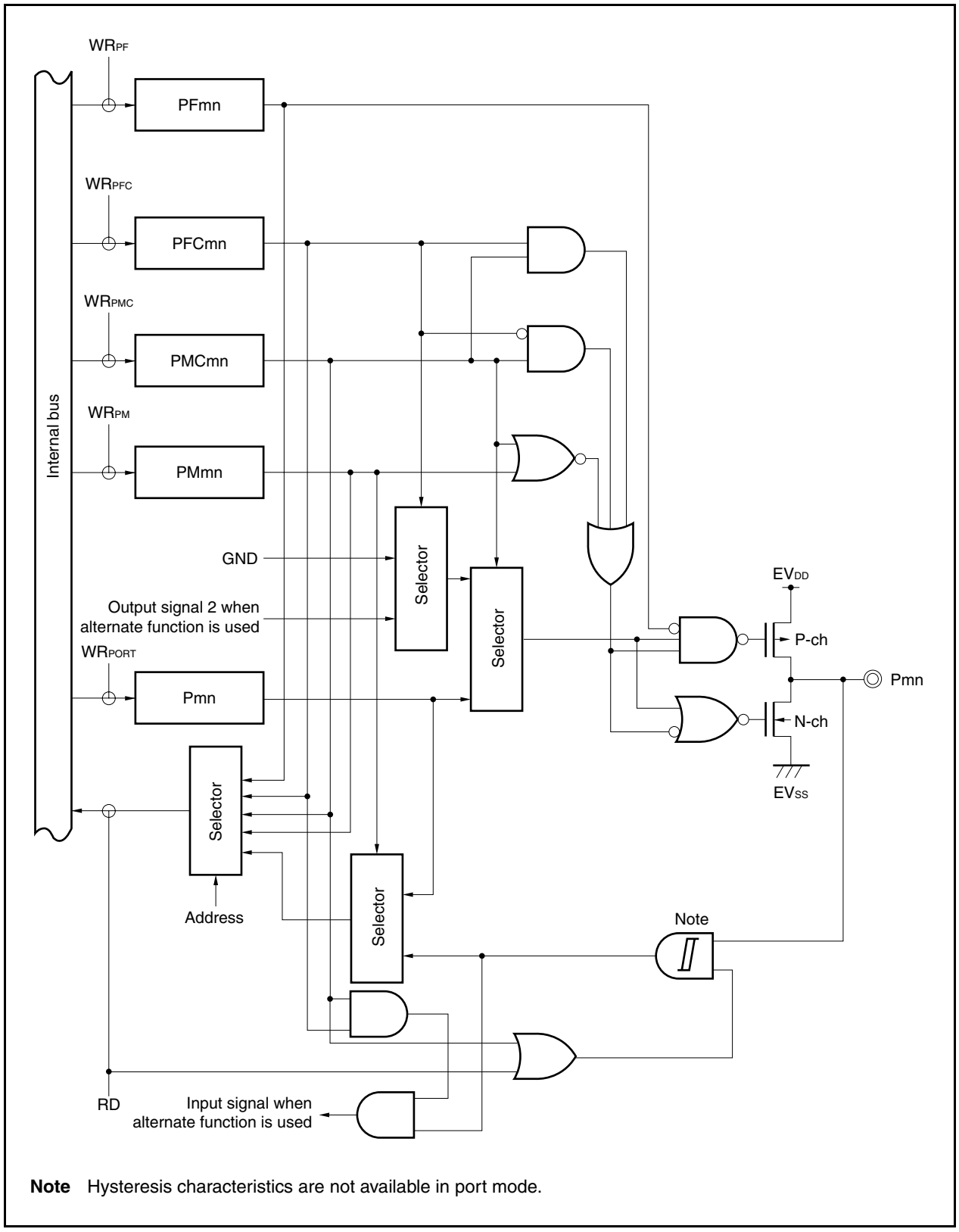
<R>

Figure 4-17. Block Diagram of Type G-9



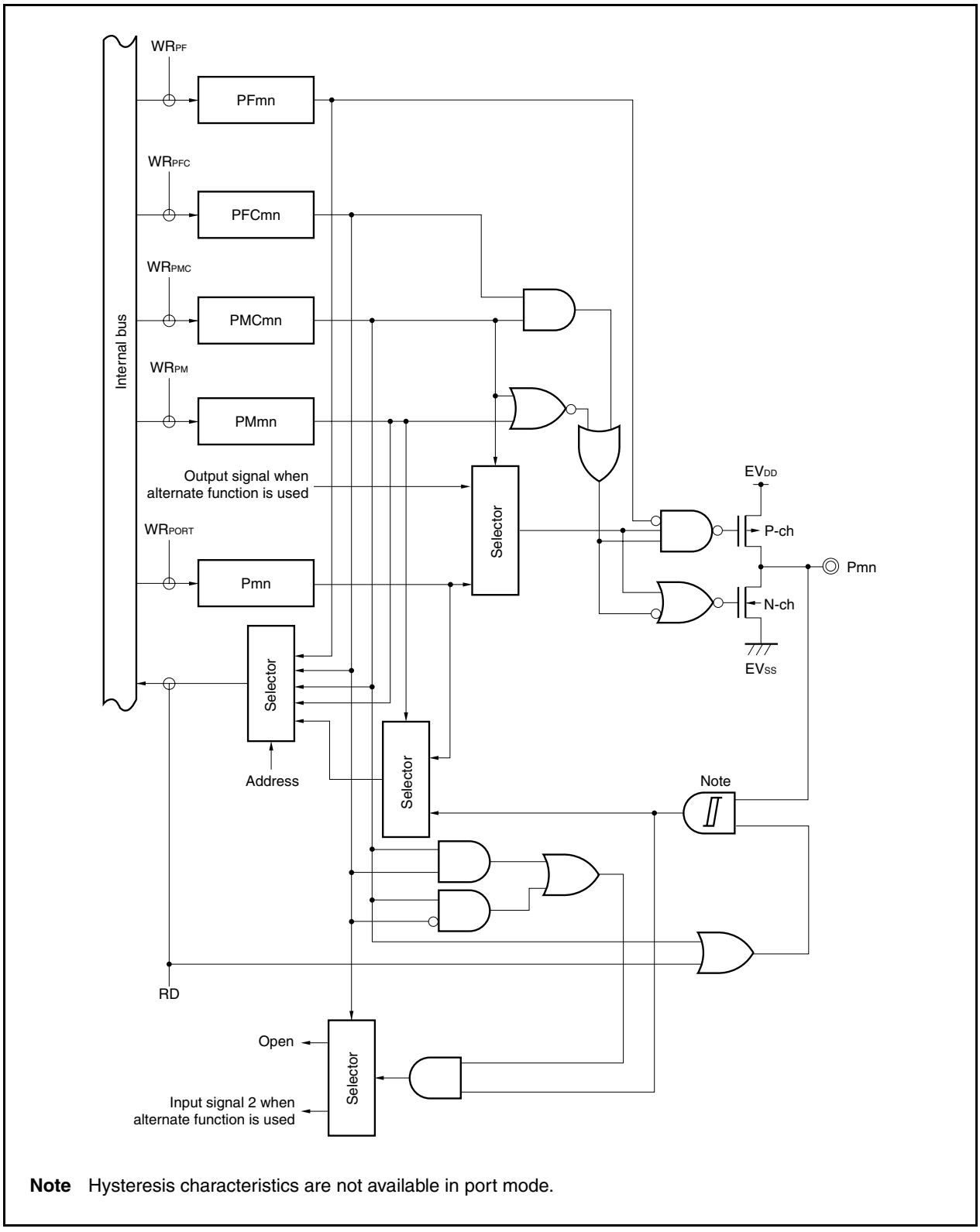
Note Hysteresis characteristics are not available in port mode.

Figure 4-18. Block Diagram of Type G-10



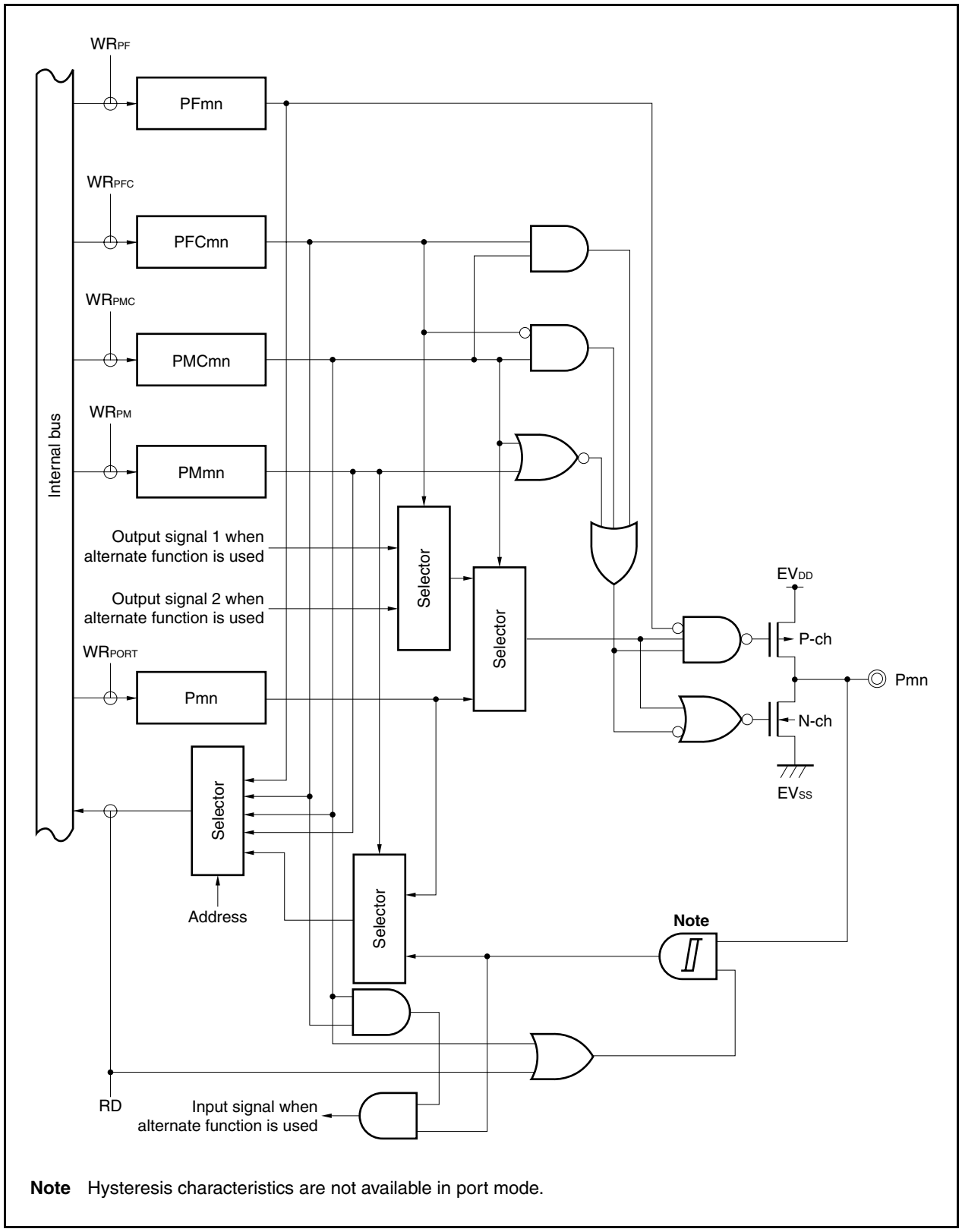
Note Hysteresis characteristics are not available in port mode.

Figure 4-19. Block Diagram of Type G-11



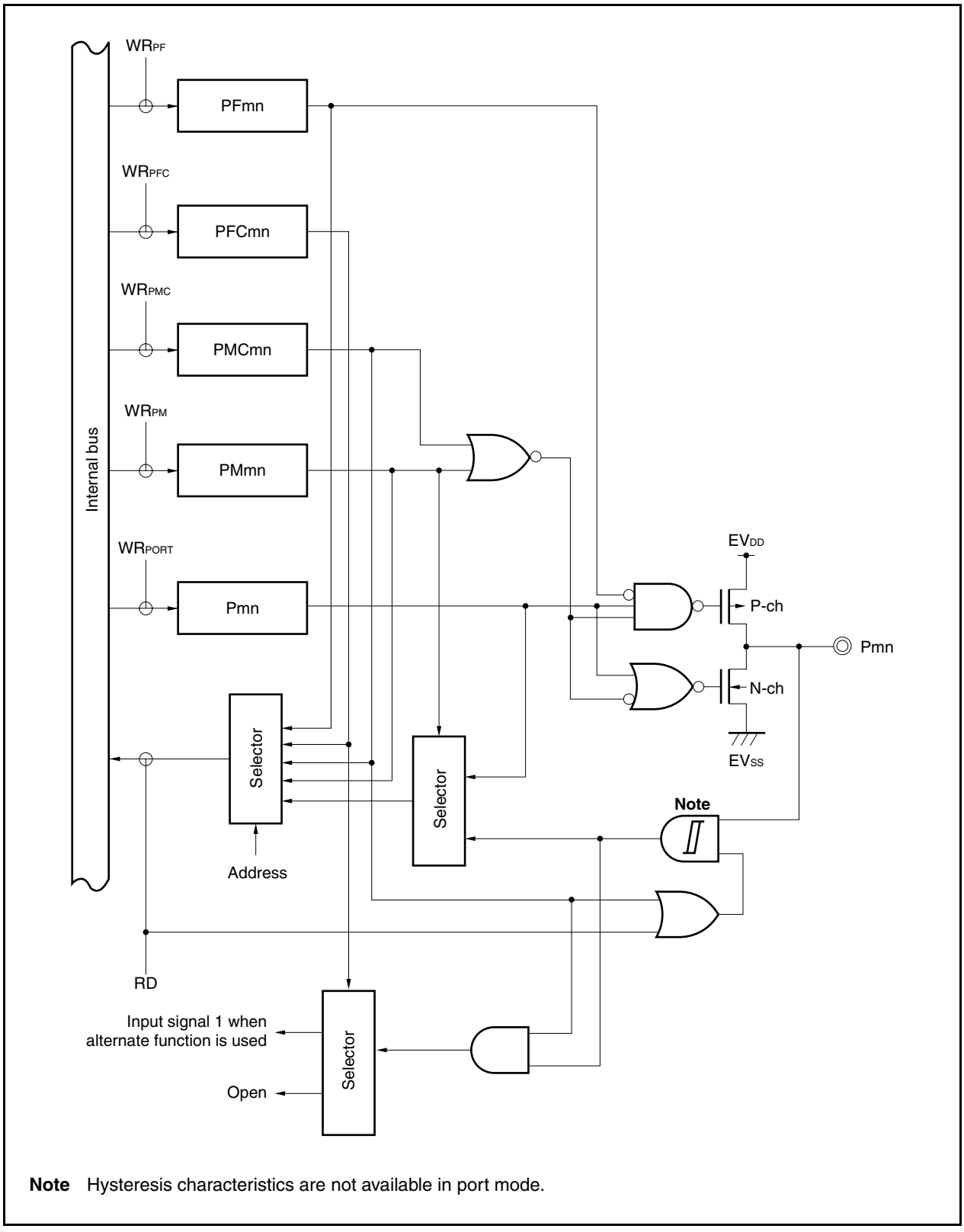
Note Hysteresis characteristics are not available in port mode.

<R> **Figure 4-20. Block Diagram of Type G-12**



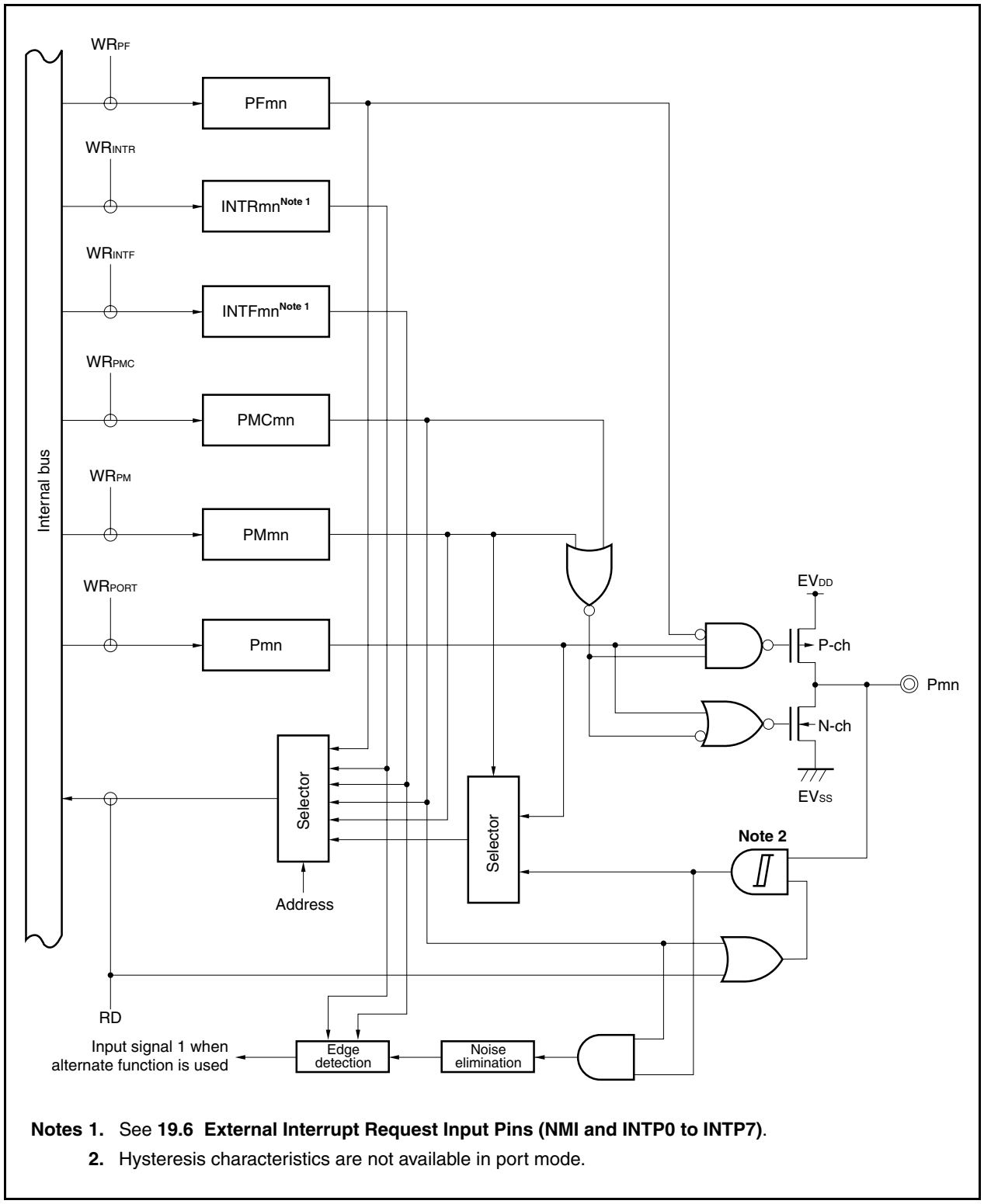
Note Hysteresis characteristics are not available in port mode.

<R> **Figure 4-21. Block Diagram of Type G-14**



Note Hysteresis characteristics are not available in port mode.

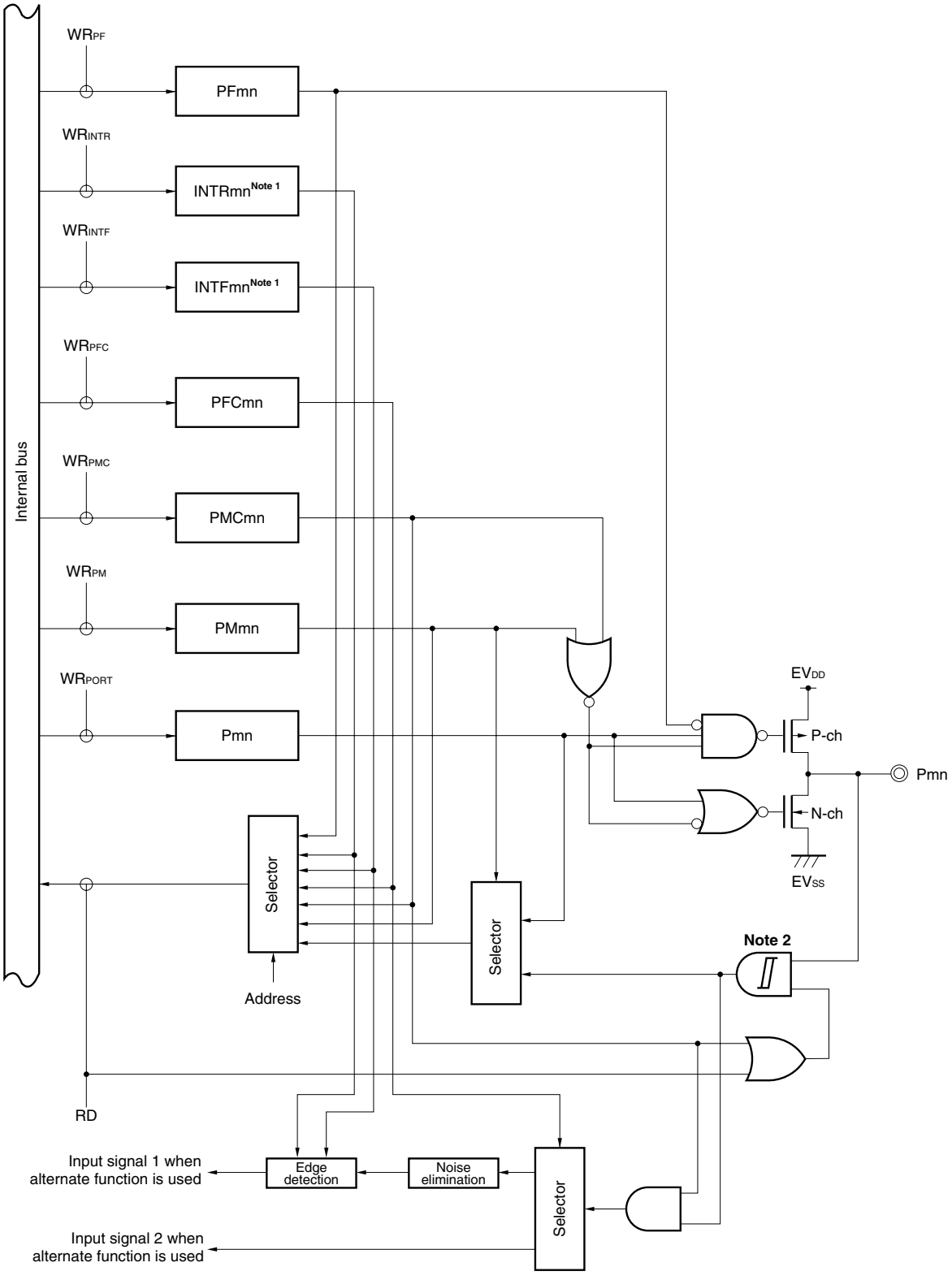
Figure 4-22. Block Diagram of Type L-1



Notes 1. See **19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7)**.

2. Hysteresis characteristics are not available in port mode.

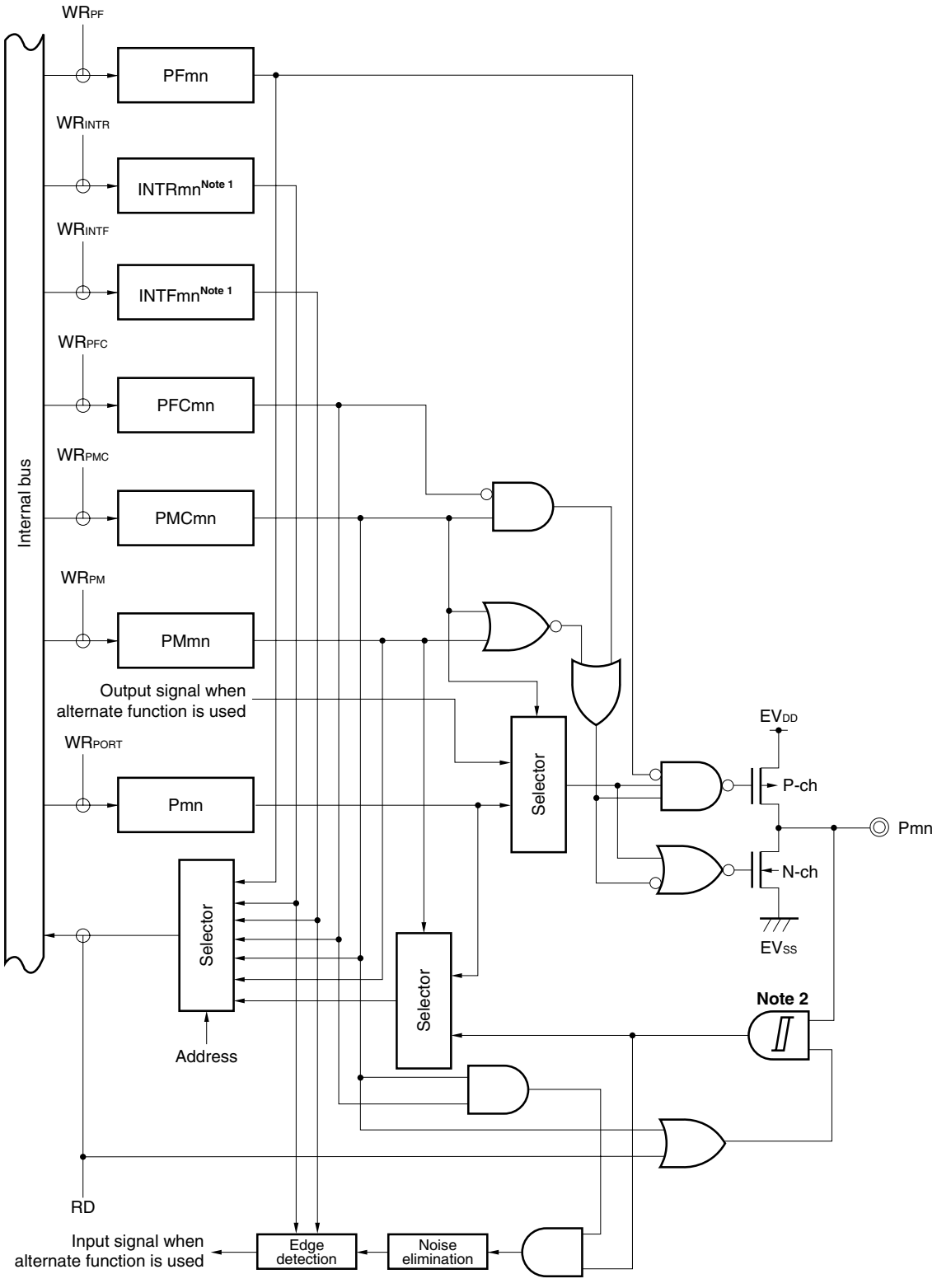
Figure 4-23. Block Diagram of Type N-1



Notes 1. See **19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7)**.

2. Hysteresis characteristics are not available in port mode.

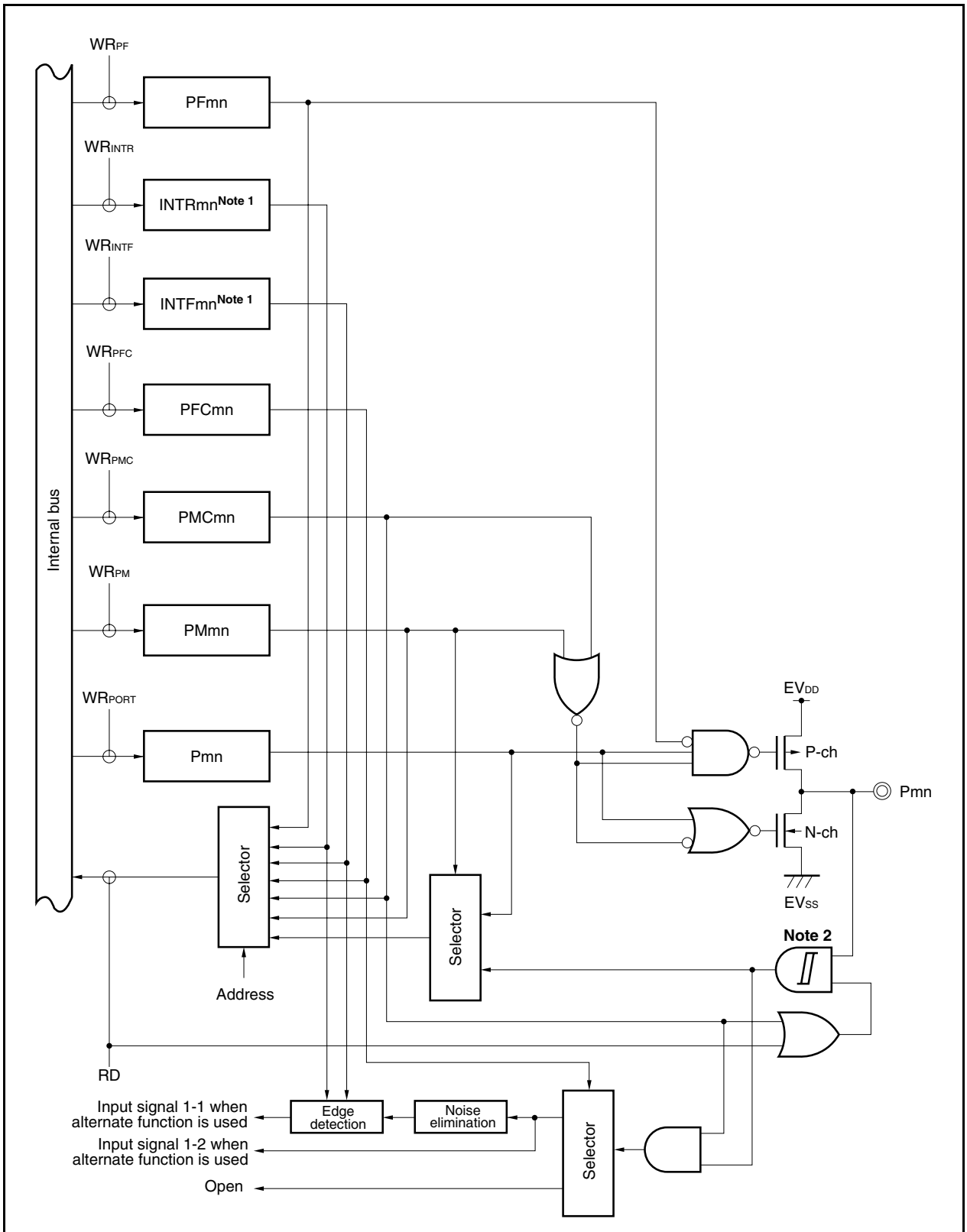
Figure 4-24. Block Diagram of Type N-2



Notes 1. See **19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7)**.

2. Hysteresis characteristics are not available in port mode.

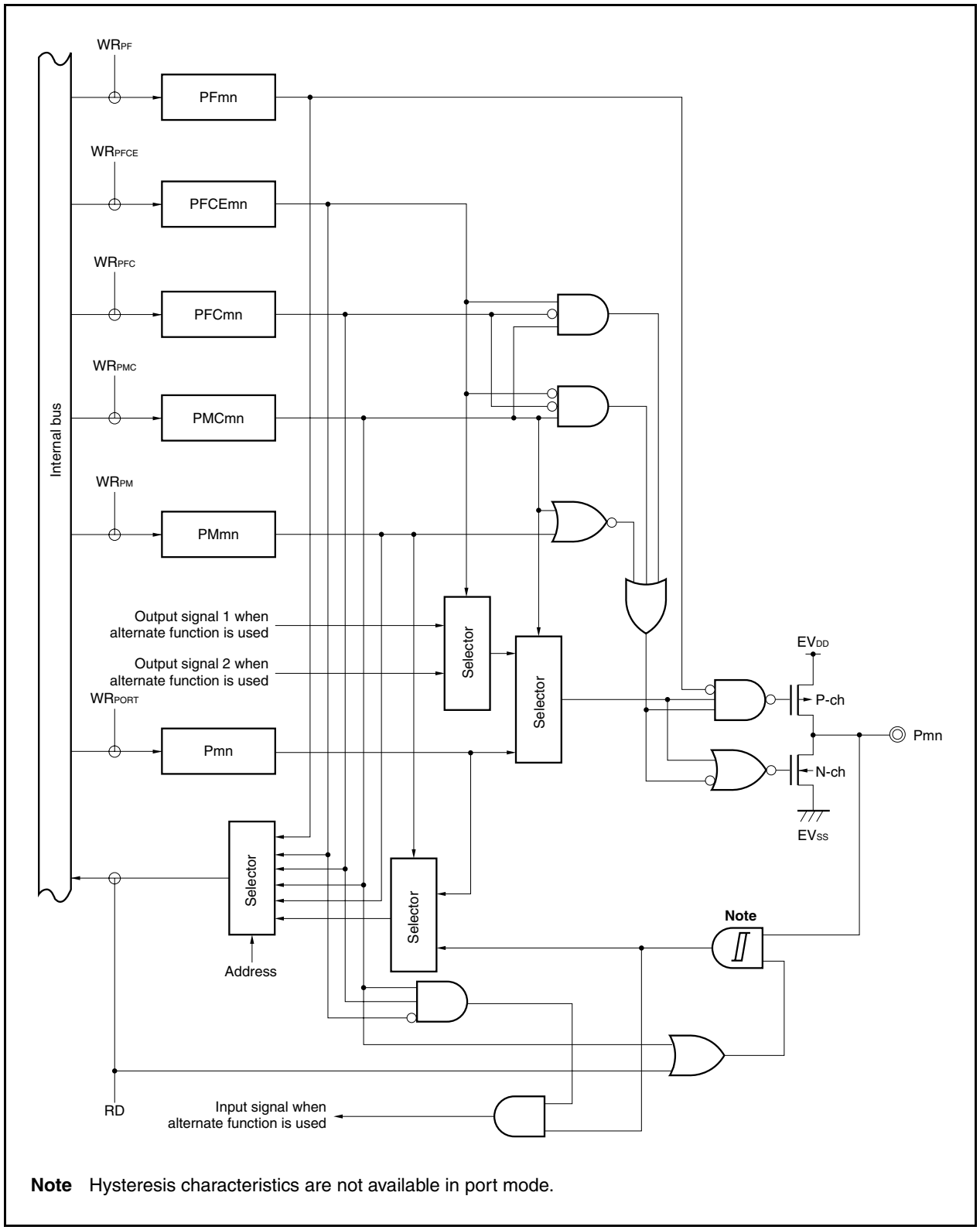
Figure 4-25. Block Diagram of Type N-4



Notes 1. See **19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7)**.

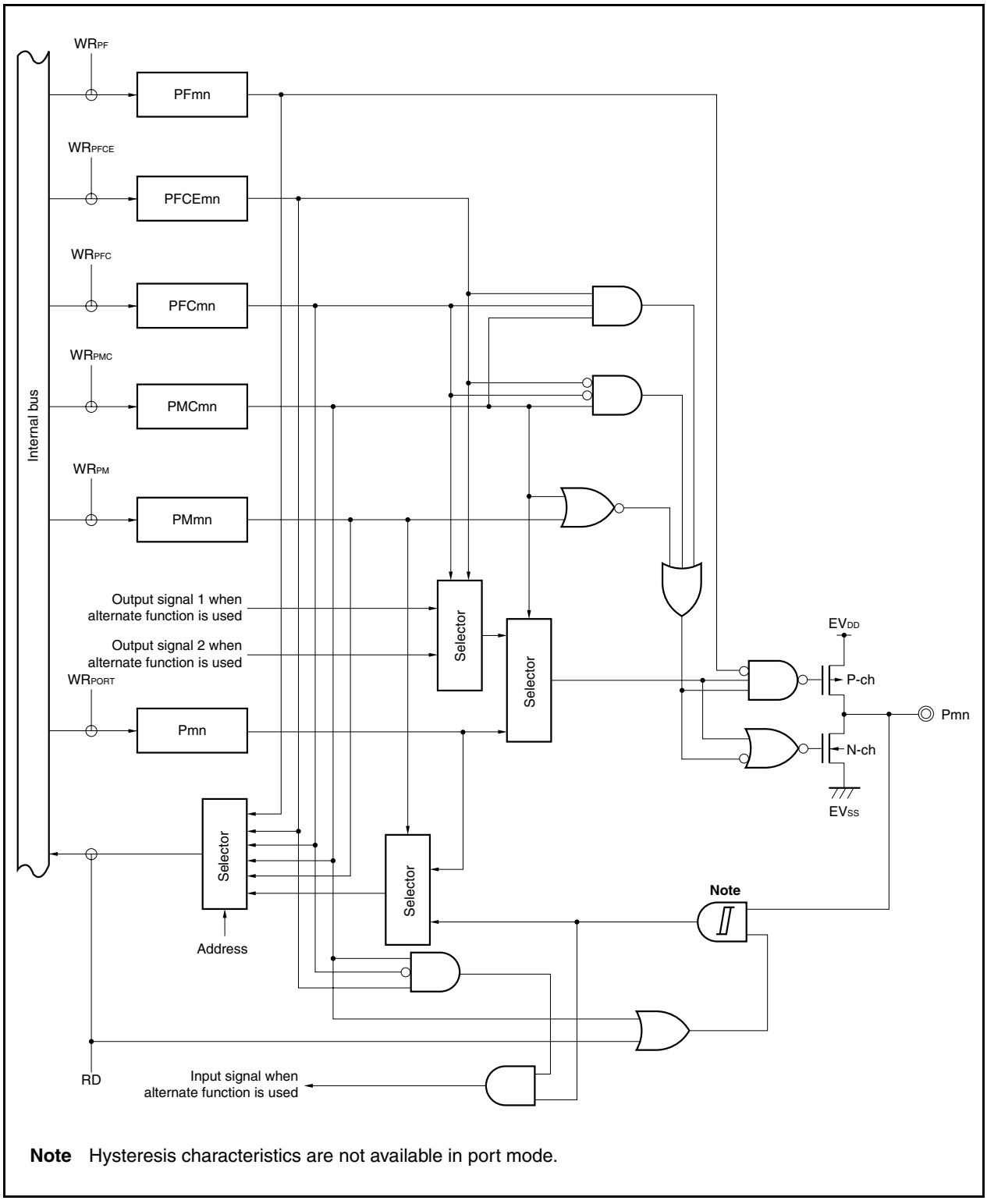
2. Hysteresis characteristics are not available in port mode.

Figure 4-26. Block Diagram of Type U-12



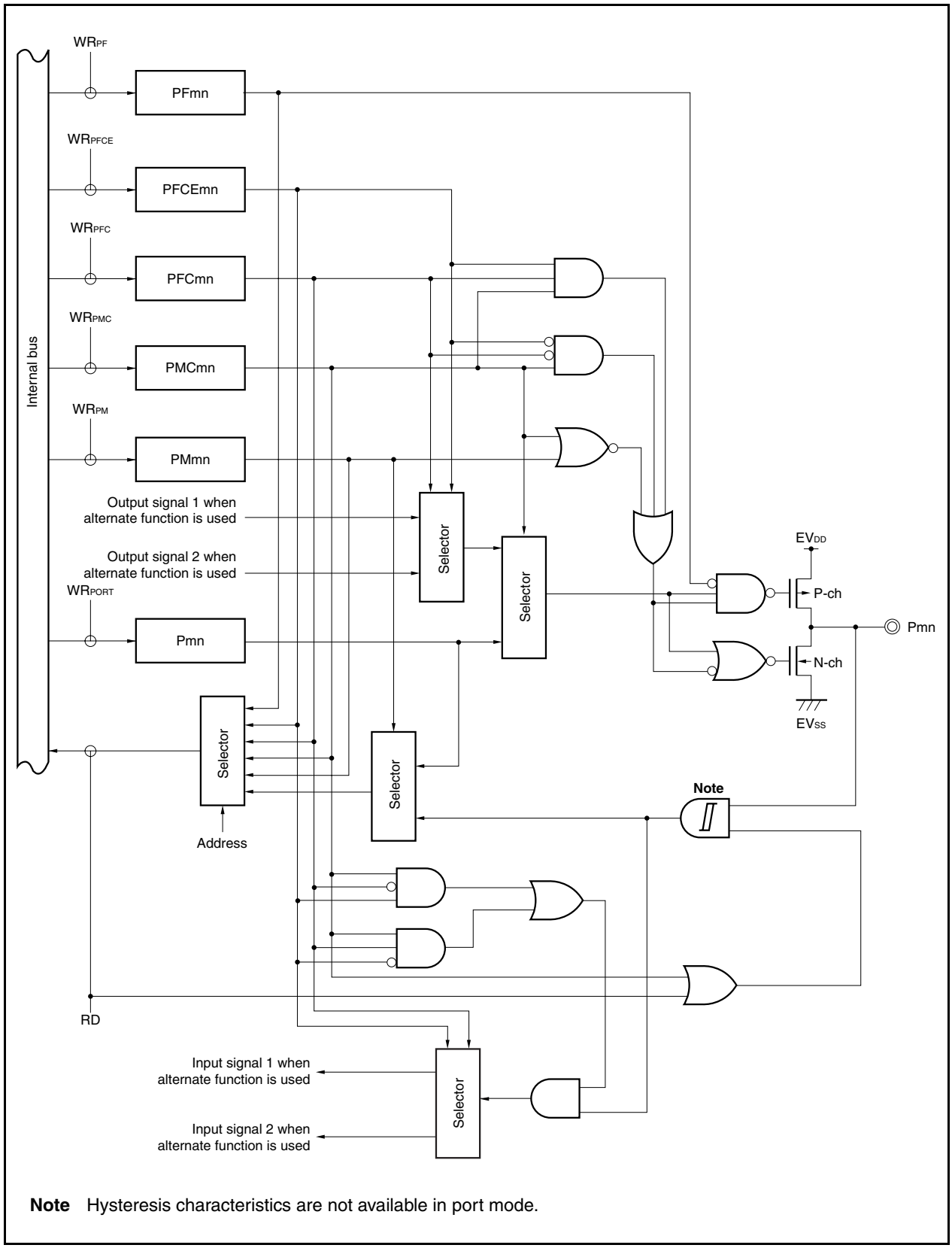
Note Hysteresis characteristics are not available in port mode.

Figure 4-27. Block Diagram of Type U-13



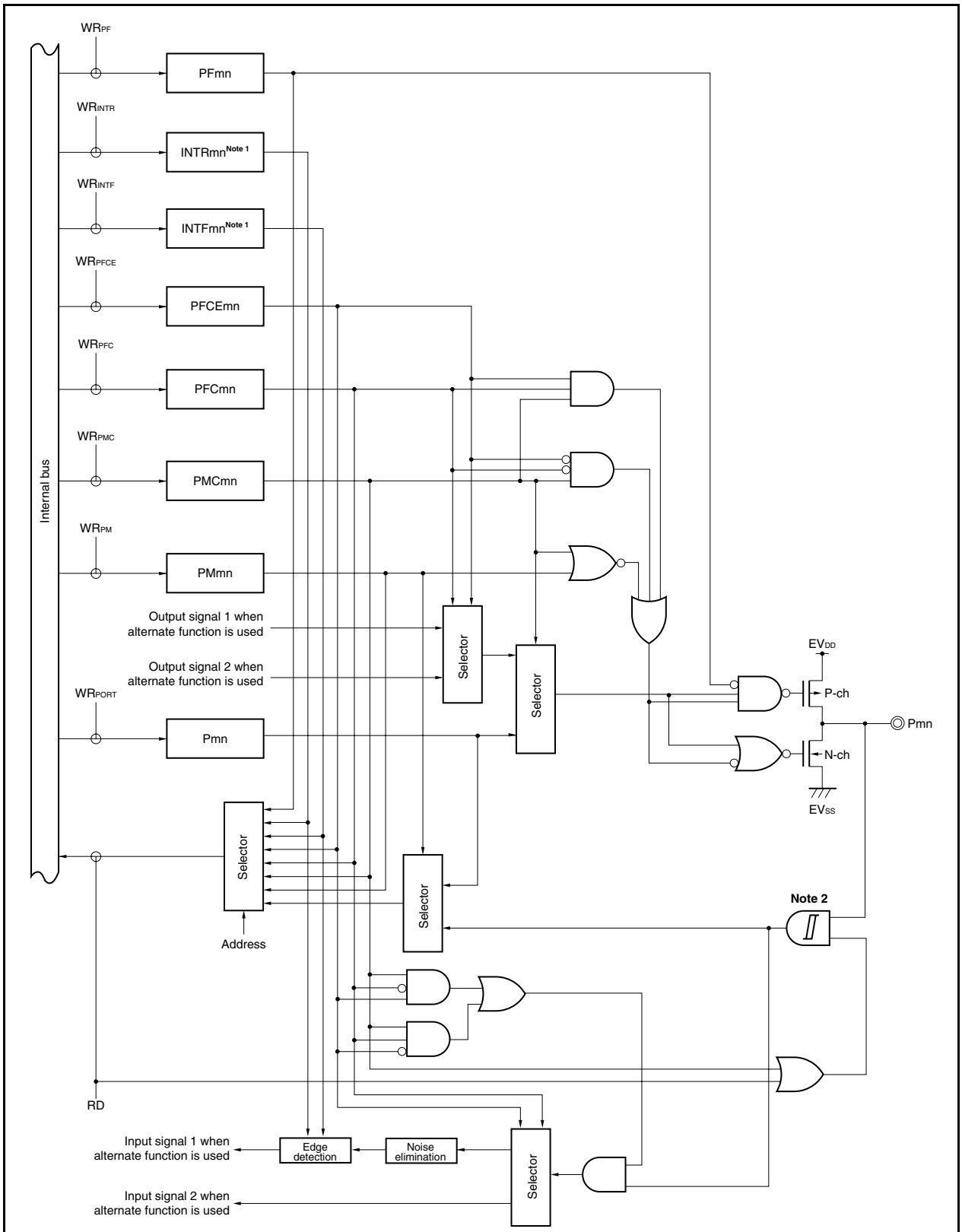
Note Hysteresis characteristics are not available in port mode.

Figure 4-28. Block Diagram of Type U-14



Note Hysteresis characteristics are not available in port mode.

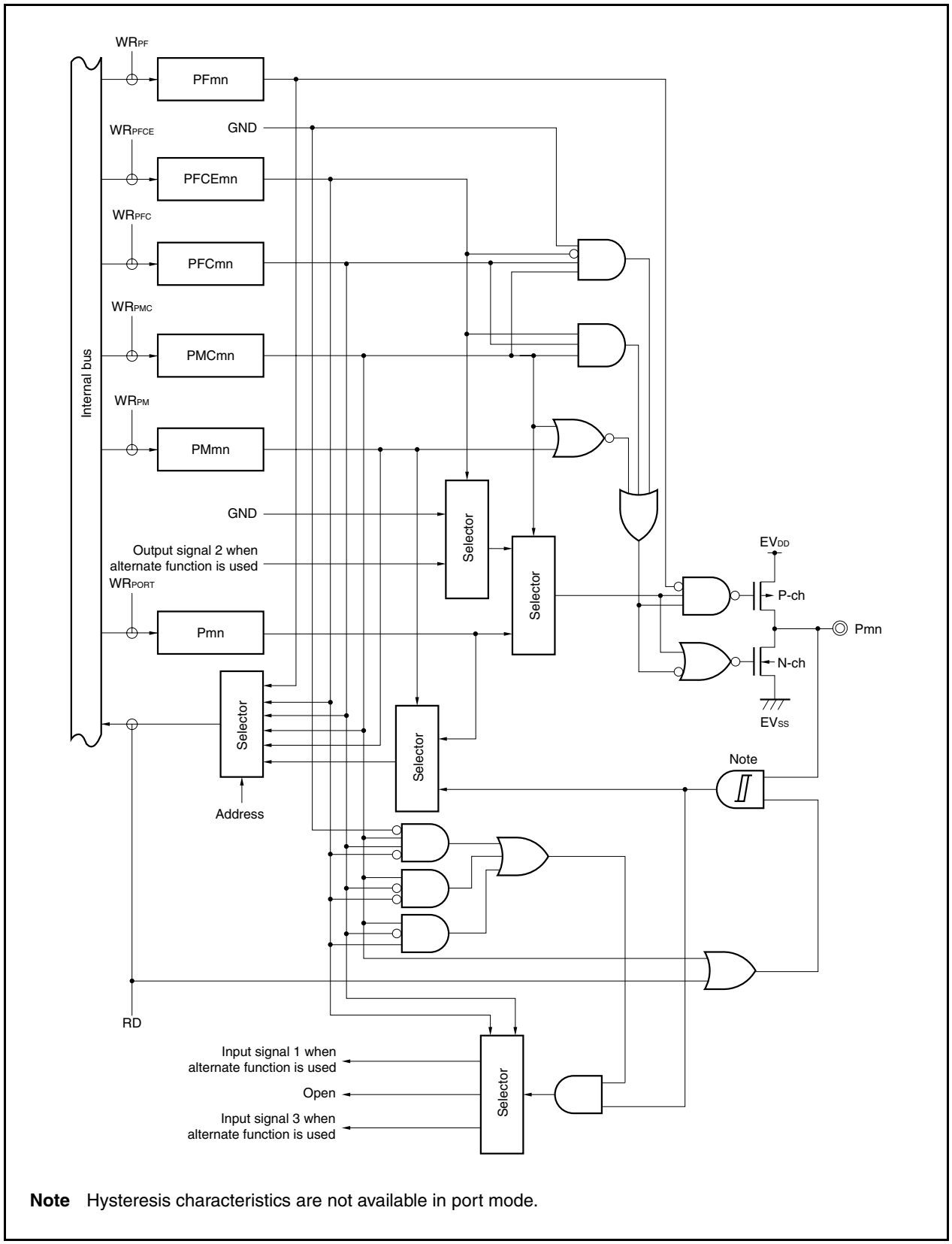
<R> **Figure 4-29. Block Diagram of Type U-15**



Notes 1. See **19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7)**.

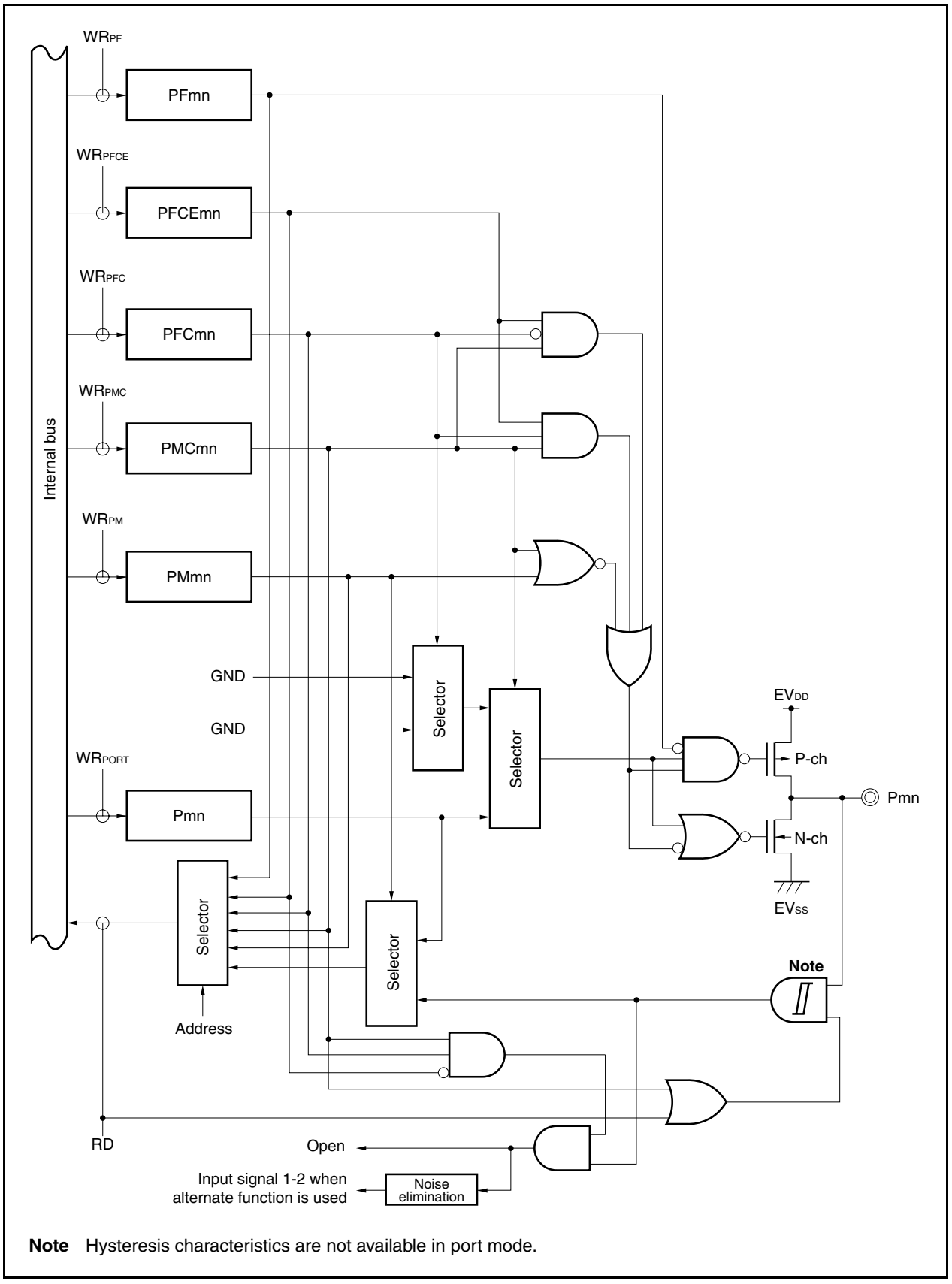
2. Hysteresis characteristics are not available in port mode.

Figure 4-30. Block Diagram of Type U-16



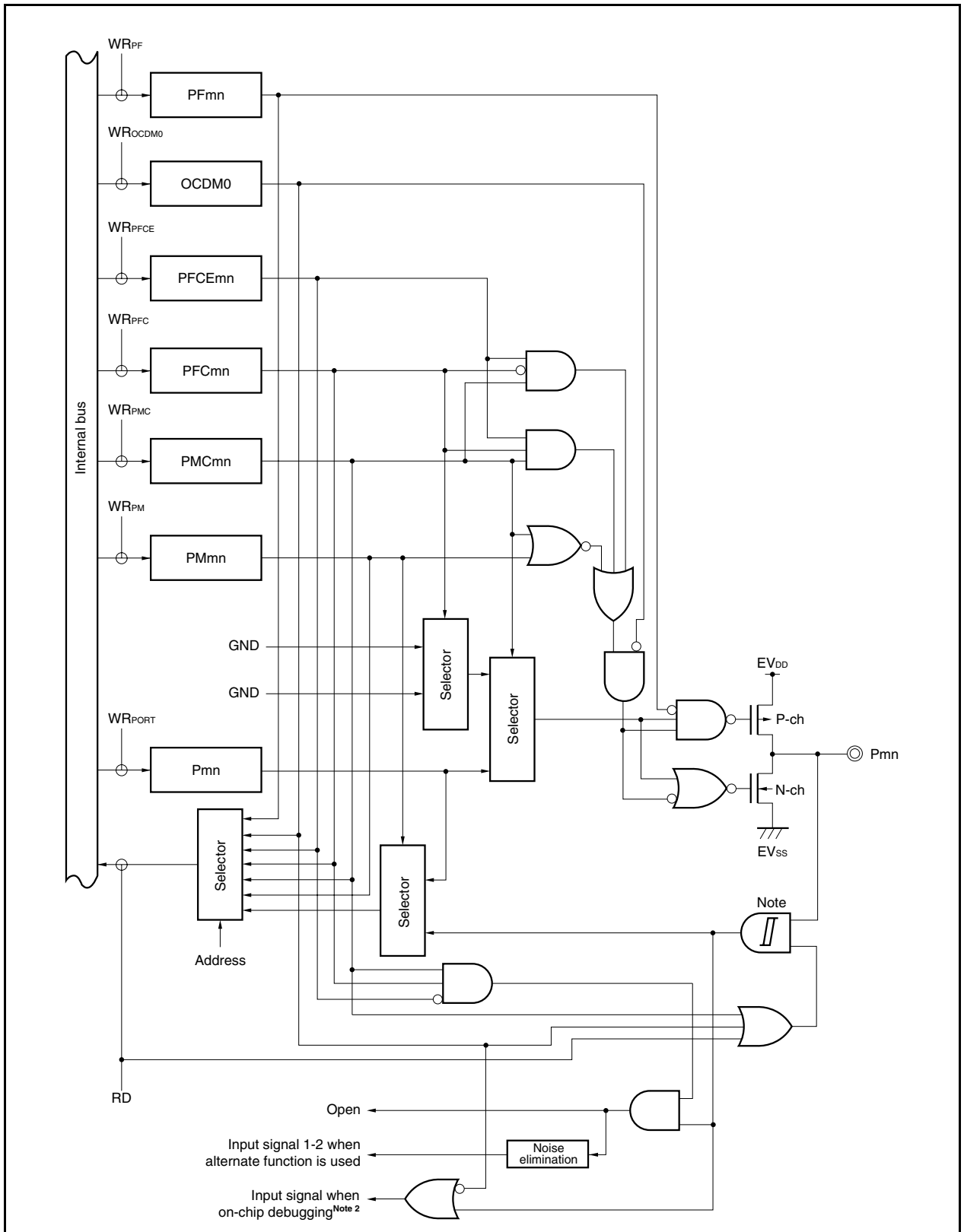
Note Hysteresis characteristics are not available in port mode.

Figure 4-31. Block Diagram of Type U-20



Note Hysteresis characteristics are not available in port mode.

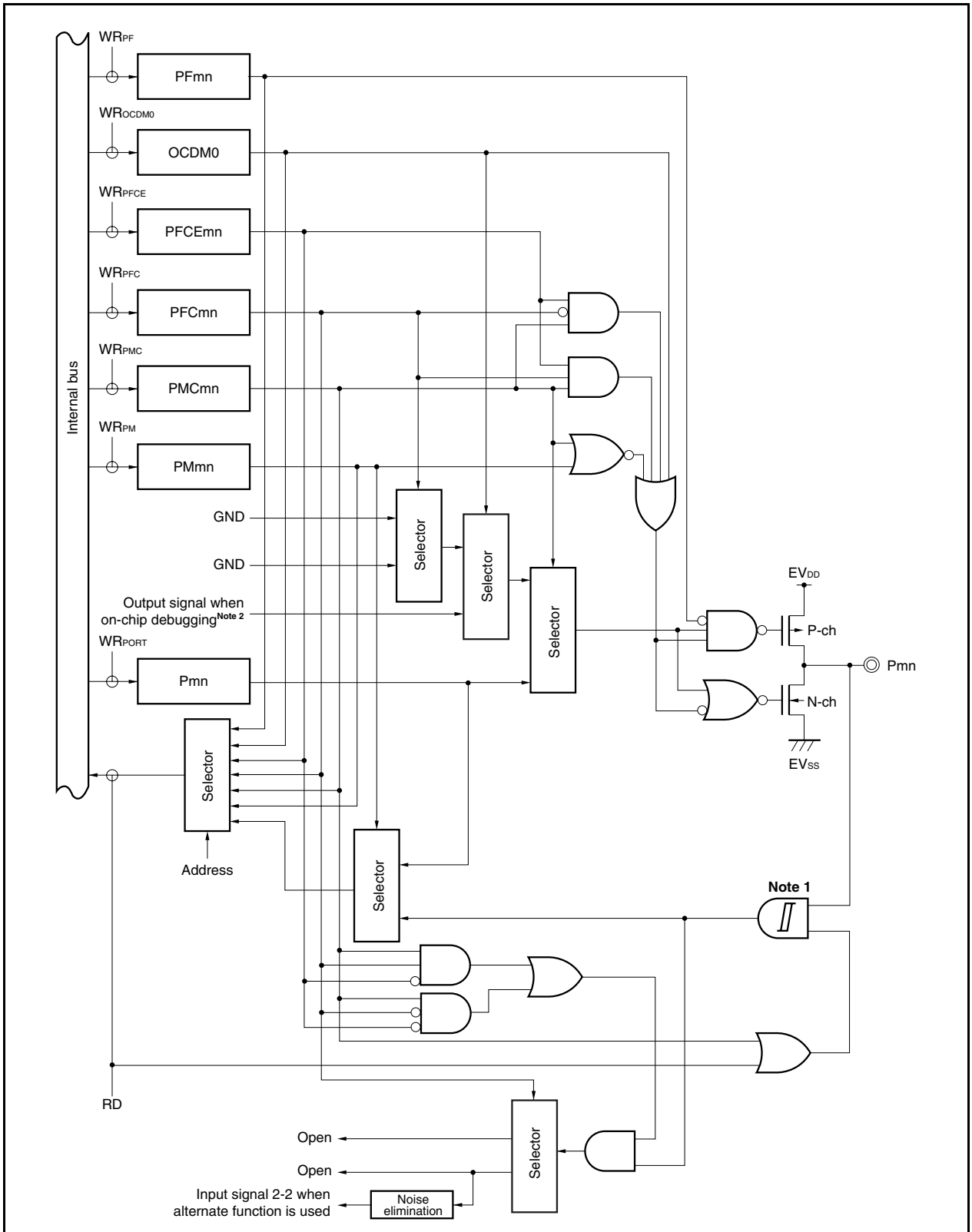
Figure 4-32. Block Diagram of Type U-21



Notes 1. Hysteresis characteristics are not available in port mode.

2. The on-chip debug function is not available in the V850ES/SG1.

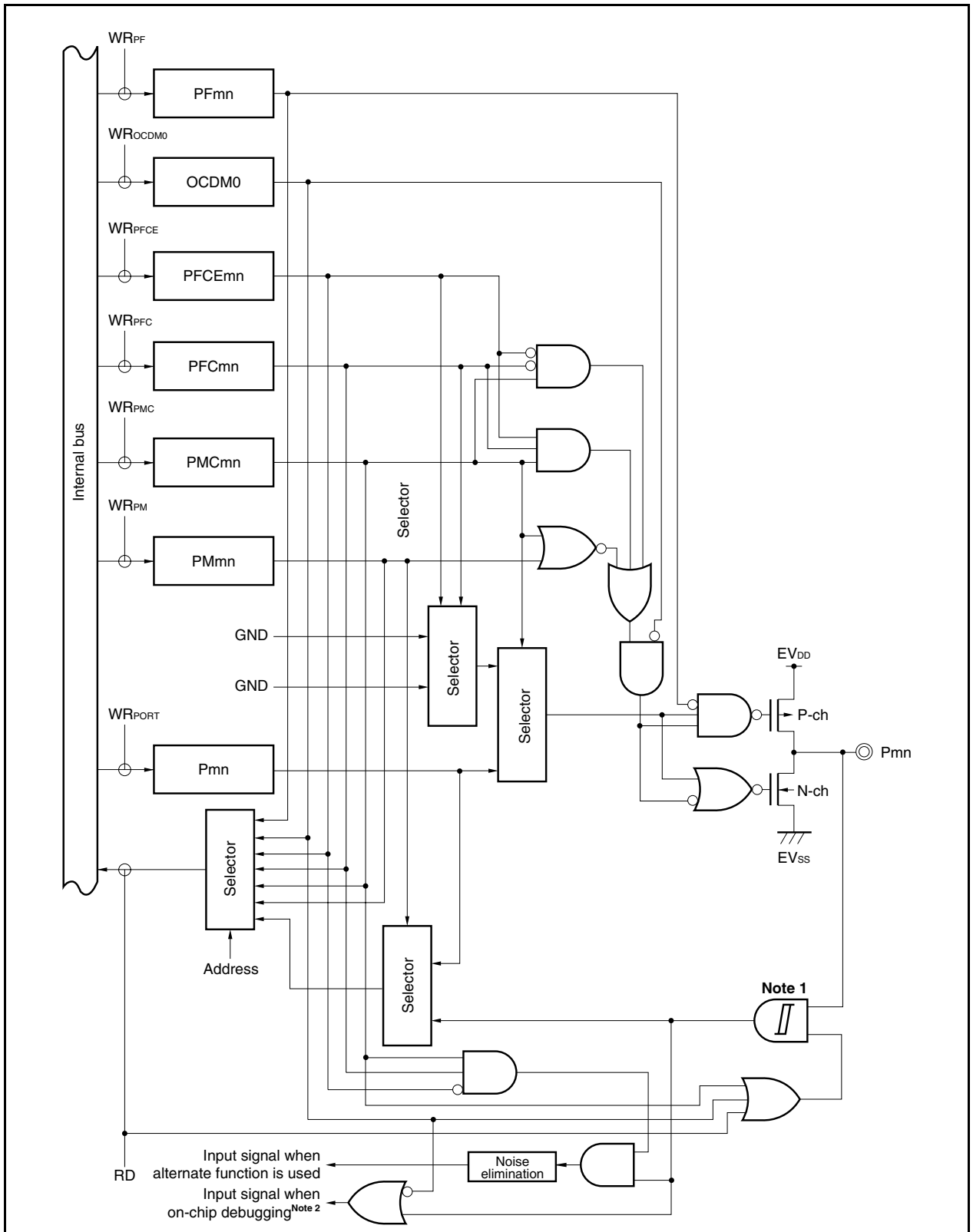
Figure 4-33. Block Diagram of Type U-22



Notes 1. Hysteresis characteristics are not available in port mode.

2. The on-chip debug function is not available in the V850ES/SG1.

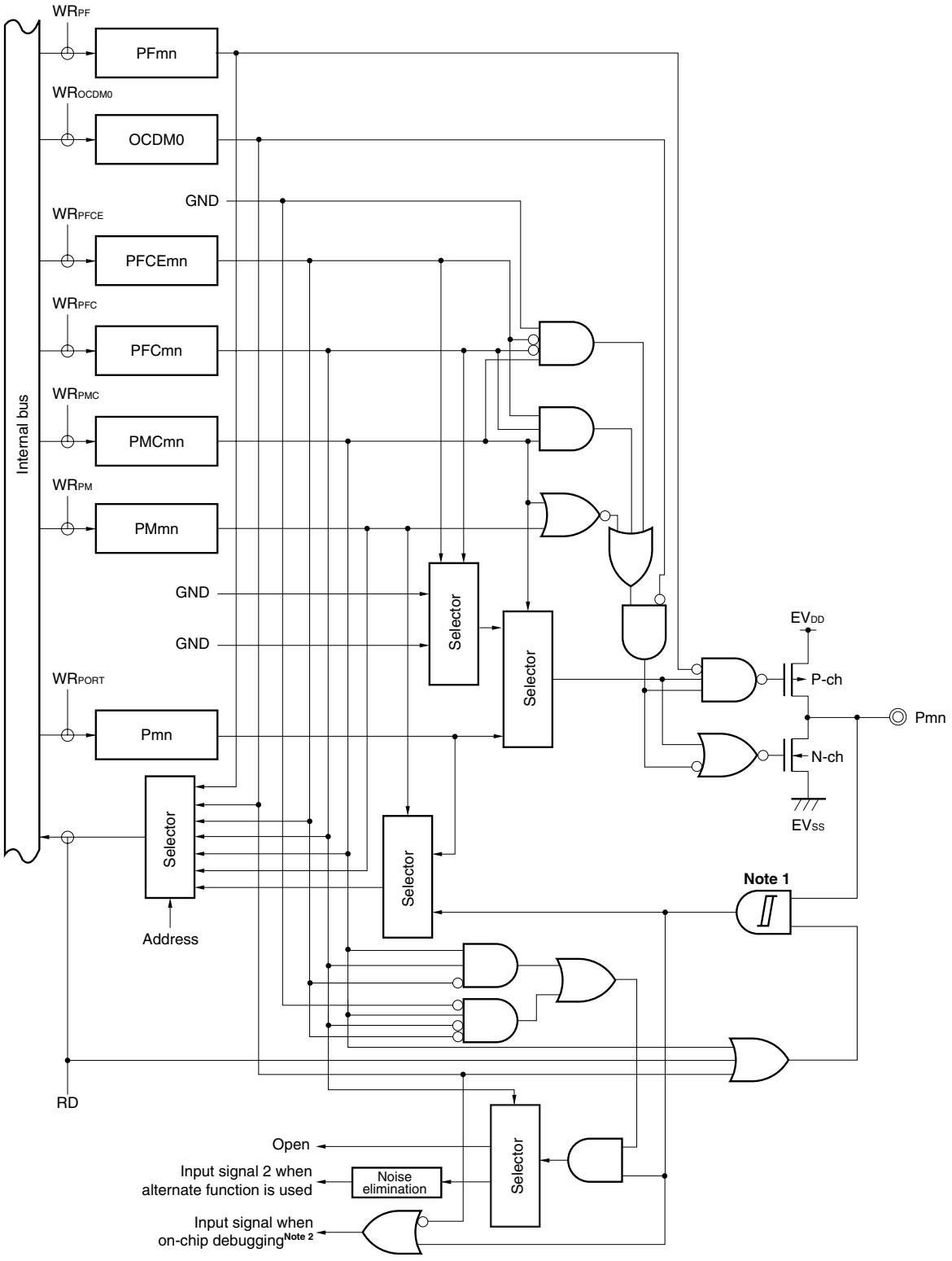
Figure 4-34. Block Diagram of Type U-23



Notes 1. Hysteresis characteristics are not available in port mode.

2. The on-chip debug function is not available in the V850ES/SG1.

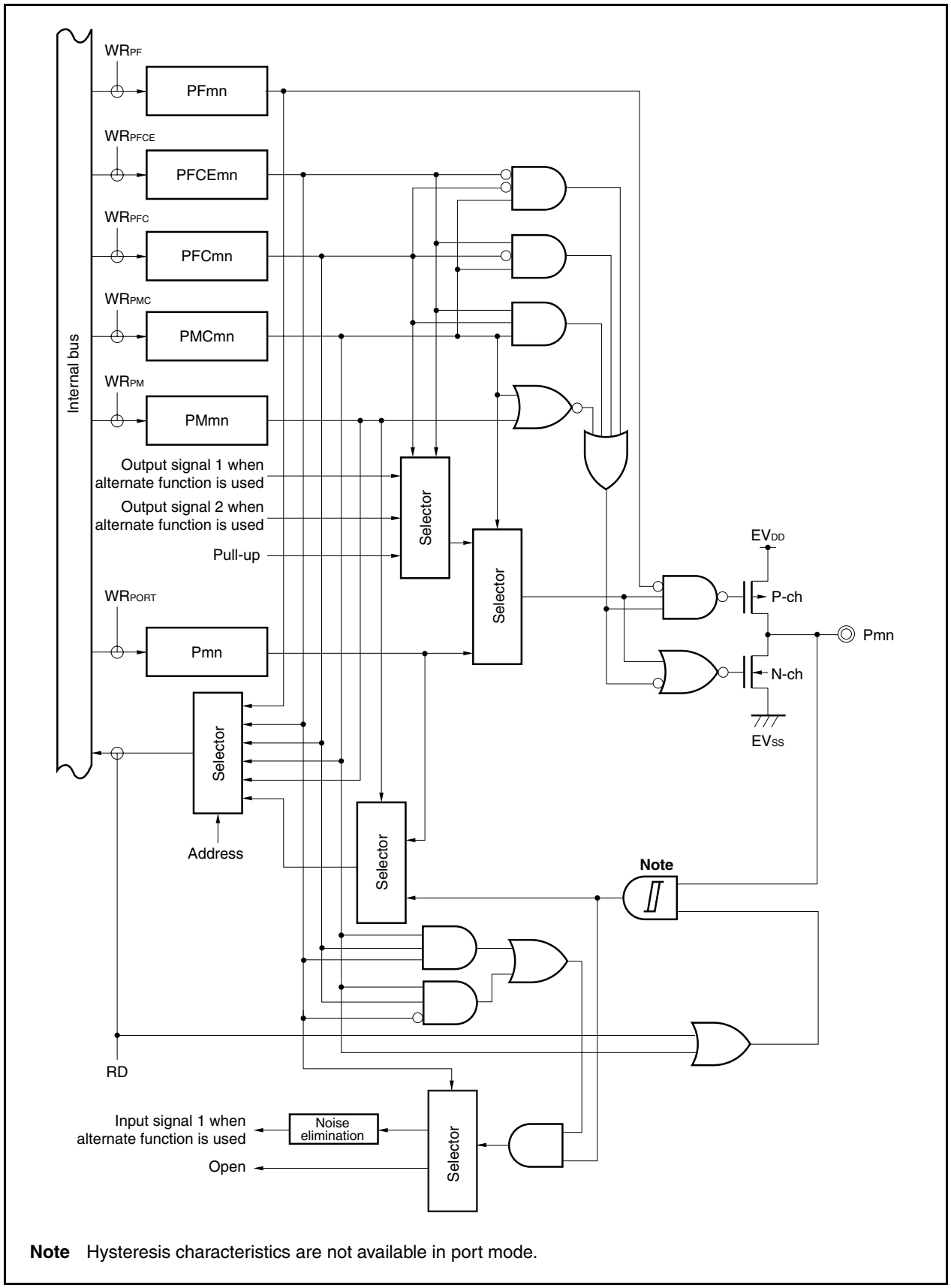
Figure 4-35. Block Diagram of Type U-24



Notes 1. Hysteresis characteristics are not available in port mode.

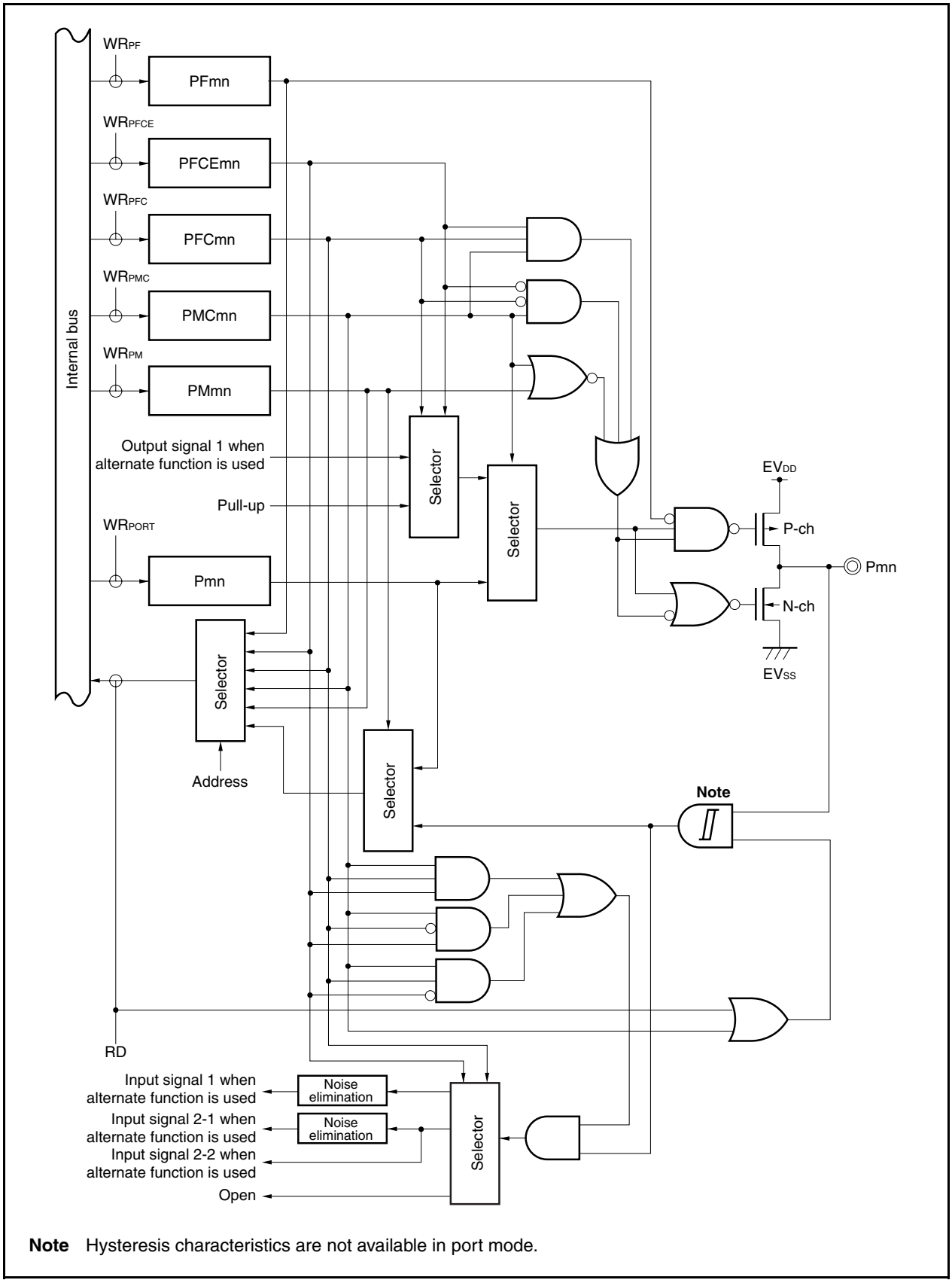
2. The on-chip debug function is not available in the V850ES/SG1.

Figure 4-36. Block Diagram of Type U-25



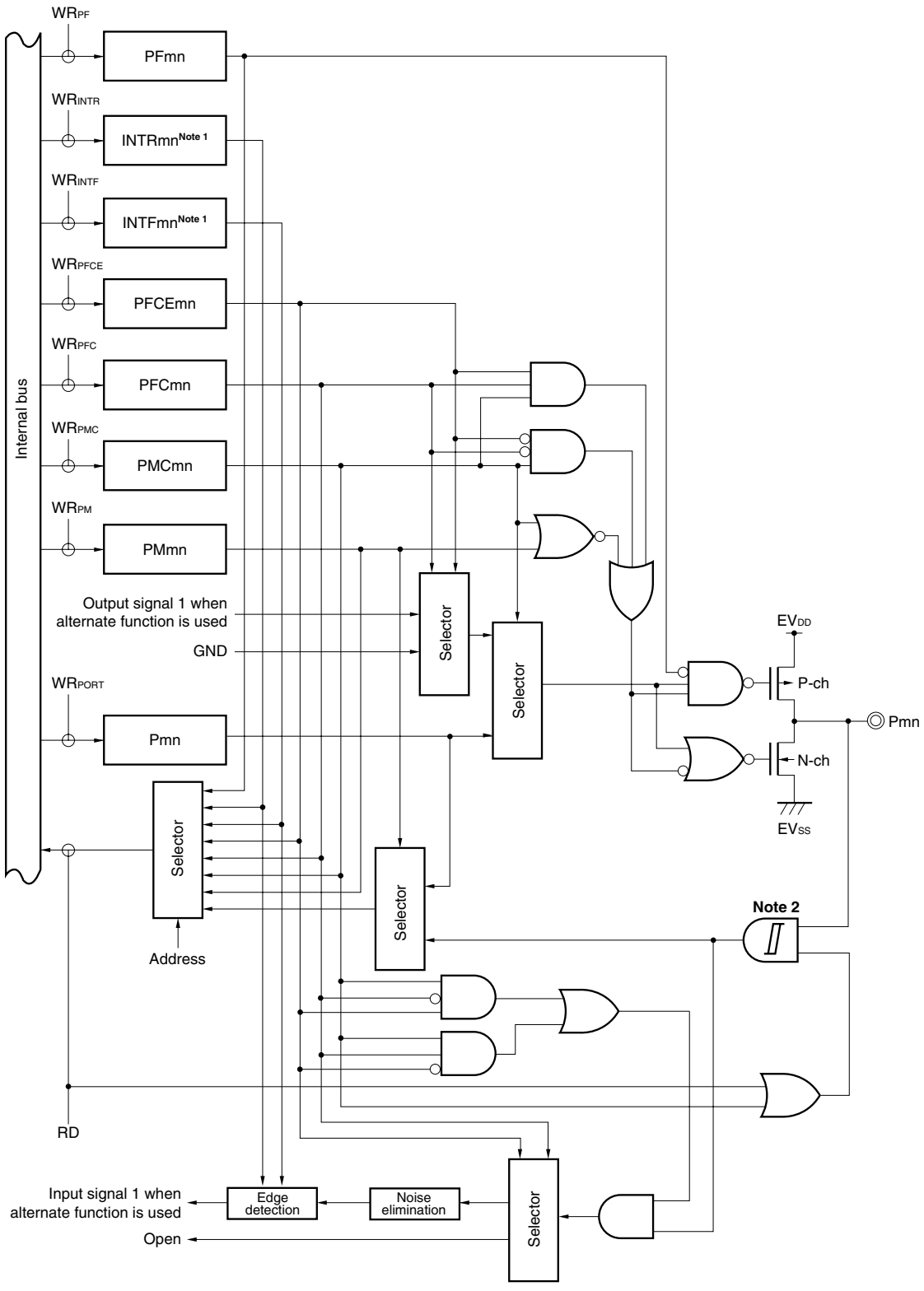
Note Hysteresis characteristics are not available in port mode.

Figure 4-37. Block Diagram of Type U-26



Note Hysteresis characteristics are not available in port mode.

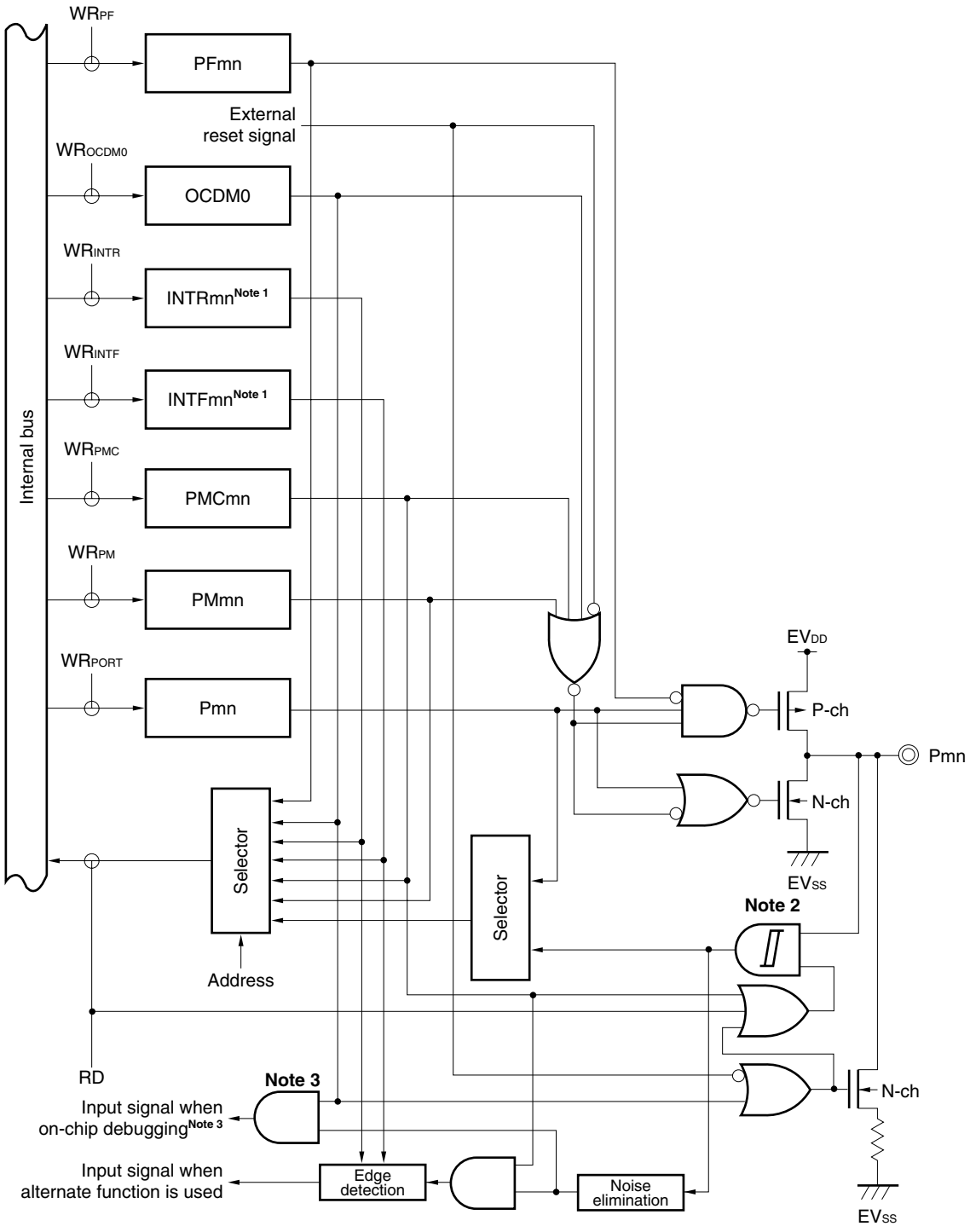
Figure 4-38. Block Diagram of Type U-27



Notes 1. See **19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7)**.

2. Hysteresis characteristics are not available in port mode.

Figure 4-39. Block Diagram of Type AA-1



Notes 1. See **19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7)**.

2. Hysteresis characteristics are not available in port mode.

<R> **3.** The on-chip debug function is not available in the V850ES/SG1.

4.5 Port Register Settings When Alternate Function Is Used

Table 4-15 shows the port register settings when each port is used for an alternate function. When using a port pin as an alternate-function pin, refer to the description of each pin.

<R>**Table 4-15. Using Port Pin as Alternate-Function Pin (1/6)**

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P02	NMI	Input	P02 = Setting not required	PM02 = Setting not required	PMC02 = 1	–	–	
P03	INTP0	Input	P03 = Setting not required	PM03 = Setting not required	PMC03 = 1	–	PFC03 = 0	
	ADTRG	Input	P03 = Setting not required	PM03 = Setting not required	PMC03 = 1	–	PFC03 = 1	
P04	INTP1	Input	P04 = Setting not required	PM04 = Setting not required	PMC04 = 1	–	–	
P05Note 1	INTP2	Input	P05 = Setting not required	PM05 = Setting not required	PMC05 = 1	–	–	
P06	INTP3	Input	P06 = Setting not required	PM06 = Setting not required	PMC06 = 1	–	–	
P10	ANO0Note 2	Output	P10 = Setting not required	PM10 = 1	–	–	–	
P11	ANO1Note 2	Output	P11 = Setting not required	PM11 = 1	–	–	–	
P30	TXDA0	Output	P30 = Setting not required	PM30 = Setting not required	PMC30 = 1	–	PFC30 = 0	
P31	RXDA0	Input	P31 = Setting not required	PM31 = Setting not required	PMC31 = 1	–	**Note 3**, PFC31 = 0	
	INTP7	Input	P31 = Setting not required	PM31 = Setting not required	PMC31 = 1	–	**Note 3**, PFC31 = 0	
P32	ASCKA0	Input	P32 = Setting not required	PM32 = Setting not required	PMC32 = 1	PFCE32 = 0	PFC32 = 0	
	TIP00	Input	P32 = Setting not required	PM32 = Setting not required	PMC32 = 1	PFCE32 = 1	PFC32 = 0	
	TOP00	Output	P32 = Setting not required	PM32 = Setting not required	PMC32 = 1	PFCE32 = 1	PFC32 = 1	
P33	TIP01	Input	P33 = Setting not required	PM33 = Setting not required	PMC33 = 1	–	PFC33 = 0	
	TOP01	Output	P33 = Setting not required	PM33 = Setting not required	PMC33 = 1	–	PFC33 = 1	
P34	TIP10	Input	P34 = Setting not required	PM34 = Setting not required	PMC34 = 1	–	PFC34 = 0	
	TOP10	Output	P34 = Setting not required	PM34 = Setting not required	PMC34 = 1	–	PFC34 = 1	

Notes 1. This pin functions alternately as the on-chip debug function pin in the flash memory version (V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)).

2. μPD703253Y only

3. The INTP7 pin and RXDA0 pin are alternate-function pins. When using the pin as the RXDA0 pin, disable edge detection for the alternate-function INTP7 pin (clear the INTF3.INTF31 bit and INTR3.INTR31 bit to 0). When using the pin as the INTP7 pin, stop the UARTA0 reception operation (clear the UA0CTL0.UA0RXE bit to 0).

Caution In the μPD703253Y, when using one of the P10 and P11 pins as an I/O port and the other as a D/A output pin (ANO0, ANO1), do so in an application where the port I/O level does not change during D/A output.

<R> **Table 4-15. Using Port Pin as Alternate-Function Pin (2/6)**

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P35	TIP11	Input	P35 = Setting not required	PM35 = Setting not required	PMC35 = 1	–	PFC35 = 0	
	TOP11	Output	P35 = Setting not required	PM35 = Setting not required	PMC35 = 1	–	PFC35 = 1	
P36	CTXD0Note 1	Output	P36 = Setting not required	PM36 = Setting not required	PMC36 = 1	–	PFC36 = 0	
	$\overline{\text{IETX0}}$Note 2	Output	P36 = Setting not required	PM36 = Setting not required	PMC36 = 1	–	PFC36 = 1	
P37	CRXD0Note 1	Input	P37 = Setting not required	PM37 = Setting not required	PMC37 = 1	–	PFC37 = 0	
	$\overline{\text{IERX0}}$Note 2	Input	P37 = Setting not required	PM37 = Setting not required	PMC37 = 1	–	PFC37 = 1	
P38	SDA00	I/O	P38 = Setting not required	PM38 = Setting not required	PMC38 = 1	–	PFC38 = 1	PF38 (PF3) = 1
P39	SCL00	I/O	P39 = Setting not required	PM39 = Setting not required	PMC39 = 1	–	PFC39 = 1	PF39 (PF3) = 1
P40	SIB0	Input	P40 = Setting not required	PM40 = Setting not required	PMC40 = 1	–	PFC40 = 0	
	SDA01	I/O	P40 = Setting not required	PM40 = Setting not required	PMC40 = 1	–	PFC40 = 1	PF40 (PF4) = 1
P41	SOB0	Output	P41 = Setting not required	PM41 = Setting not required	PMC41 = 1	–	PFC41 = 0	
	SCL01	I/O	P41 = Setting not required	PM41 = Setting not required	PMC41 = 1	–	PFC41 = 1	PF41 (PF4) = 1
P42	$\overline{\text{SCKB0}}$	I/O	P42 = Setting not required	PM42 = Setting not required	PMC42 = 1	–	–	
P50	KR0	Input	P50 = Setting not required	PM50 = Setting not required	PMC50 = 1	PFCE50 = 0	PFC50 = 1	
P51	KR1	Input	P51 = Setting not required	PM51 = Setting not required	PMC51 = 1	PFCE51 = 0	PFC51 = 1	
P52Note 3	KR2	Input	P52 = Setting not required	PM52 = Setting not required	PMC52 = 1	PFCE52 = 0	PFC52 = 1	
P53Note 3	KR3	Input	P53 = Setting not required	PM53 = Setting not required	PMC53 = 1	PFCE53 = 0	PFC53 = 1	
P54Note 3	KR4	Input	P54 = Setting not required	PM54 = Setting not required	PMC54 = 1	PFCE54 = 0	PFC54 = 1	
P55Note 3	KR5	Input	P55 = Setting not required	PM55 = Setting not required	PMC55 = 1	PFCE55 = 1	PFC55 = 1	

Notes 1. μPD703253Y only

2. μPD703252Y only

3. This pin functions alternately as the on-chip debug function pin in the flash memory version (V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)).

<R> **Table 4-15. Using Port Pin as Alternate-Function Pin (3/6)**

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P70	ANI0	Input	P70 = Setting not required	PM70 = 1	–	–	–	
P71	ANI1	Input	P71 = Setting not required	PM71 = 1	–	–	–	
P72	ANI2	Input	P72 = Setting not required	PM72 = 1	–	–	–	
P73	ANI3	Input	P73 = Setting not required	PM73 = 1	–	–	–	
P74	ANI4	Input	P74 = Setting not required	PM74 = 1	–	–	–	
P75	ANI5	Input	P75 = Setting not required	PM75 = 1	–	–	–	
P76	ANI6	Input	P76 = Setting not required	PM76 = 1	–	–	–	
P77	ANI7	Input	P77 = Setting not required	PM77 = 1	–	–	–	
P78	ANI8	Input	P78 = Setting not required	PM78 = 1	–	–	–	
P79	ANI9	Input	P79 = Setting not required	PM79 = 1	–	–	–	
P710	ANI10	Input	P710 = Setting not required	PM710 = 1	–	–	–	
P711	ANI11	Input	P711 = Setting not required	PM711 = 1	–	–	–	
P90	A0	Output	P90 = Setting not required	PM90 = Setting not required	PMC90 = 1	PFCE90 = 0	PFC90 = 0	**Note 1**
	KR6	Input	P90 = Setting not required	PM90 = Setting not required	PMC90 = 1	PFCE90 = 0	PFC90 = 1	
	TXDA1	Output	P90 = Setting not required	PM90 = Setting not required	PMC90 = 1	PFCE90 = 1	PFC90 = 0	
P91	A1	Output	P91 = Setting not required	PM91 = Setting not required	PMC91 = 1	PFCE91 = 0	PFC91 = 0	**Note 1**
	KR7	Input	P91 = Setting not required	PM91 = Setting not required	PMC91 = 1	PFCE91 = 0	PFC91 = 1	
	RXDA1/KR7Note 2	Input	P91 = Setting not required	PM91 = Setting not required	PMC91 = 1	PFCE91 = 1	PFC91 = 0	

Notes 1. Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once.

2. The RXDA1 and KR7 pins must not be used at the same time. When using the RXDA1 pin, do not use the KR7 pin. When using the KR7 pin, do not use the RXDA1 pin (it is recommended to set the PFC91 bit to 1 and clear the PFCE91 bit to 0).

<R> **Table 4-15. Using Port Pin as Alternate-Function Pin (4/6)**

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P92	A2	Output	P92 = Setting not required	PM92 = Setting not required	PMC92 = 1	PFCE92 = 0	PFC92 = 0	**Note**
	TIP41	Input	P92 = Setting not required	PM92 = Setting not required	PMC92 = 1	PFCE92 = 0	PFC92 = 1	
	TOP41	Output	P92 = Setting not required	PM92 = Setting not required	PMC92 = 1	PFCE92 = 1	PFC92 = 0	
P93	A3	Output	P93 = Setting not required	PM93 = Setting not required	PMC93 = 1	PFCE93 = 0	PFC93 = 0	**Note**
	TIP40	Input	P93 = Setting not required	PM93 = Setting not required	PMC93 = 1	PFCE93 = 0	PFC93 = 1	
	TOP40	Output	P93 = Setting not required	PM93 = Setting not required	PMC93 = 1	PFCE93 = 1	PFC93 = 0	
P94	A4	Output	P94 = Setting not required	PM94 = Setting not required	PMC94 = 1	PFCE94 = 0	PFC94 = 0	**Note**
	TIP31	Input	P94 = Setting not required	PM94 = Setting not required	PMC94 = 1	PFCE94 = 0	PFC94 = 1	
	TOP31	Output	P94 = Setting not required	PM94 = Setting not required	PMC94 = 1	PFCE94 = 1	PFC94 = 0	
P95	A5	Output	P95 = Setting not required	PM95 = Setting not required	PMC95 = 1	PFCE95 = 0	PFC95 = 0	**Note**
	TIP30	Input	P95 = Setting not required	PM95 = Setting not required	PMC95 = 1	PFCE95 = 0	PFC95 = 1	
	TOP30	Output	P95 = Setting not required	PM95 = Setting not required	PMC95 = 1	PFCE95 = 1	PFC95 = 0	
P96	A6	Output	P96 = Setting not required	PM96 = Setting not required	PMC96 = 1	PFCE96 = 0	PFC96 = 0	**Note**
	TIP21	Input	P96 = Setting not required	PM96 = Setting not required	PMC96 = 1	PFCE96 = 1	PFC96 = 0	
	TOP21	Output	P96 = Setting not required	PM96 = Setting not required	PMC96 = 1	PFCE96 = 1	PFC96 = 1	
P97	A7	Output	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFCE97 = 0	PFC97 = 0	**Note**
	SIB1	Input	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFCE97 = 0	PFC97 = 1	
	TIP20	Input	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFCE97 = 1	PFC97 = 0	
	TOP20	Output	P97 = Setting not required	PM97 = Setting not required	PMC97 = 1	PFCE97 = 1	PFC97 = 1	
P98	A8	Output	P98 = Setting not required	PM98 = Setting not required	PMC98 = 1	–	PFC98 = 0	**Note**
	SOB1	Output	P98 = Setting not required	PM98 = Setting not required	PMC98 = 1	–	PFC98 = 1	
P99	A9	Output	P99 = Setting not required	PM99 = Setting not required	PMC99 = 1	–	PFC99 = 0	**Note**
	$\overline{\text{SCKB1}}$	I/O	P99 = Setting not required	PM99 = Setting not required	PMC99 = 1	–	PFC99 = 1	

Note Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once.

<R> **Table 4-15. Using Port Pin as Alternate-Function Pin (5/6)**

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
P910	A10	Output	P910 = Setting not required	PM910 = Setting not required	PMC910 = 1	–	PFC910 = 0	**Note 1**
	SIB3	Input	P910 = Setting not required	PM910 = Setting not required	PMC910 = 1	–	PFC910 = 1	
P911	A11	Output	P911 = Setting not required	PM911 = Setting not required	PMC911 = 1	–	PFC911 = 0	**Note 1**
	SOB3	Output	P911 = Setting not required	PM911 = Setting not required	PMC911 = 1	–	PFC911 = 1	
P912	A12	Output	P912 = Setting not required	PM912 = Setting not required	PMC912 = 1	–	PFC912 = 0	**Note 1**
	$\overline{\text{SCKB3}}$	I/O	P912 = Setting not required	PM912 = Setting not required	PMC912 = 1	–	PFC912 = 1	
P913	A13	Output	P913 = Setting not required	PM913 = Setting not required	PMC913 = 1	–	PFC913 = 0	**Note 1**
	INTP4	Input	P913 = Setting not required	PM913 = Setting not required	PMC913 = 1	–	PFC913 = 1	
P914	A14	Output	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFCE914 = 0	PFC914 = 0	**Note 1**
	INTP5	Input	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFCE914 = 0	PFC914 = 1	
	TIP51 **Note 2**	Input	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFCE914 = 1	PFC914 = 0	
	TOP51 **Note 2**	Output	P914 = Setting not required	PM914 = Setting not required	PMC914 = 1	PFCE914 = 1	PFC914 = 1	
P915	A15	Output	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFCE915 = 0	PFC915 = 0	**Note 1**
	INTP6	Input	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFCE915 = 0	PFC915 = 1	
	TIP50 **Note 2**	Input	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFCE915 = 1	PFC915 = 0	
	TOP50 **Note 2**	Output	P915 = Setting not required	PM915 = Setting not required	PMC915 = 1	PFCE915 = 1	PFC915 = 1	
PCM0	$\overline{\text{WAIT}}$	Input	PCM0 = Setting not required	PMCM0 = Setting not required	PMCCM0 = 1	–	–	
PCM1	CLKOUT	Output	PCM1 = Setting not required	PMCM1 = Setting not required	PMCCM1 = 1	–	–	
PCM2	$\overline{\text{HLDAK}}$	Output	PCM2 = Setting not required	PMCM2 = Setting not required	PMCCM2 = 1	–	–	
PCM3	$\overline{\text{HLDRQ}}$	Input	PCM3 = Setting not required	PMCM3 = Setting not required	PMCCM3 = 1	–	–	
PCT0	$\overline{\text{WR0}}$	Output	PCT0 = Setting not required	PMCT0 = Setting not required	PMCCT0 = 1	–	–	
PCT1	$\overline{\text{WR1}}$	Output	PCT1 = Setting not required	PMCT1 = Setting not required	PMCCT1 = 1	–	–	
PCT4	$\overline{\text{RD}}$	Output	PCT4 = Setting not required	PMCT4 = Setting not required	PMCCT4 = 1	–	–	
PCT6	ASTB	Output	PCT6 = Setting not required	PMCT6 = Setting not required	PMCCT6 = 1	–	–	

Notes 1. Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once.

2. μPD703253Y only

Table 4-15. Using Port Pin as Alternate-Function Pin (6/6)

Pin Name	Alternate Function		Pnx Bit of Pn Register	PMnx Bit of PMn Register	PMCnx Bit of PMCn Register	PFCEnx Bit of PFCEn Register	PFCnx Bit of PFCn Register	Other Bits (Registers)
	Name	I/O						
PDH0	A16	Output	PDH0 = Setting not required	PMDH0 = Setting not required	PMCDH0 = 1	–	–	
PDH1	A17	Output	PDH1 = Setting not required	PMDH1 = Setting not required	PMCDH1 = 1	–	–	
PDH2	A18	Output	PDH2 = Setting not required	PMDH2 = Setting not required	PMCDH2 = 1	–	–	
PDH3	A19	Output	PDH3 = Setting not required	PMDH3 = Setting not required	PMCDH3 = 1	–	–	
PDH4	A20	Output	PDH4 = Setting not required	PMDH4 = Setting not required	PMCDH4 = 1	–	–	
PDH5	A21	Output	PDH5 = Setting not required	PMDH5 = Setting not required	PMCDH5 = 1	–	–	
PDL0	AD0	I/O	PDL0 = Setting not required	PMDL0 = Setting not required	PMCDL0 = 1	–	–	
PDL1	AD1	I/O	PDL1 = Setting not required	PMDL1 = Setting not required	PMCDL1 = 1	–	–	
PDL2	AD2	I/O	PDL2 = Setting not required	PMDL2 = Setting not required	PMCDL2 = 1	–	–	
PDL3	AD3	I/O	PDL3 = Setting not required	PMDL3 = Setting not required	PMCDL3 = 1	–	–	
PDL4	AD4	I/O	PDL4 = Setting not required	PMDL4 = Setting not required	PMCDL4 = 1	–	–	
PDL5	AD5	I/O	PDL5 = Setting not required	PMDL5 = Setting not required	PMCDL5 = 1	–	–	
PDL6	AD6	I/O	PDL6 = Setting not required	PMDL6 = Setting not required	PMCDL6 = 1	–	–	
PDL7	AD7	I/O	PDL7 = Setting not required	PMDL7 = Setting not required	PMCDL7 = 1	–	–	
PDL8	AD8	I/O	PDL8 = Setting not required	PMDL8 = Setting not required	PMCDL8 = 1	–	–	
PDL9	AD9	I/O	PDL9 = Setting not required	PMDL9 = Setting not required	PMCDL9 = 1	–	–	
PDL10	AD10	I/O	PDL10 = Setting not required	PMDL10 = Setting not required	PMCDL10 = 1	–	–	
PDL11	AD11	I/O	PDL11 = Setting not required	PMDL11 = Setting not required	PMCDL11 = 1	–	–	
PDL12	AD12	I/O	PDL12 = Setting not required	PMDL12 = Setting not required	PMCDL12 = 1	–	–	
PDL13	AD13	I/O	PDL13 = Setting not required	PMDL13 = Setting not required	PMCDL13 = 1	–	–	
PDL14	AD14	I/O	PDL14 = Setting not required	PMDL14 = Setting not required	PMCDL14 = 1	–	–	
PDL15	AD15	I/O	PDL15 = Setting not required	PMDL15 = Setting not required	PMCDL15 = 1	–	–	

4.6 Cautions

4.6.1 Cautions on setting port pins

(1) In the V850ES/SG1, the general-purpose port function and several peripheral function I/O pin share a pin. To switch between the general-purpose port (port mode) and the peripheral function I/O pin (alternate-function mode), set by the PMCn register. In regards to this register setting sequence, note with caution the following.

(a) Cautions on switching from port mode to alternate-function mode
To switch from the port mode to alternate-function mode in the following order.

<1> Set the PFn register[Note]: N-ch open-drain setting
<2> Set the PFCn and PFCEn registers: Alternate-function selection
<3> Set the corresponding bit of the PMCn register to 1: Switch to alternate-function mode

If the PMCn register is set first, note with caution that, at that moment or depending on the change of the pin states in accordance with the setting of the PFn, PFCn, and PFCEn registers, unexpected operations may occur.
A concrete example is shown as Example below.

Note N-ch open-drain output pin only

Caution Regardless of the port mode/alternate-function mode, the Pn register is read and written as follows.
- **Pn register read: Read the port output latch value (when PMn.PMnm bit = 0), or read the pin states (PMn.PMnm bit = 1).**
- **Pn register write: Write to the port output latch**

[Example] SCL01 pin setting example
The SCL01 pin is used alternately with the P41/SOB0 pin. Select the valid pin functions with the PMC4, PFC4, and PF4 registers.

PMC41 Bit	PFC41 Bit	PF41 Bit	Valid Pin Functions
0	don't care	1	P41 (in output port mode, N-ch open-drain output)
1	0	1	SOB0 output (N-ch open-drain output)
	1	1	SCL01 I/O (N-ch open-drain output)

The order of setting in which malfunction may occur on switching from the P41 pin to the SCL01 pin are shown below.

Setting Order	Setting Contents	Pin States	Pin Level
<1>	Initial value (PMC41 bit = 0, PFC41 bit = 0, PF41 bit = 0)	Port mode (input)	Hi-Z
<2>	PMC41 bit ← 1	SOB0 output	Low level (high level depending on the CSIB0 setting)
<3>	PFC41 bit ← 1	SCL01 I/O	High level (CMOS output)
<4>	PF41 bit ← 1	SCL01 I/O	Hi-Z (N-ch open-drain output)

In <2>, I^2C communication may be affected since the alternate-function SOB0 output is output to the pin. In the CMOS output period of <2> or <3>, unnecessary current may be generated.

(b) Cautions on alternate-function mode (input)

The input signal to the alternate-function block is low level when the PMCn.PMCnm bit is 0 due to the AND output of the PMCn register set value and the pin level. Thus, depending on the port setting and alternate-function operation enable timing, unexpected operations may occur. Therefore, switch between the port mode and alternate-function mode in the following sequence.

- To switch from port mode to alternate-function mode (input)
  Set the pins to the alternate-function mode using the PMCn register and then enable the alternate-function operation.
- To switch from alternate-function mode (input) to port mode
  Stop the alternate-function operation and then switch the pins to the port mode.

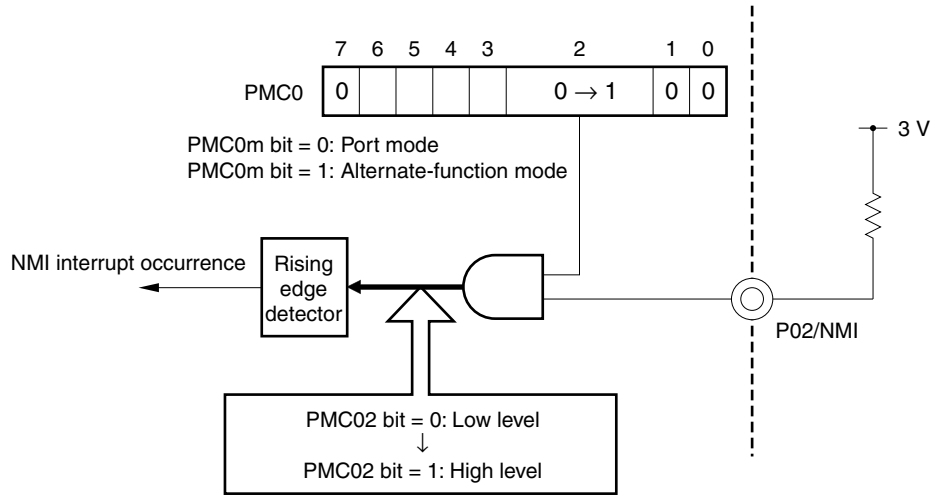
The concrete examples are shown as Example 1 and Example 2.

[Example 1] Switch from general-purpose port (P02) to external interrupt pin (NMI)

When the P02/NMI pin is pulled up as shown in Figure 4-40 and the rising edge is specified in the NMI pin edge detection setting, even though high level is input continuously to the NMI pin during switching from the P02 pin to the an NMI pin (PMC02 bit = 0 → 1), this is detected as a rising edge as if the low level changed to high level, and an NMI interrupt occurs.

To avoid it, set the NMI pin's valid edge after switching from the P02 pin to the NMI pin.

Figure 4-40. Example of Switching from P02 to NMI (Incorrect)



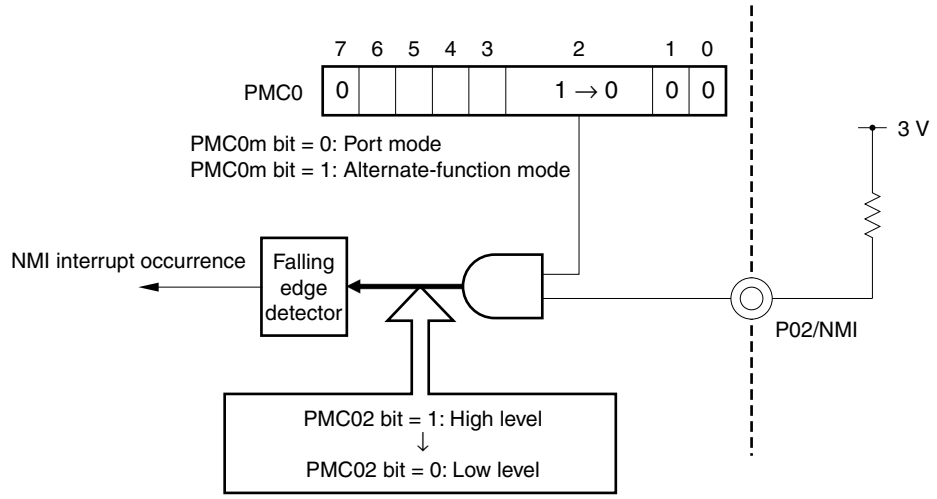
Remark m = 2 to 6

[Example 2] Switch from external pin (NMI) to general-purpose port (P02)

When the P02/NMI pin is pulled up as shown in Figure 4-41 and the falling edge is specified in the NMI pin edge detection setting, even though high level is input continuously to the NMI pin at switching from the NMI pin to the P02 pin (PMC02 bit = 1 → 0), this is detected as falling edge as if high level changed to low level, and NMI interrupt occurs.

To avoid this, set the NMI pin edge detection as “No edge detected” before switching to the P02 pin.

Figure 4-41. Example of Switching from NMI to P02 (Incorrect)



Remark m = 2 to 6

(2) In port mode, the PFn.PFnm bit is valid only in the output mode (PMn.PMnm bit = 0). In the input mode (PMnm bit = 1), the value of the PFnm bit is not reflected in the buffer.

4.6.2 Cautions on bit manipulation instruction for port n register (Pn)

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

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

<Example> When P90 pin is an output port, P91 to P97 pins are input ports (all pin statuses are high level), and the value of the port latch is 00H, if the output of P90 pin is changed from low level to high level via a bit manipulation instruction, the value of the port latch is FFH.

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

A bit manipulation instruction is executed in the following order in the V850ES/SG1.

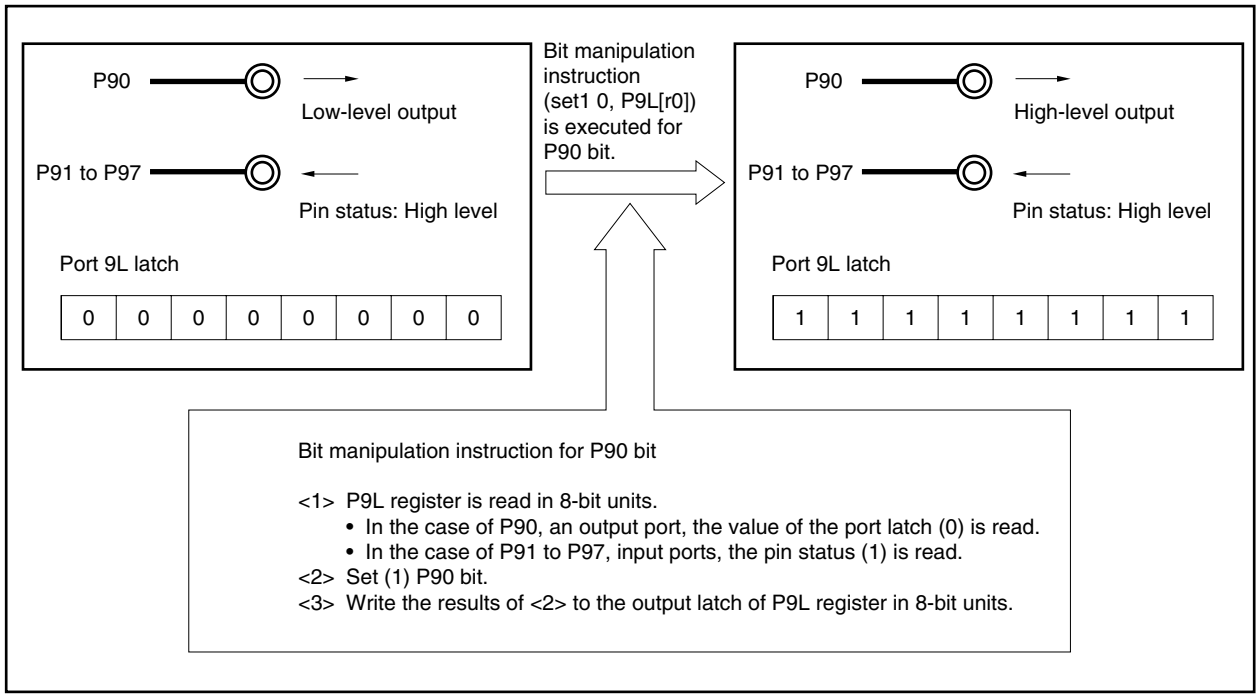
<1> The Pn register is read in 8-bit units.
<2> The targeted one bit is manipulated.
<3> The Pn register is written in 8-bit units.

In step <1>, the value of the output latch (0) of P90 pin, which is an output port, is read, while the pin statuses of P91 to P97 pins, which are input ports, are read. If the pin statuses of P91 to P97 pins are high level at this time, the read value is FEH.

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

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

Figure 4-42. Bit Manipulation Instruction (P90 Pin)



4.6.3 Cautions on on-chip debug pins

The V850ES/SG1 does not have an on-chip debug function. However, when using the V850ES/SG2 <R> (μPD70F3261Y, 70F3271Y, 70F3281Y) as the flash memory version for developing software, the following actions are required.

- Clear the OCDM0 bit of the OCDM register (special register) (0)

At this time, fix the P05/INTP2 pin to low level from when reset by the $\overline{\text{RESET}}$ pin is released until the above action is taken.

If a high level is input to the P05/INTP2 pin before the above action is taken, it may cause a malfunction (CPU deadlock). Handle the P05/INTP2 pin with the utmost care.

For details, refer to the **V850ES/SG2, V850ES/SG2-H Hardware User's Manual (U16541E)**.

(1) On-chip debug mode register (OCDM)

The OCDM register is used to select the normal operation mode or on-chip debug mode. This register is a special register and can be written only in a combination of specific sequences (see **3.4.8 Special registers**). This register can be read or written in 8-bit or 1-bit units.

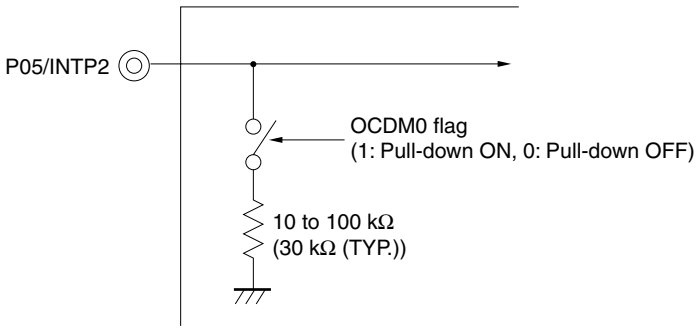
After reset: 01H[Note] R/W Address: FFFFF9FCH

	7	6	5	4	3	2	1	<0>
OCDM	0	0	0	0	0	0	0	OCDM0

OCDM0	Operation mode
0	Disconnects the on-chip pull-down resistor of the P05/INTP2 pin.
1	Connects the on-chip pull-down resistor of the P05/INTP2 pin.

Note $\overline{\text{RESET}}$ input sets this register to 01H. After reset by the WDT2RES signal or clock monitor (CLM), however, the value of the OCDM register is retained.

Caution The V850ES/SG1 does not have an on-chip debug function but the P05/INTP2 pin has an on-chip pull-down resistor. With the V850ES/SG1, therefore, the on-chip pull-down resistor must be disconnected by clearing the OCDM0 bit to 0.



4.6.4 Cautions on P10, P11, and P53 pins when power is turned on

<R> The following pins may momentarily output an undefined level upon power application.

- P10/ANO0 pin
- P11/ANO1 pin
- P53/KR3 pin

4.6.5 Hysteresis characteristics

In port mode, the following port pins do not have hysteresis characteristics.

P02 to P06
<R> P31 to P35 and P37 to P39
P40 to P42
P50 to P55
<R> P90 to P97, P99, P910, and P912 to P915

<R> 4.6.6 Cautions on separate bus mode

Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once. If even one of the A0 to A15 pins is not used in the separate bus mode, port 9 pins can be used as port pins or other alternate-function pins.

CHAPTER 5 BUS CONTROL FUNCTION

The V850ES/SG1 is provided with an external bus interface function by which external memories such as ROM and RAM, and I/O can be connected.

5.1 Features

- ○ Output is selectable from a multiplexed bus with a minimum of 3 bus cycles and a separate bus with a minimum of 2 bus cycles.
- ○ 8-bit/16-bit data bus selectable
- ○ Wait function
  - Programmable wait function of up to 7 states
  - External wait function using $\overline{\text{WAIT}}$ pin
- ○ Idle state function
- ○ Bus hold function
- <R> ○ Up to 4 MB of physical memory connectable
- ○ The bus can be controlled at a voltage that is different from the operating voltage when $BV_{DD} \leq EV_{DD} = V_{DD}$. However, in separate bus mode or when the A20 and A21 pins are used, set $BV_{DD} = EV_{DD} = V_{DD}$.

5.2 Bus Control Pins

The pins used to connect an external device are listed in the table below.

Table 5-1. Bus Control Pins (Multiplexed Bus)

Bus Control Pin	Alternate-Function Pin	I/O	Function
AD0 to AD15	PDL0 to PDL15	I/O	Address/data bus
A16 to A21	PDH0 to PDH5	Output	Address bus
$\overline{\text{WAIT}}$	PCM0	Input	External wait control
CLKOUT	PCM1	Output	Internal system clock
$\overline{\text{WR0}}$, $\overline{\text{WR1}}$	PCT0, PCT1	Output	Write strobe signal
$\overline{\text{RD}}$	PCT4	Output	Read strobe signal
ASTB	PCT6	Output	Address strobe signal
$\overline{\text{HLDRQ}}$	PCM3	Input	Bus hold control
$\overline{\text{HLDAK}}$	PCM2	Output	

Table 5-2. External Control Pins (Separate Bus)

Bus Control Pin	Alternate-Function Pin	I/O	Function
AD0 to AD15	PDL0 to PDL15	I/O	Data bus
A0 to A15	P90 to P915	Output	Address bus
A16 to A21	PDH0 to PDH5	Output	Address bus
$\overline{\text{WAIT}}$	PCM0	Input	External wait control
CLKOUT	PCM1	Output	Internal system clock
$\overline{\text{WR0}}$, $\overline{\text{WR1}}$	PCT0, PCT1	Output	Write strobe signal
$\overline{\text{RD}}$	PCT4	Output	Read strobe signal
$\overline{\text{HLDRQ}}$	PCM3	Input	Bus hold control
$\overline{\text{HLDAK}}$	PCM2	Output	

5.2.1 Pin status when internal ROM, internal RAM, or on-chip peripheral I/O is accessed

When the internal ROM, internal RAM, or on-chip peripheral I/O are accessed, the status of each pin is as follows.

Table 5-3. Pin Statuses When Internal ROM, Internal RAM, or On-Chip Peripheral I/O Is Accessed

Separate Bus Mode		Multiplexed Bus Mode	
Address bus (A21 to A0)	Undefined	Address bus (A21 to A16)	Undefined
Data bus (AD15 to AD0)	Hi-Z	Address/data bus (AD15 to AD0)	Undefined
Control signal ($\overline{\text{RD}}$, $\overline{\text{WR0}}$, $\overline{\text{WR1}}$)	High level	Control signal ($\overline{\text{RD}}$, $\overline{\text{WR0}}$, $\overline{\text{WR1}}$, ASTB)	High level

Caution When a write access is performed to the internal ROM area, address, data, and control signals are activated in the same way as access to the external memory area.

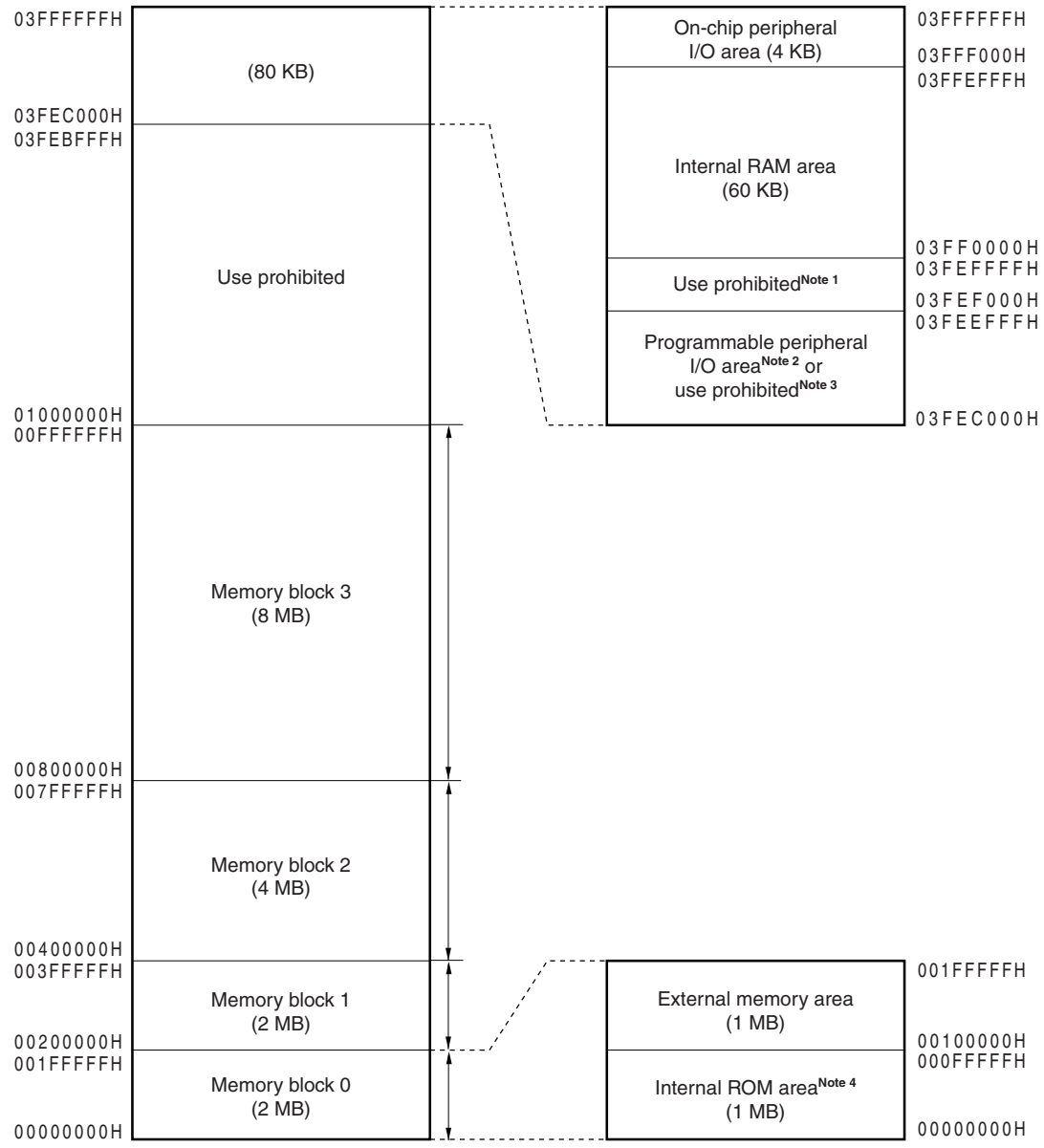
5.2.2 Pin status in each operation mode

For the pin status of the V850ES/SG1 in each operation mode, see **2.2 Pin Status**.

5.3 Memory Block Function

The 16 MB external memory space is divided into memory blocks of (lower) 2 MB, 2 MB, 4 MB, and 8 MB. The programmable wait function and bus cycle operation mode for each of these blocks can be independently controlled in one-block units.

Figure 5-1. Data Memory Map: Physical Address



Notes 1. Use of addresses 03FEF000H to 03FEFFFFH is prohibited because these addresses are in the same area as the on-chip peripheral I/O area.

<R> **2.** The programmable peripheral I/O area is only seen as images of 256 MB each in the 4 GB address space.

<R> **3.** Addresses 03FEC000H to 03FEEFFFH are allocated to addresses 03FEC000H to 03FEC5FFH of the μPD703253Y as a programmable peripheral I/O area. Use of μPD703249Y and 703252Y are prohibited.

4. This area is an external memory area in the case of a data write access.

5.4 External Bus Interface Mode Control Function

The V850ES/SG1 has the following two external bus interface modes.

- Multiplexed bus mode
- Separate bus mode

These two modes can be selected by using the EXIMC register.

(1) External bus interface mode control register (EXIMC)

The EXIMC register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFFFBEH

	7	6	5	4	3	2	1	0
EXIMC	0	0	0	0	0	0	0	SMSEL

SMSEL	Mode selection
0	Multiplexed bus mode
1	Separate bus mode

Caution Set the EXIMC register from the internal ROM or internal RAM area before making an external access.

After setting the EXIMC register, be sure to insert a NOP instruction.

5.5 Bus Access

5.5.1 Number of clocks for access

The following table shows the number of basic clocks required for accessing each resource.

Bus Cycle Type \ Area (Bus Width)	Internal ROM (32 Bits)	Internal RAM (32 Bits)	External Memory (16 Bits)
Instruction fetch (normal access)	1	1 Note 1	3 + n Note 2
Instruction fetch (branch)	2	2 Note 1	3 + n Note 2
Operand data access	3	1	3 + n Note 2

Notes 1. Increases by 1 if a conflict with a data access occurs.

2. 2 + n clocks (n: Number of wait states) when the separate bus mode is selected.

Remark Unit: Clocks/access

5.5.2 Bus size setting function

Each external memory area selected by memory block n can be set by using the BSC register. However, the bus size can be set to 8 bits and 16 bits only.

The external memory area of the V850ES/SG1 is selected by memory blocks 0 to 3.

(1) Bus size configuration register (BSC)

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

Reset sets this register to 5555H.

Caution Write to the BSC register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the BSC register are complete.

After reset: 5555H R/W Address: FFFFF066H

	15	14	13	12	11	10	9	8
BSC	0	1	0	1	0	1	0	1

7	6	5	4	3	2	1	0
0	BS30	0	BS20	0	BS10	0	BS00

Memory block 3 Memory block 2 Memory block 1 Memory block 0

BSn0	Data bus width of memory block n space (n = 0 to 3)
0	8 bits
1	16 bits

Caution Be sure to set bits 14, 12, 10, and 8 to “1”, and clear bits 15, 13, 11, 9, 7, 5, 3, and 1 to “0”.

5.5.3 Access by bus size

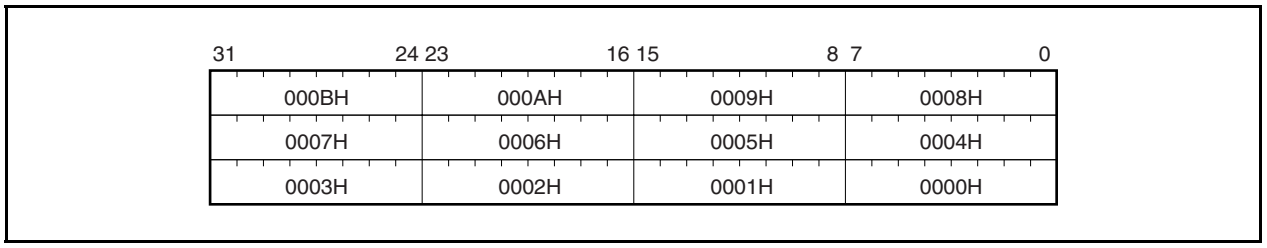
The V850ES/SG1 accesses the on-chip peripheral I/O and external memory in 8-bit, 16-bit, or 32-bit units. The bus size is as follows.

- The bus size of the on-chip peripheral I/O is fixed to 16 bits.
- The bus size of the external memory is selectable from 8 bits or 16 bits (by using the BSC register).

The operation when each of the above is accessed is described below. All data is accessed starting from the lower side.

The V850ES/SG1 supports only the little-endian format.

Figure 5-2. Little-Endian Address in Word



(1) Data space

The V850ES/SG1 has an address misalign function.

With this function, data can be placed at all addresses, regardless of the format of the data (word data or halfword data). However, if the word data or halfword data is not aligned at the boundary, a bus cycle is generated at least twice, causing the bus efficiency to drop.

(a) Halfword-length data access

A byte-length bus cycle is generated twice if the least significant bit of the address is 1.

(b) Word-length data access

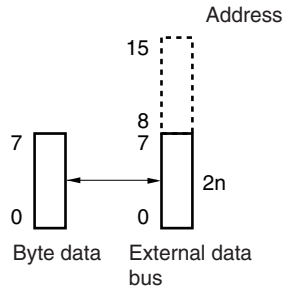
(i) A byte-length bus cycle, halfword-length bus cycle, and byte-length bus cycle are generated in that order if the least significant bit of the address is 1.

(ii) A halfword-length bus cycle is generated twice if the lower 2 bits of the address are 10.

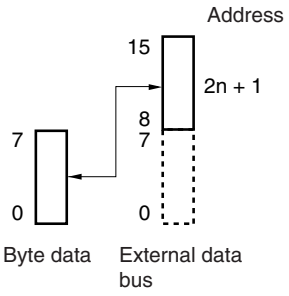
(2) Byte access (8 bits)

(a) 16-bit data bus width

<1> Access to even address (2n)

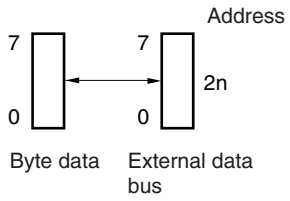


<2> Access to odd address (2n + 1)

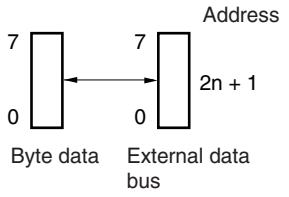


(b) 8-bit data bus width

<1> Access to even address (2n)



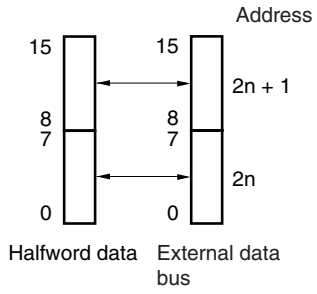
<2> Access to odd address (2n + 1)



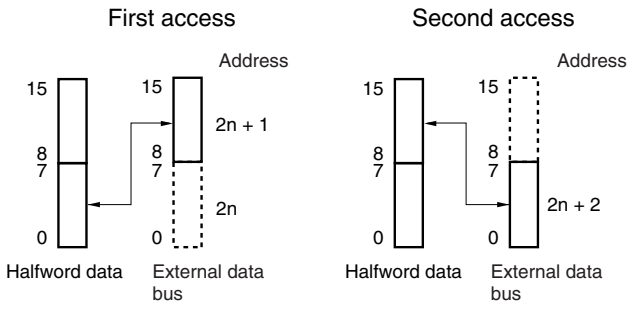
(3) Halfword access (16 bits)

(a) With 16-bit data bus width

<1> Access to even address (2n)



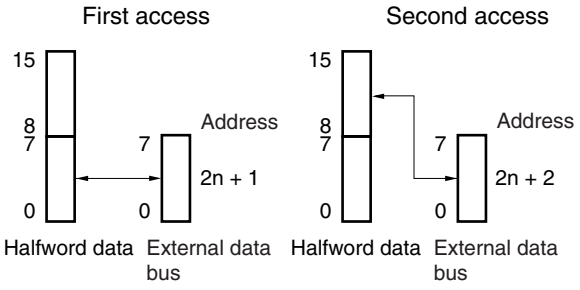
<2> Access to odd address (2n + 1)



(b) 8-bit data bus width

<1> Access to even address (2n)

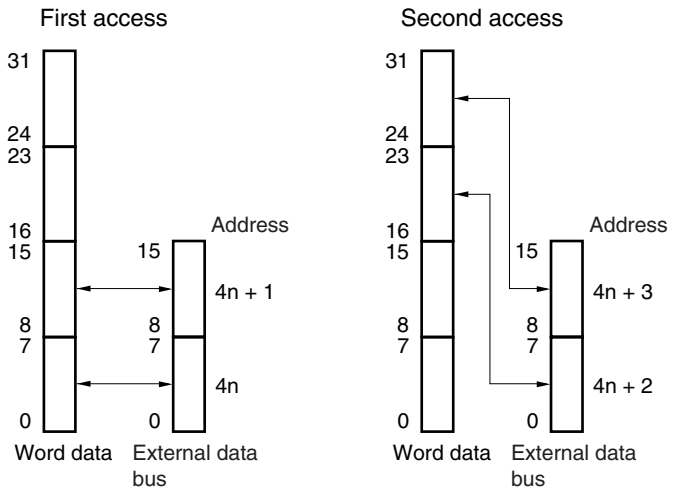
<2> Access to odd address (2n + 1)



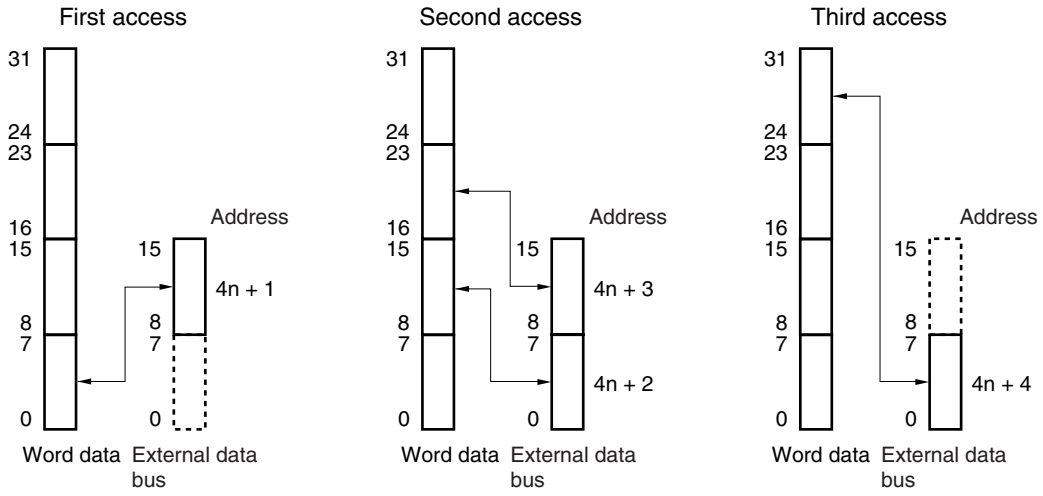
(4) Word access (32 bits)

(a) 16-bit data bus width (1/2)

<1> Access to address (4n)

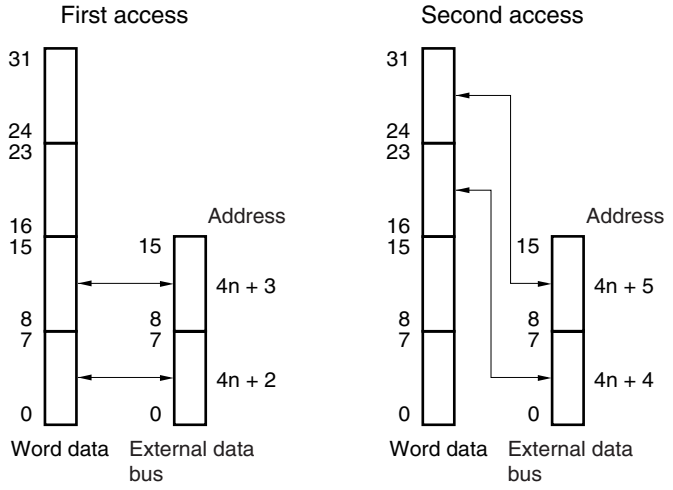


<2> Access to address (4n + 1)

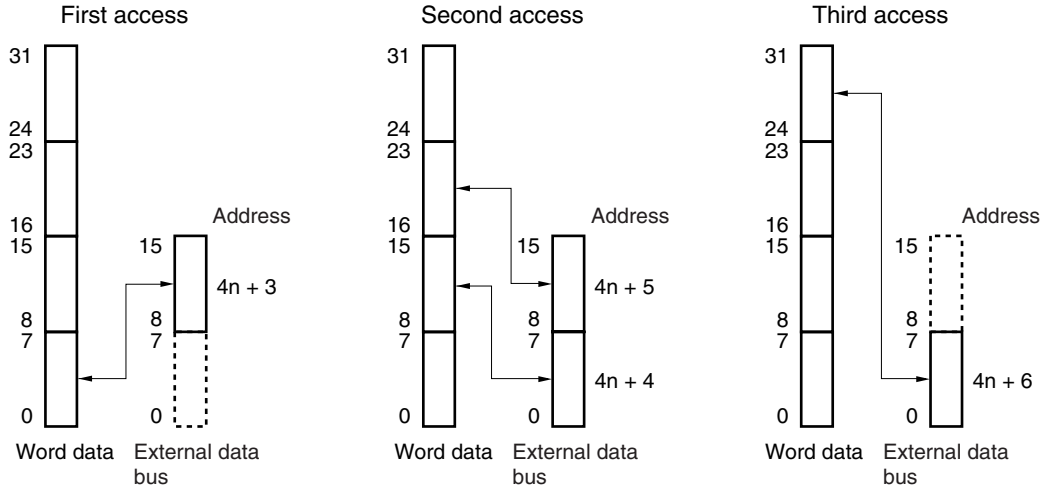


(a) 16-bit data bus width (2/2)

<3> Access to address (4n + 2)

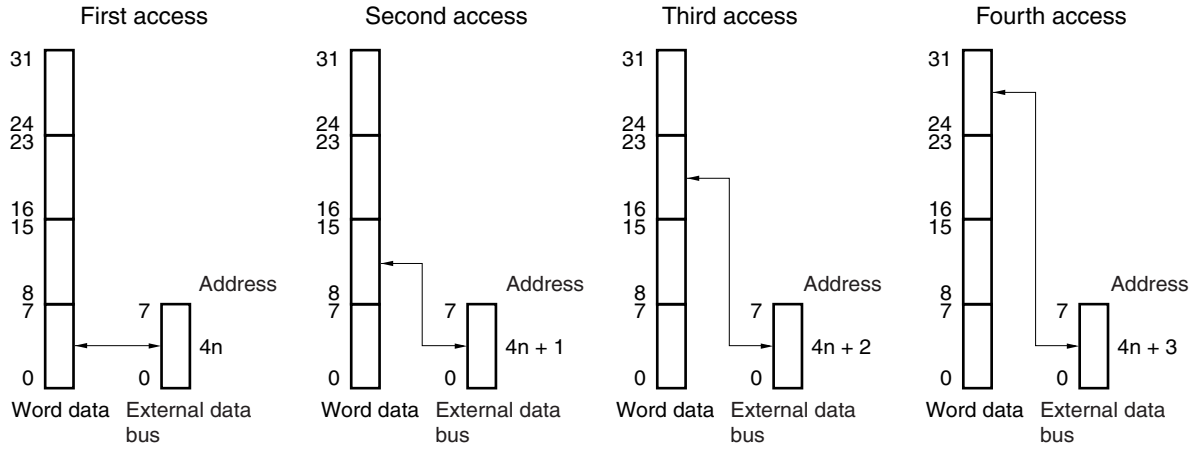


<4> Access to address (4n + 3)

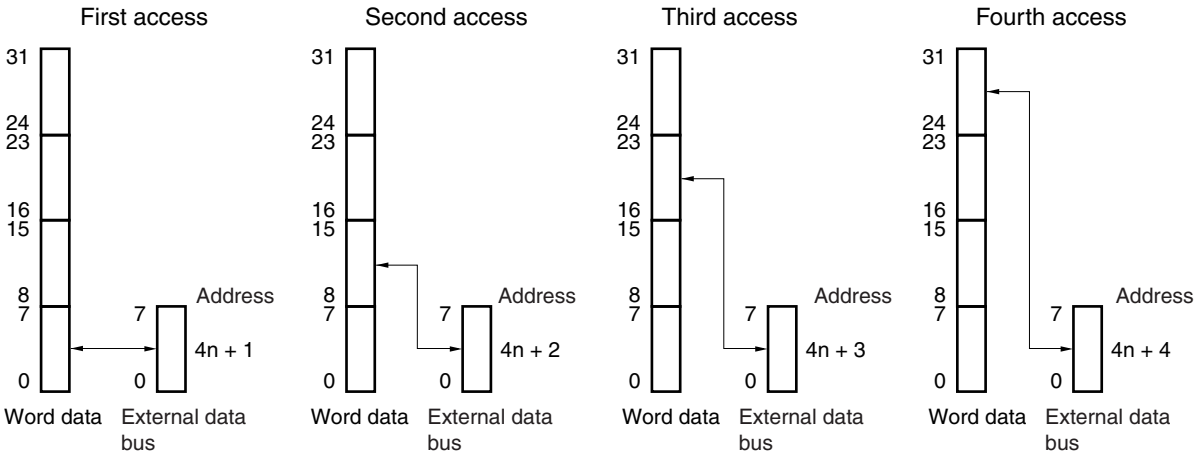


(b) 8-bit data bus width (1/2)

<1> Access to address (4n)

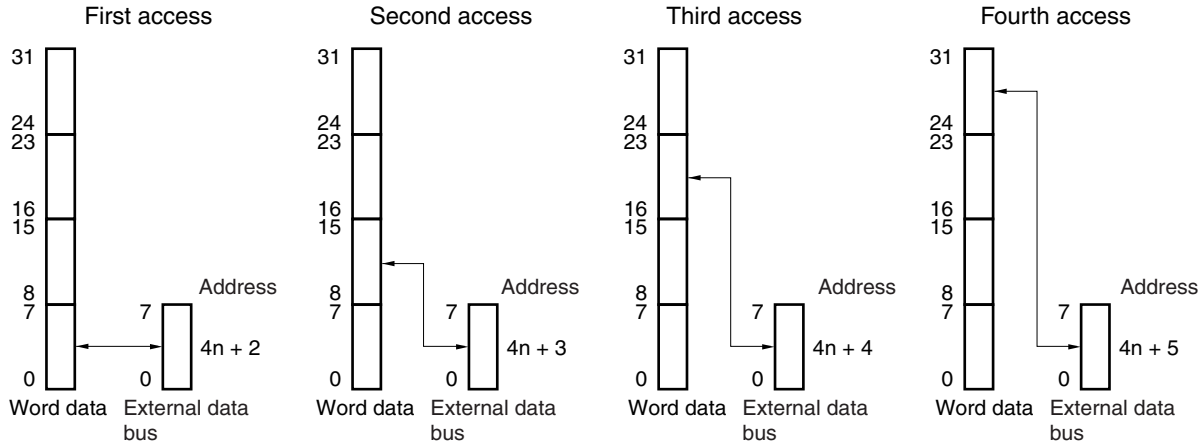


<2> Access to address (4n + 1)

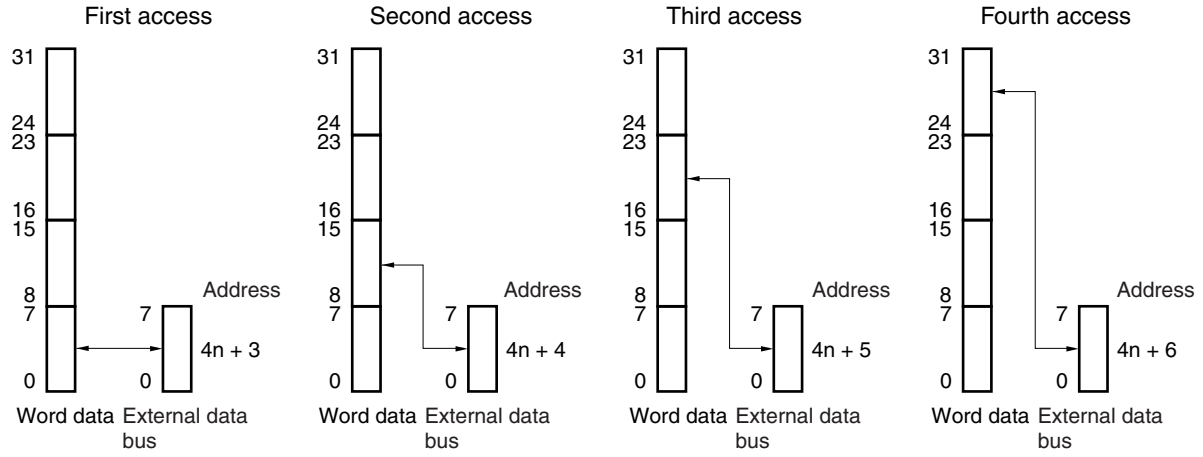


(b) 8-bit data bus width (2/2)

<3> Access to address (4n + 2)



<4> Access to address (4n + 3)



5.6 Wait Function

5.6.1 Programmable wait function

(1) Data wait control register 0 (DWC0)

To realize interfacing with a low-speed memory or I/O, up to seven data wait states can be inserted in the bus cycle that is executed for each memory block space.

The number of wait states can be programmed by using the DWC0 register . Immediately after system reset, 7 data wait states are inserted for all the blocks.

The DWC0 register can be read or written in 16-bit units.

Reset sets this register to 7777H.

Cautions 1. The internal ROM and internal RAM areas are not subject to programmable wait, and are always accessed without a wait state. The on-chip peripheral I/O area is also not subject to programmable wait, and only wait control from each peripheral function is performed.

2. Write to the DWC0 register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the DWC0 register are complete.

After reset: 7777H R/W Address: FFFFF484H

	15	14	13	12	11	10	9	8
DWC0	0	DW32	DW31	DW30	0	DW22	DW21	DW20
		Memory block 3				Memory block 2		

	7	6	5	4	3	2	1	0
	0	DW12	DW11	DW10	0	DW02	DW01	DW00
		Memory block 1				Memory block 0		

DWn2	DWn1	DWn0	Number of wait states inserted in memory block n space (n = 0 to 3)
0	0	0	None
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Caution Be sure to clear bits 15, 11, 7, and 3 to “0”.

5.6.2 External wait function

To synchronize an extremely slow external memory, I/O, or asynchronous system, any number of wait states can be inserted in the bus cycle by using the external wait pin ($\overline{\text{WAIT}}$).

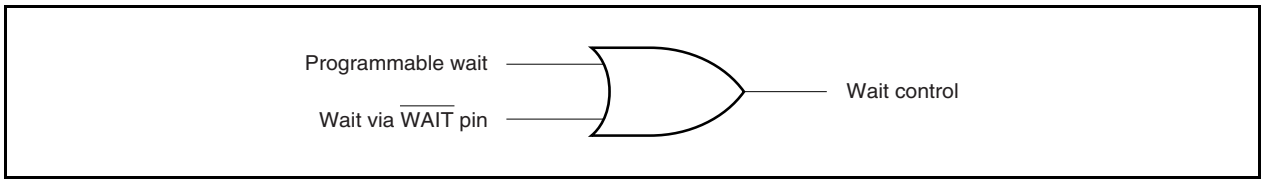
When the PCM0 pin is set to alternate function, the external wait function is enabled.

Access to each area of the internal ROM, internal RAM, and on-chip peripheral I/O is not subject to control by the external wait function, in the same manner as the programmable wait function.

The $\overline{\text{WAIT}}$ signal can be input asynchronously to CLKOUT, and is sampled at the falling edge of the clock in the T2 and TW states of the bus cycle in the multiplexed bus mode. In the separate bus mode, it is sampled at the rising edge of the clock immediately after the T1 and TW states of the bus cycle. If the setup/hold time of the sampling timing is not satisfied, a wait state is inserted in the next state, or not inserted at all.

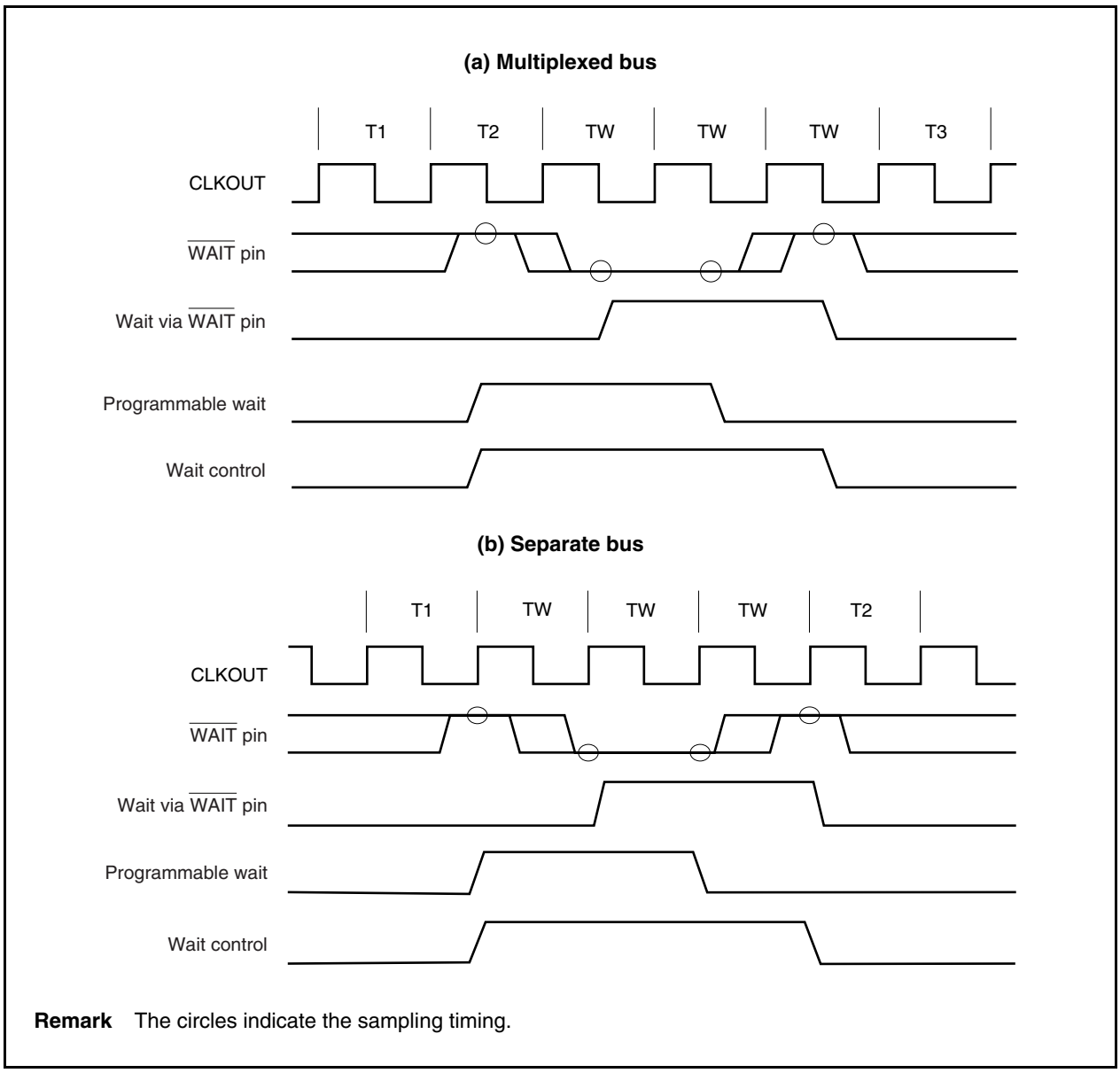
5.6.3 Relationship between programmable wait and external wait

Wait cycles are inserted as the result of an OR operation between the wait cycles specified by the set value of the programmable wait and the wait cycles controlled by the $\overline{\text{WAIT}}$ pin.



For example, if the timing of the programmable wait and the $\overline{\text{WAIT}}$ pin signal is as illustrated below, three wait states will be inserted in the bus cycle.

Figure 5-3. Inserting Wait Example



Remark The circles indicate the sampling timing.

5.6.4 Programmable address wait function

Address-setup or address-hold waits to be inserted in each bus cycle can be set by using the AWC register. Address wait insertion is set for each memory block area (memory blocks 0 to 3).

If an address setup wait is inserted, it seems that the high-clock period of the T1 state is extended by 1 clock. If an address hold wait is inserted, it seems that the low-clock period of the T1 state is extended by 1 clock.

(1) Address wait control register (AWC)

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

Reset sets this register to FFFFH.

Cautions 1. Address setup wait and address hold wait cycles are not inserted when the internal ROM area, internal RAM area, and on-chip peripheral I/O areas are accessed.

2. Write to the AWC register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the AWC register are complete.

After reset: FFFFH R/W Address: FFFFF488H

	15	14	13	12	11	10	9	8
AWC	1	1	1	1	1	1	1	1

7	6	5	4	3	2	1	0
AHW3	ASW3	AHW2	ASW2	AHW1	ASW1	AHW0	ASW0
Memory block 3		Memory block 2		Memory block 1		Memory block 0	

AHWn	Specifies insertion of address hold wait (n = 0 to 3)
0	Not inserted
1	Inserted

ASWn	Specifies insertion of address setup wait (n = 0 to 3)
0	Not inserted
1	Inserted

Caution Be sure to set bits 15 to 8 to “1”.

5.7 Idle State Insertion Function

To facilitate interfacing with low-speed memories, one idle state (TI) can be inserted after the T3 state in the bus cycle that is executed for each space selected by the memory block in the multiplex address/data bus mode. In the separate bus mode, one idle state (TI) can be inserted after the T2 state. By inserting an idle state, the data output float delay time of the memory can be secured during read access (an idle state cannot be inserted during write access).

Whether the idle state is to be inserted can be programmed by using the BCC register.

An idle state is inserted for all the areas immediately after system reset.

(1) Bus cycle control register (BCC)

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

Reset sets this register to AAAAH.

Cautions 1. The internal ROM, internal RAM, and on-chip peripheral I/O areas are not subject to idle state insertion.

2. Write to the BCC register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the BCC register are complete.

After reset: AAAAH R/W Address: FFFFF48AH

	15	14	13	12	11	10	9	8
BCC	1	0	1	0	1	0	1	0

7	6	5	4	3	2	1	0
BC31	0	BC21	0	BC11	0	BC01	0

Memory block 3 Memory block 2 Memory block 1 Memory block 0

BCn1	Specifies insertion of idle state (n = 0 to 3)
0	Not inserted
1	Inserted

Caution Be sure to set bits 15, 13, 11, and 9 to "1", and clear bits 14, 12, 10, 8, 6, 4, 2, and 0 to "0".

5.8 Bus Hold Function

5.8.1 Functional outline

The $\overline{\text{HLDRQ}}$ and $\overline{\text{HLDAK}}$ functions are valid if the PCM2 and PCM3 pins are set to alternate function.

When the $\overline{\text{HLDRQ}}$ pin is asserted (low level), indicating that another bus master has requested bus mastership, the external address/data bus goes into a high-impedance state and is released (bus hold status). If the request for the bus mastership is cleared and the $\overline{\text{HLDRQ}}$ pin is deasserted (high level), driving these pins is started again.

During the bus hold period, execution of the program in the internal ROM and internal RAM is continued until an on-chip peripheral I/O register or the external memory is accessed.

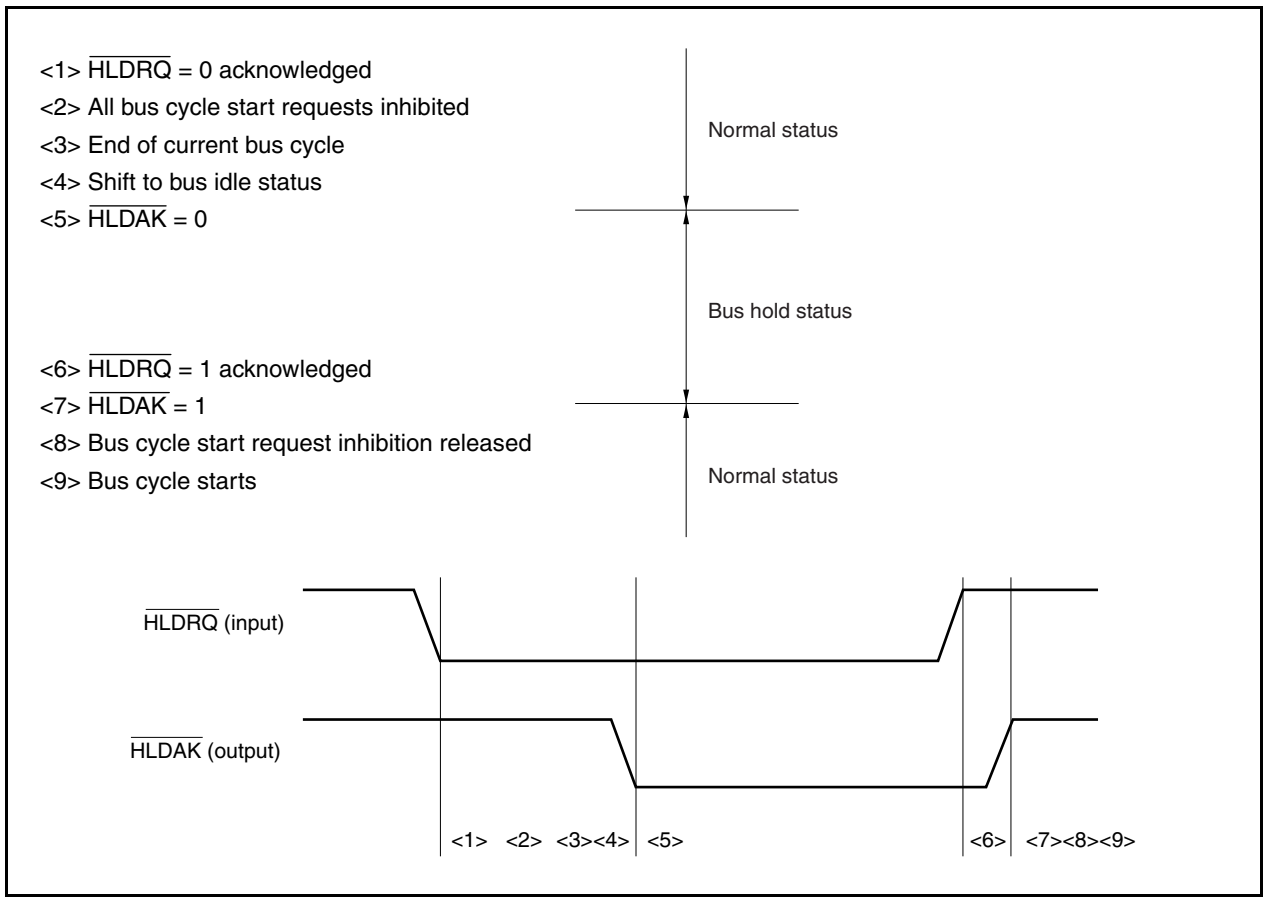
The bus hold status is indicated by assertion of the $\overline{\text{HLDAK}}$ pin (low level). The bus hold function enables the configuration of multi-processor type systems in which two or more bus masters exist.

Note that the bus hold request is not acknowledged during a multiple-access cycle initiated by the bus sizing function or a bit manipulation instruction.

Status	Data Bus Width	Access Type	Timing at Which Bus Hold Request Is Not Acknowledged
CPU bus lock	16 bits	Word access to even address	Between first and second access
		Word access to odd address	Between first and second access
			Between second and third access
		Halfword access to odd address	Between first and second access
	8 bits	Word access	Between first and second access
			Between second and third access
			Between third and fourth access
		Halfword access	Between first and second access
Read-modify-write access of bit manipulation instruction	–	–	Between read access and write access

5.8.2 Bus hold procedure

The bus hold status transition procedure is shown below.



5.8.3 Operation in power save mode

Because the internal system clock is stopped in the STOP, IDLE1, and IDLE2 modes, the bus hold status is not entered even if the $\overline{\text{HLDRQ}}$ pin is asserted.

In the HALT mode, the $\overline{\text{HLDAK}}$ pin is asserted as soon as the $\overline{\text{HLDRQ}}$ pin has been asserted, and the bus hold status is entered. When the $\overline{\text{HLDRQ}}$ pin is later deasserted, the $\overline{\text{HLDAK}}$ pin is also deasserted, and the bus hold status is cleared.

5.9 Bus Priority

<R> Bus hold, DMA transfer[Note], operand data accesses, instruction fetch (branch), and instruction fetch (successive) are executed in the external bus cycle.

<R> Bus hold has the highest priority, followed by DMA transfer[Note], operand data access, instruction fetch (branch), and instruction fetch (successive).

An instruction fetch may be inserted between the read access and write access in a read-modify-write access.

If an instruction is executed for two or more accesses, an instruction fetch and bus hold are not inserted between accesses due to bus size limitations.

<R> **Note** μPD703253Y only

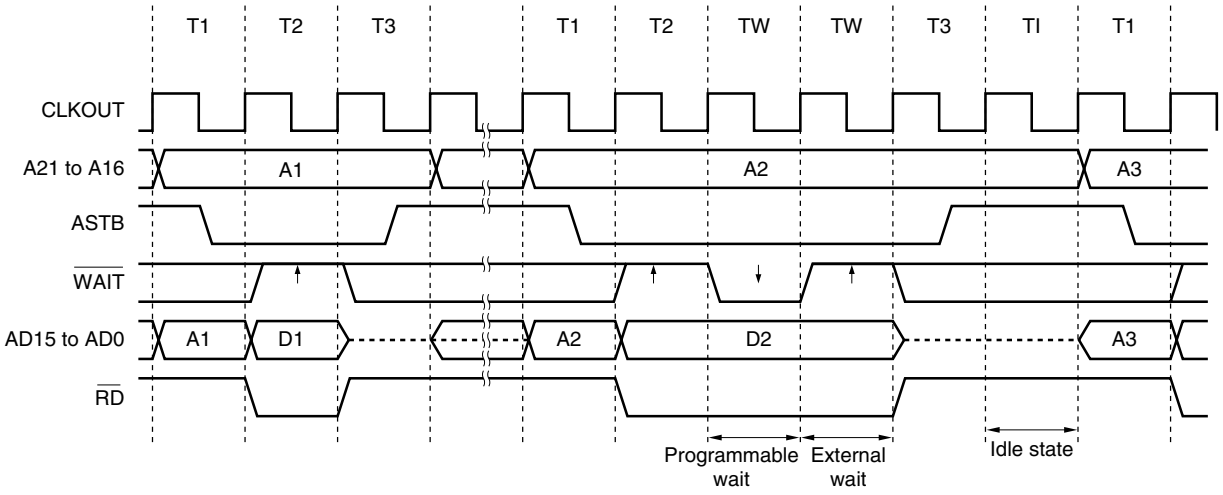
Table 5-4. Bus Priority

Priority	External Bus Cycle	Bus Master
High	Bus hold	External device
<R> ↕	DMA transfer[Note]	DMAC
	Operand data access	CPU
	Instruction fetch (branch)	CPU
Low	Instruction fetch (successive)	CPU

<R> **Note** μPD703253Y only

5.10 Bus Timing

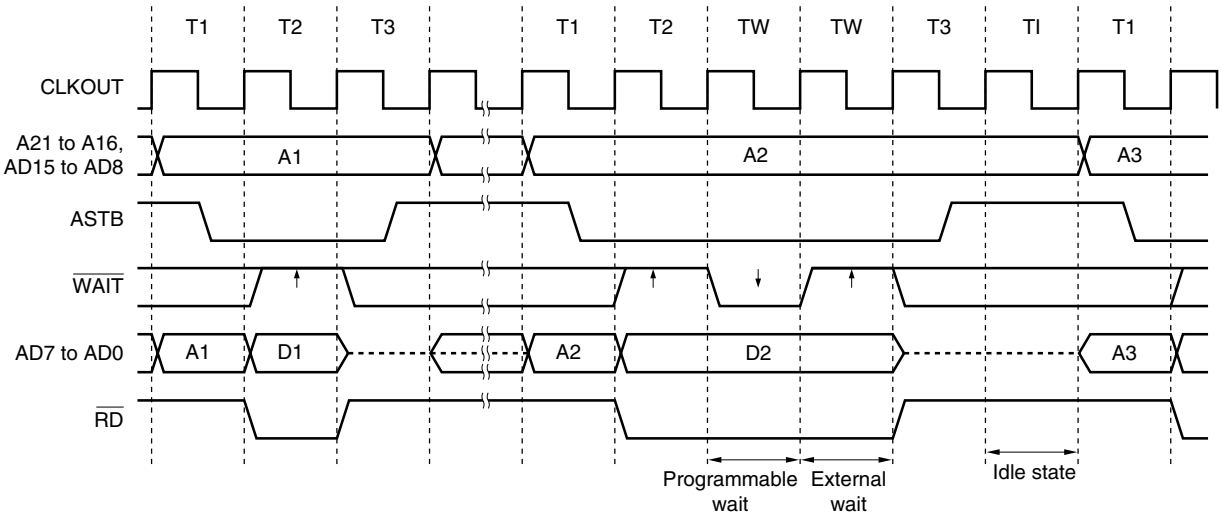
Figure 5-4. Multiplexed Bus Read Timing (Bus Size: 16 Bits, 16-Bit Access)



8-bit access	Odd address	Even address
AD15 to AD8	Active	Hi-Z
AD7 to AD0	Hi-Z	Active

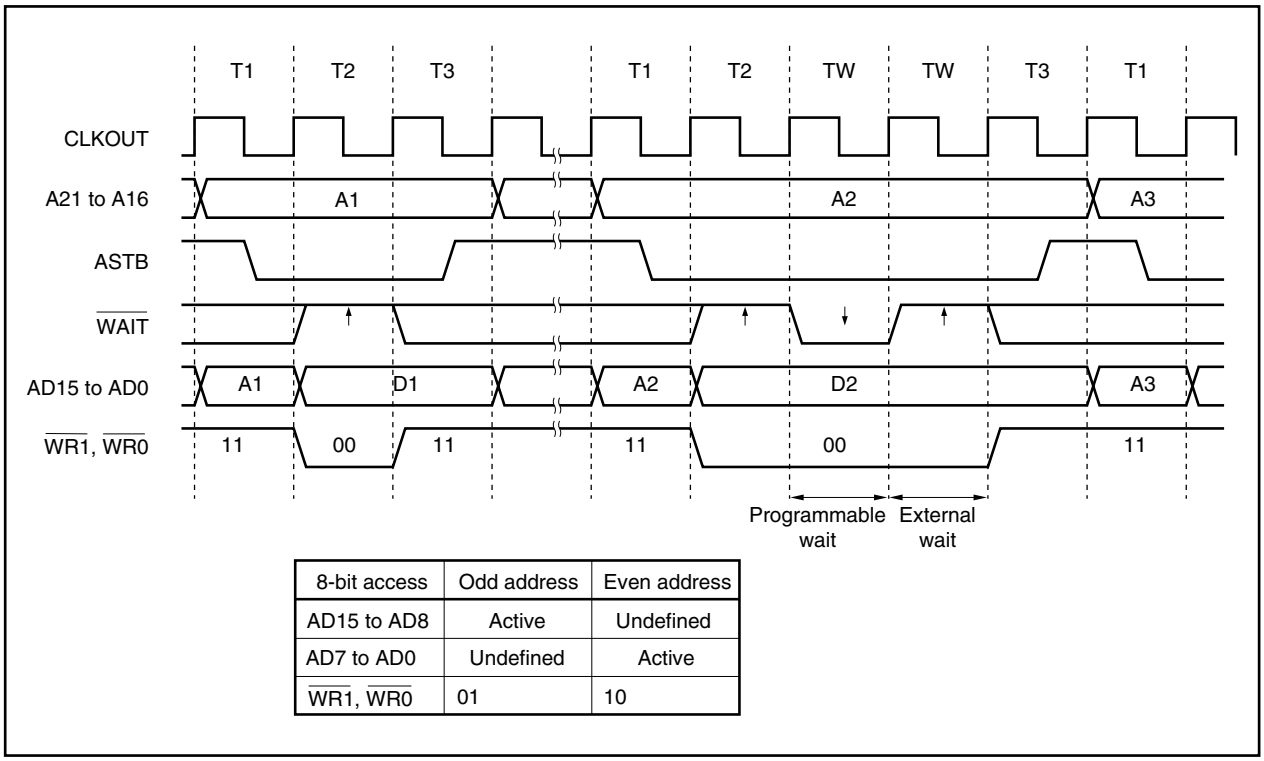
Remark The broken lines indicate high impedance.

Figure 5-5. Multiplexed Bus Read Timing (Bus Size: 8 Bits)



Remark The broken lines indicate high impedance.

Figure 5-6. Multiplexed Bus Write Timing (Bus Size: 16 Bits, 16-Bit Access)



8-bit access	Odd address	Even address
AD15 to AD8	Active	Undefined
AD7 to AD0	Undefined	Active
$\overline{WR1}$, $\overline{WR0}$	01	10

Figure 5-7. Multiplexed Bus Write Timing (Bus Size: 8 Bits)

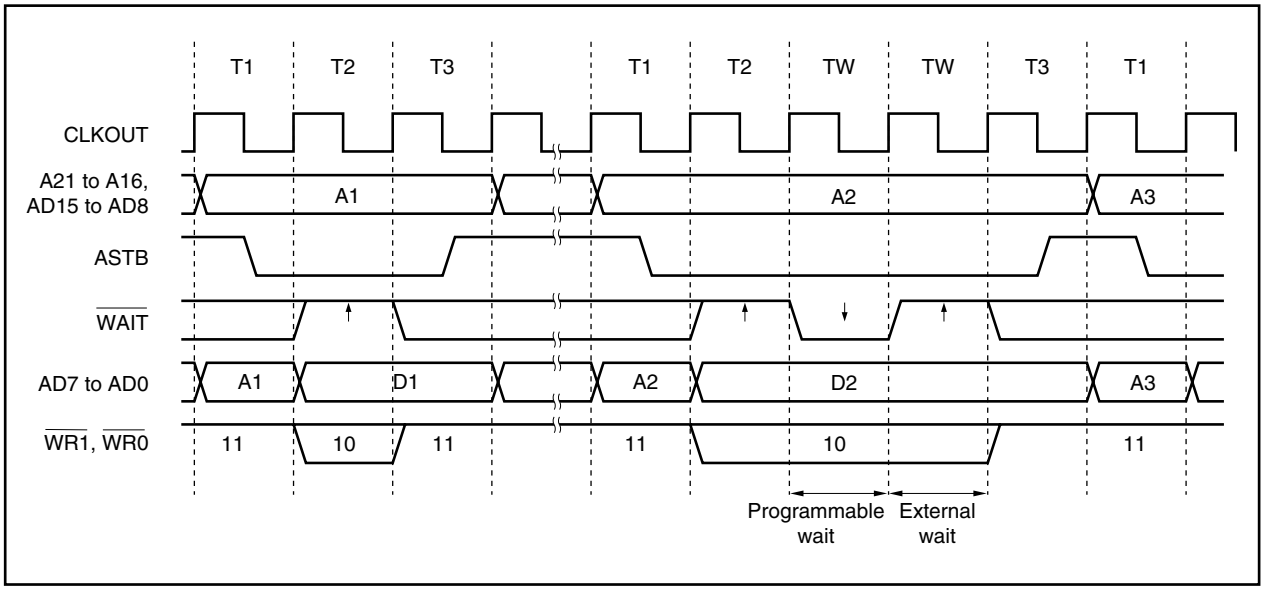
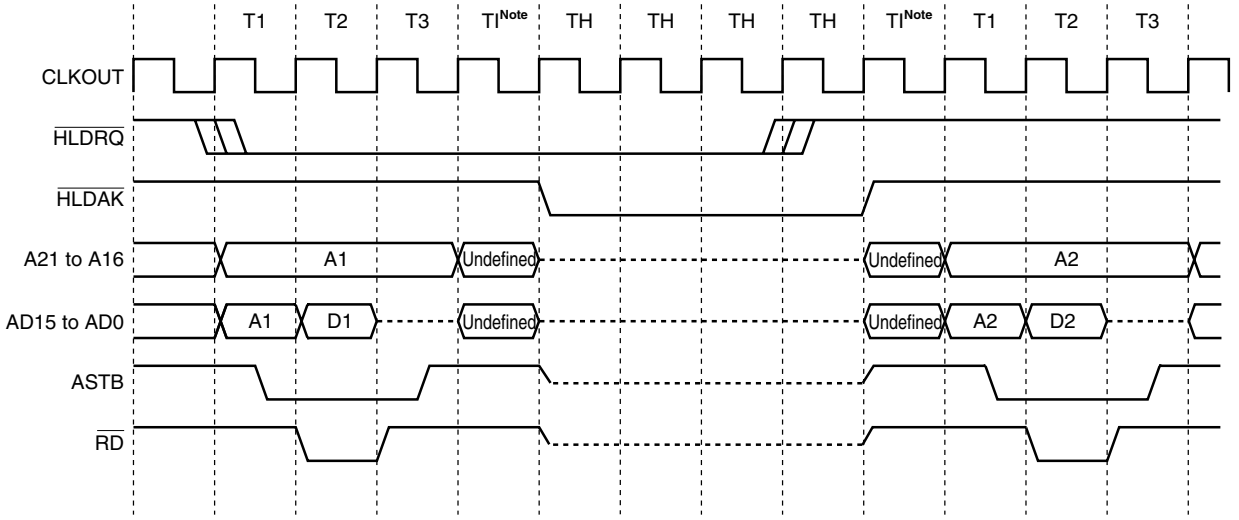


Figure 5-8. Multiplexed Bus Hold Timing (Bus Size: 16 Bits, 16-Bit Access)

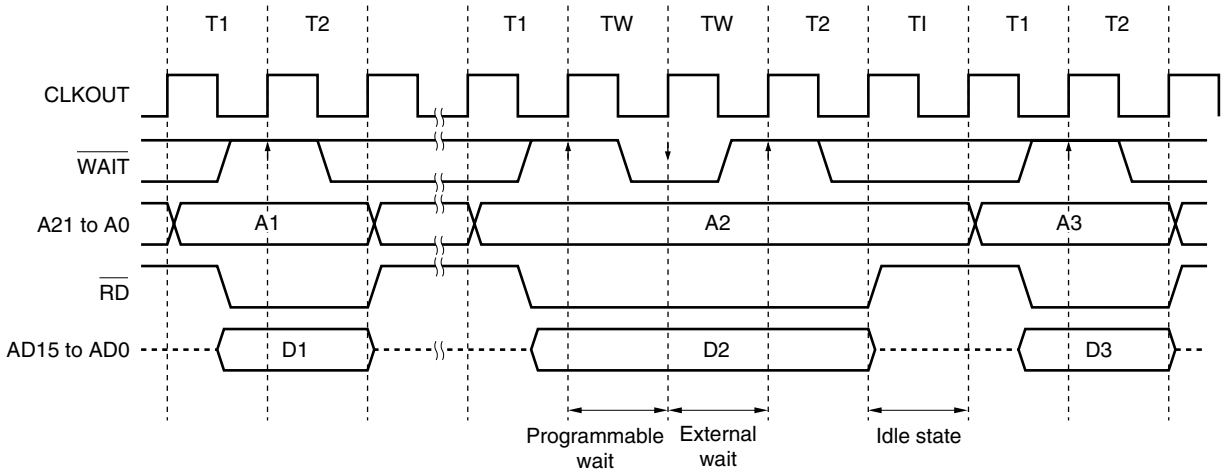


Note This idle state (TI) does not depend on the BCC register settings.

Remarks 1. See **Table 2-2** for the pin statuses in the bus hold mode.

2. The broken lines indicate high impedance.

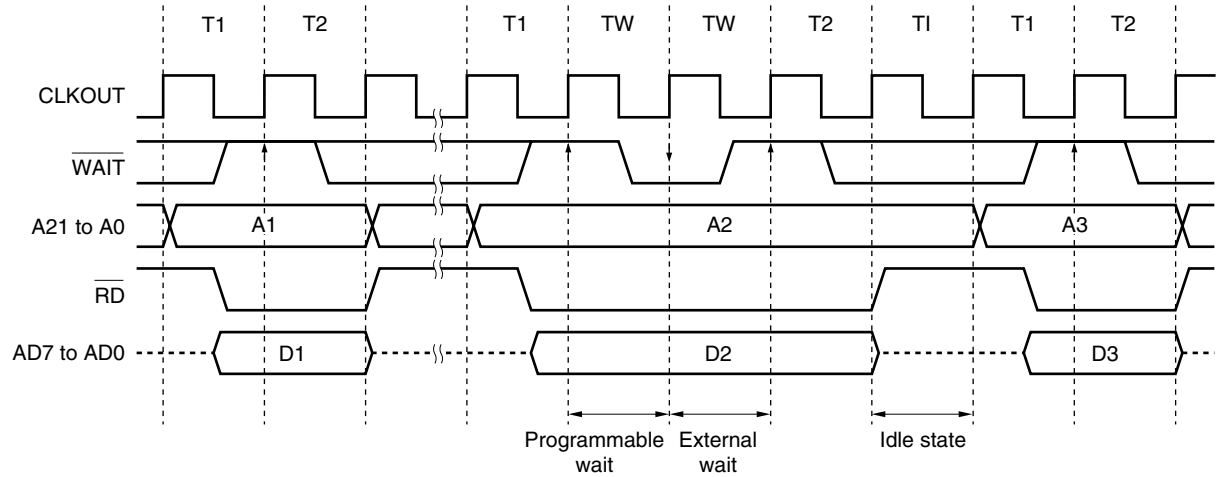
Figure 5-9. Separate Bus Read Timing (Bus Size: 16 Bits, 16-Bit Access)



8-bit access	Odd address	Even address
AD15 to AD8	Active	Hi-Z
AD7 to AD0	Hi-Z	Active

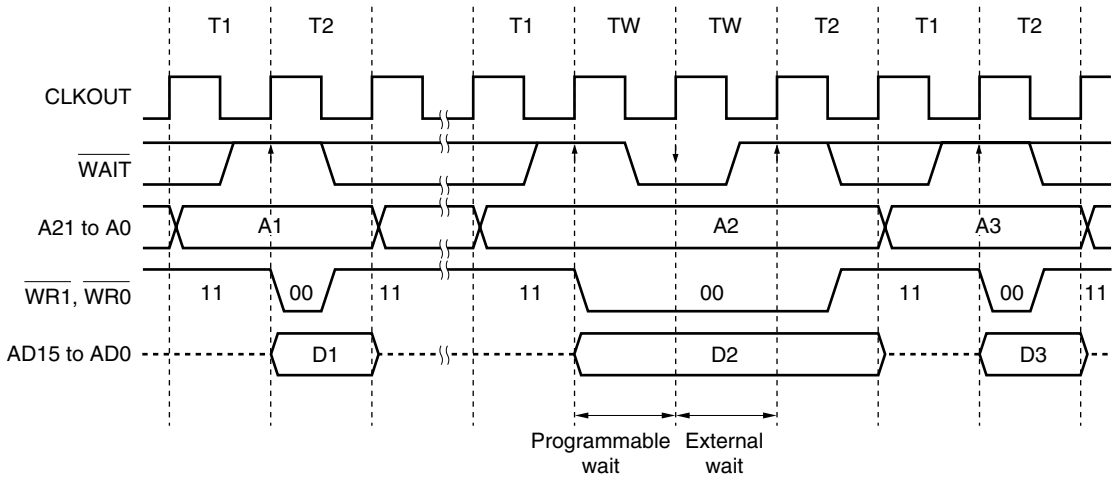
Remark The broken lines indicate high impedance.

Figure 5-10. Separate Bus Read Timing (Bus Size: 8 Bits)



Remark The broken lines indicate high impedance.

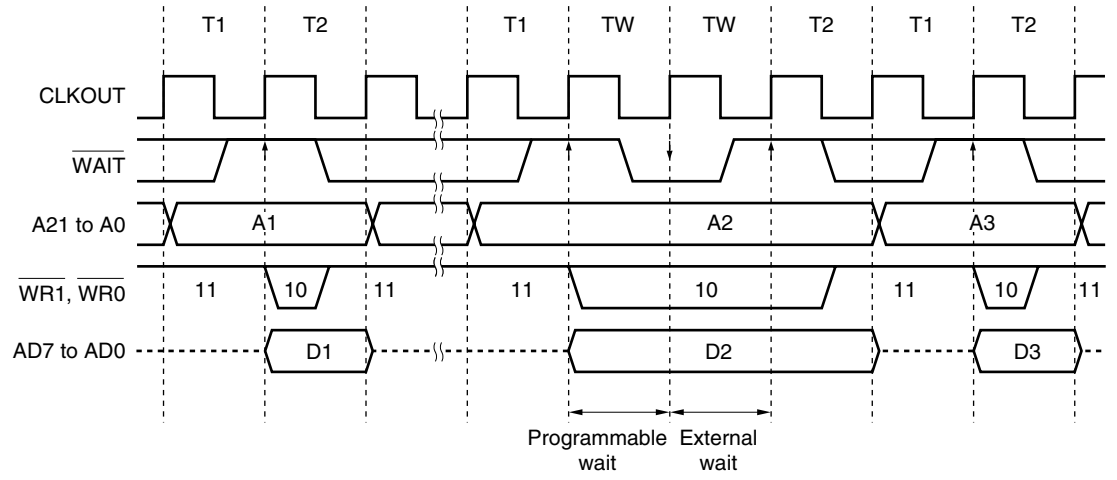
Figure 5-11. Separate Bus Write Timing (Bus Size: 16 Bits, 16-Bit Access)



8-bit access	Odd address	Even address
AD15 to AD8	Active	Undefined
AD7 to AD0	Undefined	Active
$\overline{WR1}$, $\overline{WR0}$	01	10

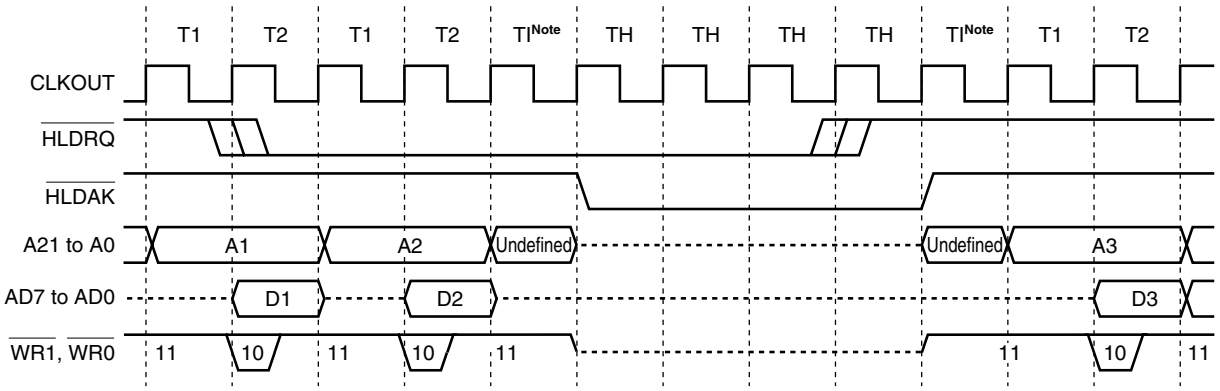
Remark The broken lines indicate high impedance.

Figure 5-12. Separate Bus Write Timing (Bus Size: 8 Bits)



Remark The broken lines indicate high impedance.

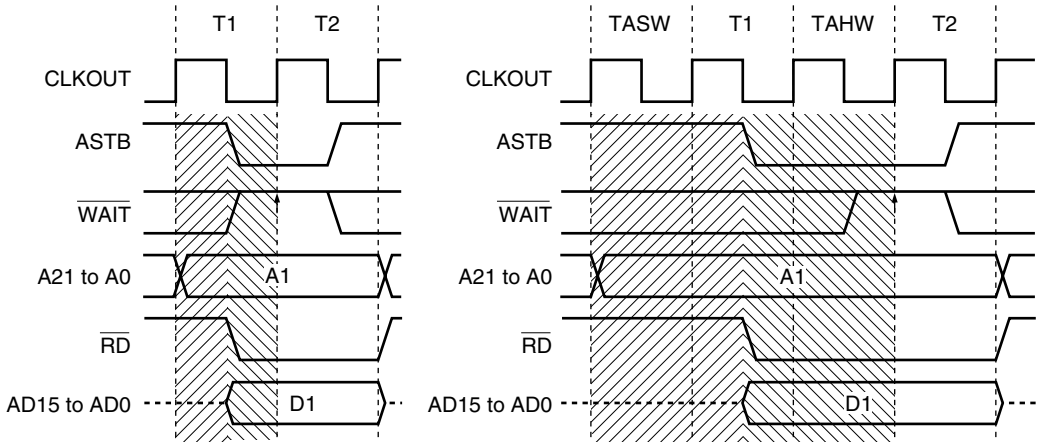
Figure 5-13. Separate Bus Hold Timing (Bus Size: 8 Bits, Write)



Note This idle state (TI) does not depend on the BCC register settings.

Remark The broken lines indicate high impedance.

Figure 5-14. Address Wait Timing (Separate Bus Read, Bus Size: 16 Bits, 16-Bit Access)



Remarks 1. TASW (address setup wait): Image of high-level width of T1 state expanded.

2. TAHW (address hold wait): Image of low-level width of T1 state expanded.

3. The broken lines indicate high impedance.

CHAPTER 6 CLOCK GENERATION FUNCTION

6.1 Overview

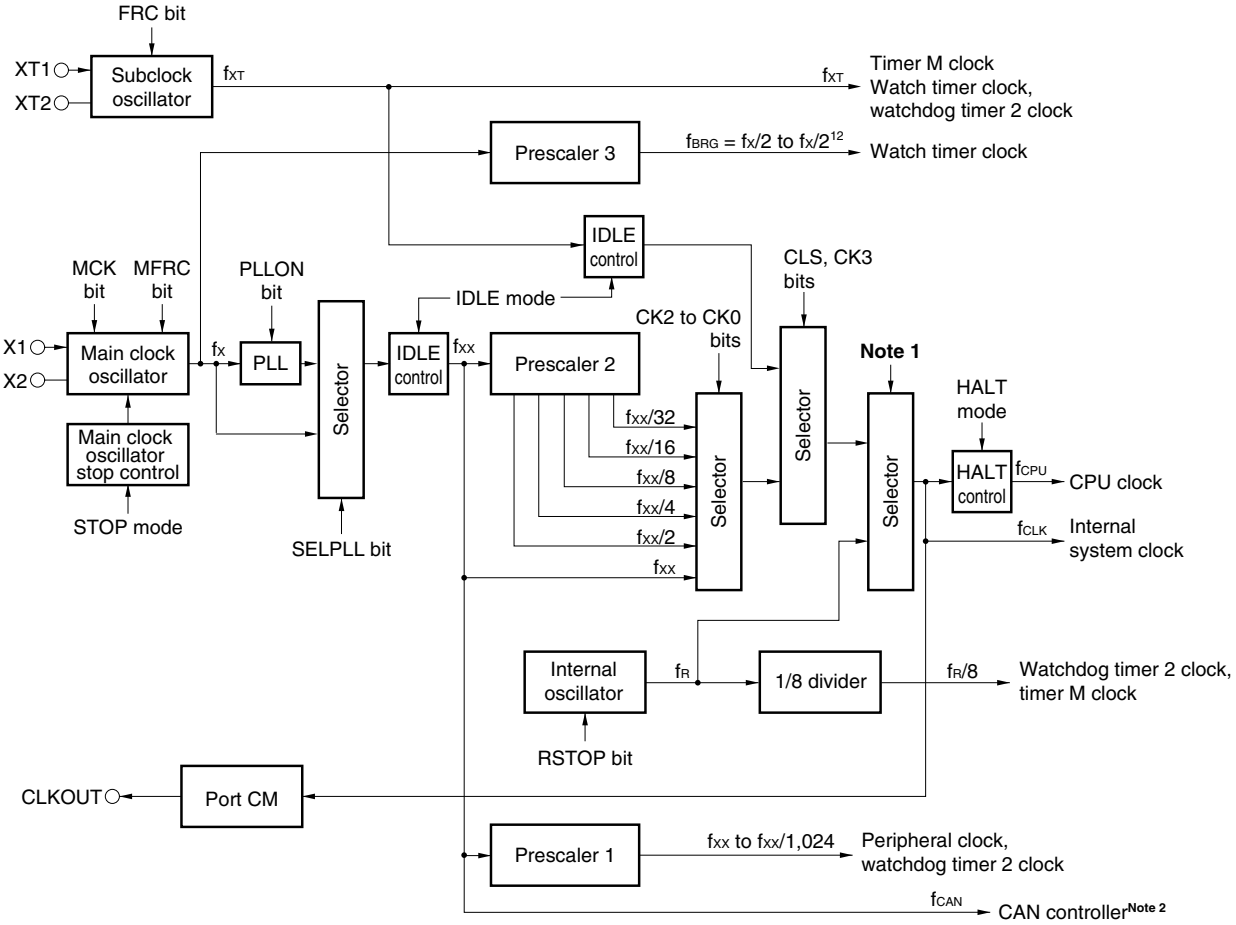
The following clock generation functions are available.

- ○ Main clock oscillator
  - In clock-through mode
    f_X = 2.5 to 10 MHz (f_{XX} = 2.5 to 10 MHz)
  - In PLL mode
    f_X = 2.5 to 5 MHz (f_{XX} = 10 to 20 MHz)
- ○ Subclock oscillator
  - f_{XT} = 32.768 kHz
- ○ Multiply (×4/×8) function by PLL (Phase Locked Loop)
  - Clock-through mode/PLL mode selectable
- ○ Internal oscillator
  - f_R = 200 kHz (TYP.)
- ○ Internal system clock generation
  - 7 steps (f_{XX}, $f_{XX}/2$, $f_{XX}/4$, $f_{XX}/8$, $f_{XX}/16$, $f_{XX}/32$, f_{XT})
- ○ Peripheral clock generation
- ○ Clock output function

Remark f_X: Main clock oscillation frequency
f_{XX}: Main clock frequency
f_{XT}: Subclock frequency
f_R: Internal oscillation clock frequency

6.2 Configuration

Figure 6-1. Clock Generator



Notes 1. The internal oscillation clock is selected when watchdog timer 2 overflows during the oscillation stabilization time.

<R> **2.** μPD703253Y only

Remark f_X: Main clock oscillation frequency
f_{XX}: Main clock frequency
f_{CLK}: Internal system clock frequency
f_{XT}: Subclock frequency
f_{CPU}: CPU clock frequency
f_{BRG}: Watch timer clock frequency
f_R: Internal oscillation clock frequency
f_{CAN}: CAN clock frequency

(1) Main clock oscillator

The main resonator oscillates the following frequencies (f_X).

- In clock-through mode
  f_X = 2.5 to 10 MHz
- In PLL mode
  f_X = 2.5 to 5 MHz

(2) Subclock oscillator

The sub-resonator oscillates a frequency of 32.768 kHz (f_{XT}).

(3) Main clock oscillator stop control

This circuit generates a control signal that stops oscillation of the main clock oscillator.

Oscillation of the main clock oscillator is stopped in the STOP mode or when the PCC.MCK bit = 1 (valid only when the PCC.CLS bit = 1).

(4) Internal oscillator

Oscillates a frequency (f_R) of 200 kHz (TYP.).

(5) Prescaler 1

This prescaler generates the clock (f_{XX} to f_{XX}/1,024) to be supplied to the following on-chip peripheral functions:

- μPD703249Y, 703252Y:
  TMP0 to TMP4, TMM0, CSIB0, CSIB1, CSIB3, UARTA0, UARTA1, I^2C00, I^2C01, ADC, WDT2, IEBus (μPD703252Y only)

<R>

- μPD703253Y:
  TMP0 to TMP5, TMM0, CSIB0, CSIB1, CSIB3, UARTA0, UARTA1, I^2C00, I^2C01, ADC, WDT2, CAN0

(6) Prescaler 2

This circuit divides the main clock (f_{XX}).

The clock generated by prescaler 2 (f_{XX} to f_{XX}/32) is supplied to the selector that generates the CPU clock (f_{CPU}) and internal system clock (f_{CLK}).

f_{CLK} is the clock supplied to the INTC, ROM correction, ROM, and RAM blocks, and can be output from the CLKOUT pin.

(7) Prescaler 3

This circuit divides the clock generated by the main clock oscillator (f_X) to a specific frequency (32.768 kHz) and supplies that clock to the watch timer block.

For details, see **CHAPTER 9 WATCH TIMER FUNCTIONS**.

(8) PLL

This circuit multiplies the clock generated by the main clock oscillator (f_X) by 4 or 8.

It operates in two modes: clock-through mode in which f_X is output as is, and PLL mode in which a multiplied clock is output. These modes can be selected by using the PLLCTL.SELPLL bit.

Whether the clock is multiplied by 4 or 8 is selected by the CKC.CKDIV0 bit, and PLL is started or stopped by the PLLCTL.PLLON bit.

6.3 Registers

(1) Processor clock control register (PCC)

The PCC register is a special register. Data can be written to this register only in combination of specific sequences (see **3.4.8 Special registers**).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 03H.

After reset: 03H R/W Address: FFFFF828H

	7	<6>	5	<4>	<3>	2	1	0
PCC	FRC	MCK	MFRC	CLS[Note]	CK3	CK2	CK1	CK0

FRC	Use of subclock on-chip feedback resistor
0	Used
1	Not used

MCK	Main clock oscillator control
0	Oscillation enabled
1	Oscillation stopped

- Even if the MCK bit is set (1) while the system is operating with the main clock as the CPU clock, the operation of the main clock does not stop. It stops after the CPU clock has been changed to the subclock.
- Before setting the MCK bit from 0 to 1, stop the on-chip peripheral functions operating with the main clock.
- When the main clock is stopped and the device is operating with the subclock, clear (0) the MCK bit and secure the oscillation stabilization time by software before switching the CPU clock to the main clock or operating the on-chip peripheral functions.

MFRC	Use of main clock on-chip feedback resistor
0	Used
1	Not used

CLS[Note]	Status of CPU clock (f_{CPU})
0	Main clock operation
1	Subclock operation

CK3	CK2	CK1	CK0	Clock selection (f_{CLK}/f_{CPU})
0	0	0	0	f_{XX}
0	0	0	1	$f_{XX}/2$
0	0	1	0	$f_{XX}/4$
0	0	1	1	$f_{XX}/8$
0	1	0	0	$f_{XX}/16$
0	1	0	1	$f_{XX}/32$
0	1	1	×	Setting prohibited
1	×	×	×	f_{XT}

Note The CLS bit is a read-only bit.

Cautions 1. Do not change the CPU clock (by using the CK3 to CK0 bits) while CLKOUT is being output.

2. Use a bit manipulation instruction to manipulate the CK3 bit. When using an 8-bit manipulation instruction, do not change the set values of the CK2 to CK0 bits.

Remark ×: don't care

(a) Example of setting main clock operation → subclock operation

<1> CK3 bit ← 1: Use of a bit manipulation instruction is recommended. Do not change the CK2 to CK0 bits.

<2> Subclock operation: Read the CLS bit to check if subclock operation has started. It takes the following time after the CK3 bit is set until subclock operation is started.

Max.: 1/f_{XT} (1/subclock frequency)

<3> MCK bit ← 1: Set the MCK bit to 1 only when stopping the main clock.

Cautions 1. When stopping the main clock, stop the PLL. Also stop the operations of the on-chip peripheral functions operating with the main clock.

2. If the following conditions are not satisfied, change the CK2 to CK0 bits so that the conditions are satisfied, then change to the subclock operation mode.

Internal system clock (f_{CLK}) > Subclock (f_{XT}: 32.768 kHz) × 4

Remark Internal system clock (f_{CLK}): Clock generated from the main clock (f_{XX}) by setting bits CK2 to CK0

[Description example]

```
<R>          _DMA_DISABLE:Note
<R>          clrl        0, DCHCn[r0]        -- DMA operation disabled.  n = 0 to 3Note
<1>  _SET_SUB_RUN :
             st.b        r0, PRCMD[r0]
             set1        3, PCC[r0]          -- CK3 bit ← 1
<2>  _CHECK_CLS :
             tst1        4, PCC[r0]          -- Wait until subclock operation starts.
             bz          _CHECK_CLS
<3>  _STOP_MAIN_CLOCK :
             st.b        r0, PRCMD[r0]
             set1        6, PCC[r0]          -- MCK bit ← 1, main clock is stopped.
<R>          _DMA_ENABLE:Note
<R>          set1        0, DCHCn[r0]        -- DMA operation enabled.  n = 0 to 3Note
```

<R> **Note** μPD703253Y only

Remark The description above is simply an example. Note that in <2> above, the CLS bit is read in a closed loop.

(b) Example of setting subclock operation → main clock operation

<1> MCK bit ← 1: Main clock starts oscillating

<2> Insert waits by the program and wait until the oscillation stabilization time of the main clock elapses.

<3> CK3 bit ← 1: Use of a bit manipulation instruction is recommended. Do not change the CK2 to CK0 bits.

<4> Main clock operation: It takes the following time after the CK3 bit is set until main clock operation is started.

Max.: 1/f_{XT} (1/subclock frequency)

Therefore, insert one NOP instruction immediately after setting the CK3 bit to 0 or read the CLS bit to check if main clock operation has started.

Caution Enable operation of the on-chip peripheral functions operating with the main clock only after the oscillation of the main clock stabilizes. If their operations are enabled before the lapse of the oscillation stabilization time, a malfunction may occur.

[Description example]

```
<R>      _DMA_DISABLE:Note
<R>      clrl      0, DCHCn[r0]          -- DMA operation disabled. n = 0 to 3Note
<1>  _START_MAIN_OSC :
         st.b      r0, PRCMD[r0]         -- Release of protection of special registers
         clr1      6, PCC[r0]            -- Main clock starts oscillating.
<2>  movea     0x55, r0, r11            -- Wait for oscillation stabilization time.
     _WAIT_OST :
     nop
     nop
     nop
     addi      -1, r11, r11
     cmp       r0, r11
     bne              _WAIT_OST
<3>  st.b      r0, PRCMD[r0]
     clr1      3, PCC[r0]               -- CK3 ← 0
<4>  _CHECK_CLS :
     tst1      4, PCC[r0]               -- Wait until main clock operation starts.
     bnz       _CHECK_CLS
<R>      _DMA_ENABLE:Note
<R>      setl      0, DCHCn[r0]          -- DMA operation enabled. n = 0 to 3Note
```

<R> **Note** μPD703253Y only

Remark The description above is simply an example. Note that in <4> above, the CLS bit is read in a closed loop.

(2) Internal oscillation mode register (RCM)

The RCM register is an 8-bit register that sets the operation mode of the internal oscillator.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF80CH

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

RSTOP	Oscillation/stop of internal oscillator
0	Internal oscillator oscillation
1	Internal oscillator stopped

Cautions 1. The internal oscillator cannot be stopped while the CPU is operating on the internal oscillation clock (CCLS.CCLSF bit = 1). Do not set the RSTOP bit to 1.

2. The internal oscillator oscillates if the CCLS.CCLSF bit is set to 1 (when WDT overflow occurs during oscillation stabilization) even when the RSTOP bit is set to 1. At this time, the RSTOP bit remains being set to 1.

(3) CPU operation clock status register (CCLS)

The CCLS register indicates the status of the CPU operation clock.

This register is read-only, in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H[Note] R Address: FFFFF82EH

	7	6	5	4	3	2	1	0
CCLS	0	0	0	0	0	0	0	CCLSF

CCLSF	CPU operation clock status
0	Operating on main clock (f_X) or subclock (f_{XT}).
1	Operating on internal oscillation clock (f_R).

<R> **Note** If WDT overflow occurs during oscillation stabilization time after a reset is released, the CPU operates on the internal oscillation clock (f_R). At this time, the CCLSF bit is set to 1 and the reset value is 01H.

6.4 Operation

6.4.1 Operation of each clock

The following table shows the operation status of each clock.

Table 6-1. Operation Status of Each Clock

Register Setting and Operation Status / Target Clock	PCC Register								
	CLK Bit = 0, MCK Bit = 0					CLS Bit = 1, MCK Bit = 0		CLS Bit = 1, MCK Bit = 1	
	During Reset	During Oscillation Stabilization Time Count	HALT Mode	IDLE1, IDLE2 Mode	STOP Mode	Subclock Mode	Sub-IDLE Mode	Subclock Mode	Sub-IDLE Mode
Main clock oscillator (f_X)	×	√	√	√	×	√	√	×	×
Subclock oscillator (f_{XT})	√	√	√	√	√	√	√	√	√
CPU clock (f_{CPU})	×	×	×	×	×	√	×	√	×
Internal system clock (f_{CLK})	×	×	√	×	×	√	×	√	×
Main clock (in PLL mode, f_{XX})	×	√[Note]	√	×	×	√	√	×	×
Peripheral clock (f_{XX} to f_{XX}/1,024)	×	×	√	×	×	√	×	×	×
WT clock (main)	×	√	√	√	×	√	√	×	×
WT clock (sub)	√	√	√	√	√	√	√	√	√
WDT2 clock (internal oscillation)	×	√	√	√	√	√	√	√	√
WDT2 clock (main)	×	×	√	×	×	√	×	×	×
WDT2 clock (sub)	√	√	√	√	√	√	√	√	√

Note Lockup time

Remark √: Operable
×: Stopped

6.4.2 Clock output function

The clock output function is used to output the internal system clock (f_{CLK}) from the CLKOUT pin.

The internal system clock (f_{CLK}) is selected by using the PCC.CK3 to PCC.CK0 bits.

The CLKOUT pin functions alternately as the PCM1 pin and functions as a clock output pin if so specified by the control register of port CM.

The status of the CLKOUT pin is the same as the internal system clock in Table 6-1 and the pin can output the clock when it is in the operable status. It outputs a low level in the stopped status. However, the CLKOUT pin is in the port mode (PCM1 pin: input mode) after reset and until it is set in the output mode. Therefore, the status of the pin is Hi-Z.

6.5 PLL Function

6.5.1 Overview

In the V850ES/SG1, an operating clock that is 4 or 8 times higher than the oscillation frequency output by the PLL function or the clock-through mode can be selected as the operating clock of the CPU and on-chip peripheral functions.

When PLL function is used: Input clock = 2.5 to 5 MHz (output: 10 to 20 MHz)
Clock-through mode: Input clock = 2.5 to 10 MHz (output: 2.5 to 10 MHz)

6.5.2 Registers

(1) PLL control register (PLLCTL)

The PLLCTL register is an 8-bit register that controls the PLL function.
This register can be read or written in 8-bit or 1-bit units.
Reset sets this register to 01H.

After reset: 01H R/W Address: FFFFF82CH

	7	6	5	4	3	2	<1>	<0>
PLLCTL	0	0	0	0	0	0	SELPLL	PLLON

PLLON	PLL operation stop register
0	PLL stopped
1	PLL operating (After PLL operation starts, a lockup time is required for frequency stabilization)

SELPLL	CPU operation clock selection register
0	Clock-through mode
1	PLL mode

<R> **Cautions 1. To stop the PLL operation, first set the clock through mode (SELPLL bit = 0), wait for at least 8 clocks, and then stop the PLL (PLLON bit = 0). When the PLLON bit is cleared to 0, the SELPLL bit is automatically cleared to 0 (clock-through mode), but be sure to stop the PLL in the above procedure.**

2. The SELPLL bit can be set to 1 only when the PLL clock frequency is stabilized. If not (unlocked), "0" is written to the SELPLL bit if data is written to it.

(2) Clock control register (CKC)

The CKC register is a special register. Data can be written to this register only in a combination of specific sequence (see **3.4.8 Special registers**).

The CKC register controls the internal system clock in the PLL mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 0AH.

After reset: 0AH R/W Address: FFFFF822H

	7	6	5	4	3	2	1	0
CKC	0	0	0	0	1	0	1	CKDIV0

CKDIV0	Internal system clock (f_{XX}) in PLL mode
0	$f_{XX} = 4 \times f_X$ (f_X = 2.5 to 5.0 MHz)
1	$f_{XX} = 8 \times f_X$ (f_X = 2.5 MHz)

Cautions 1. The PLL mode cannot be used at 5.0 MHz < $f_X \leq$ 10.0 MHz.

2. Before changing the multiplication factor between 4 and 8 by using the CKC register, set the clock-through mode and stop the PLL.

3. Be sure to set bits 3 and 1 to "1" and clear bits 7 to 4 and 2 to "0".

Remark Both the CPU clock and peripheral clock are divided by the CKC register, but only the CPU clock is divided by the PCC register.

(3) Lock register (LOCKR)

Phase lock occurs at a given frequency following power application or immediately after the STOP mode is released, and the time required for stabilization is the lockup time (frequency stabilization time). This state until stabilization is called the lockup status, and the stabilized state is called the locked status.

The LOCKR register includes a LOCK bit that reflects the PLL frequency stabilization status.

This register is read-only, in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R Address: FFFFF824H

	7	6	5	4	3	2	1	<0>
LOCKR	0	0	0	0	0	0	0	LOCK

LOCK	PLL lock status check
0	Locked status
1	Unlocked status

Caution The LOCK register does not reflect the lock status of the PLL in real time. The set/clear conditions are as follows.

[Set conditions]

- Upon system reset[Note]
- In IDLE2 or STOP mode
- Upon setting of PLL stop (clearing of PLLCTL.PLLON bit to 0)
- Upon stopping main clock and using CPU with subclock (setting of PCC.CK3 bit to 1 and setting of PCC.MCK bit to 1)

Note This register is set to 01H by reset and cleared to 00H after the reset has been released and the oscillation stabilization time has elapsed.

[Clear conditions]

- Upon overflow of oscillation stabilization time following reset release (OSTS register default time (see **21.2 (3) Oscillation stabilization time select register (OSTS)**))
- Upon oscillation stabilization timer overflow (time set by OSTS register) following STOP mode release, when the STOP mode was set in the PLL operating status
- Upon PLL lockup time timer overflow (time set by PLLS register) when the PLLCTL.PLLON bit is changed from 0 to 1
- After the setup time inserted upon release of the IDLE2 mode is released (time set by the OSTS register) when the IDLE2 mode is set during PLL operation.

(4) PLL lockup time specification register (PLLS)

The PLLS register is an 8-bit register used to select the PLL lockup time when the PLLCTL.PLLON bit is changed from 0 to 1.

This register can be read or written in 8-bit units.

Reset sets this register to 03H.

After reset: 03H R/W Address: FFFFF6C1H

	7	6	5	4	3	2	1	0
PLLS	0	0	0	0	0	0	PLLS1	PLLS0

PLLS1	PLLS0	Selection of PLL lockup time
0	0	$2^{10}/f_X$
0	1	$2^{11}f_X$
1	0	$2^{12}/f_X$
1	1	$2^{13}/f_X$ (default value)

Cautions 1. Set so that the lockup time is 800 μs or longer.

2. Do not change the PLLS register setting during the lockup period.

6.5.3 Usage

(1) When PLL is used

- After the reset signal has been released, the PLL operates (PLLCTL.PLLON bit = 1), but because the default mode is the clock-through mode (PLLCTL.SELPLL bit = 0), select the PLL mode (SELPLL bit = 1).
- To enable PLL operation, first set the PLLON bit to 1, and then set the SELPLL bit to 1 after the LOCKR.LOCK bit = 0. To stop the PLL, first select the clock-through mode (SELPLL bit = 0), wait for 8 clocks or more, and then stop the PLL (PLLON bit = 0).
- The PLL stops during transition to the IDLE2 or STOP mode regardless of the setting and is restored from the IDLE2 or STOP mode to the status before transition. The time required for restoration is as follows.

(a) When transiting to the IDLE2 or STOP mode from the clock through mode
- STOP mode: Set the OSTS register so that the oscillation stabilization time is 1 ms (min.) or longer.
- IDLE2 mode: Set the OSTS register so that the setup time is 350 μs (min.) or longer.

<R> (b) When transiting to the IDLE 2 or STOP mode while remaining in the PLL operation mode
- STOP mode: Set the OSTS register so that the oscillation stabilization time is 1 ms (min.) or longer.
- IDLE2 mode: Set the OSTS register so that the setup time is 800 μs (min.) or longer.

When transiting to the IDLE1 mode, the PLL does not stop. Stop the PLL if necessary.

(2) When PLL is not used

- The clock-through mode (SELPLL bit = 0) is selected after the reset signal has been released, but the PLL is operating (PLLON bit = 1) and must therefore be stopped (PLLON bit = 0).

CHAPTER 7 16-BIT TIMER/EVENT COUNTER P (TMP)

Timer P (TMP) is a 16-bit timer/event counter.

<R> The μPD703249Y and 703252Y have TMP0 to TMP4. The μPD703253Y has TMP0 to TMP5.

7.1 Overview

An outline of TMPn is shown below.

- Clock selection: 8 ways
- Capture/trigger input pins: 2
- External event count input pins: 1
- External trigger input pins: 1
- Timer/counters: 1
- Capture/compare registers: 2
- Capture/compare match interrupt request signals: 2
- <R> Overflow interrupt request signals: 1
- Timer output pins: 2

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

7.2 Functions

TMPn has the following functions.

- Interval timer
- External event counter
- External trigger pulse output
- One-shot pulse output
- PWM output
- Free-running timer
- Pulse width measurement

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

7.3 Configuration

TMPn includes the following hardware.

Table 7-1. Configuration of TMPn

Item	Configuration
Timer register	16-bit counter × 1 each
Registers	TMPn capture/compare registers 0, 1 (TPnCCR0, TPnCCR1) TMPn counter read buffer register (TPnCNT) CCR0, CCR1 buffer registers
Timer inputs	2 (TIPn0[Note 1], TIPn1 pins)
Timer outputs	2 (TOPn0, TOPn1 pins)
Control registers[Note 2]	TMPn control registers 0, 1 (TPnCTL0, TPnCTL1) TMPn I/O control registers 0 to 2 (TPnIOC0 to TPnIOC2) TMPn option register 0 (TPnOPT0)

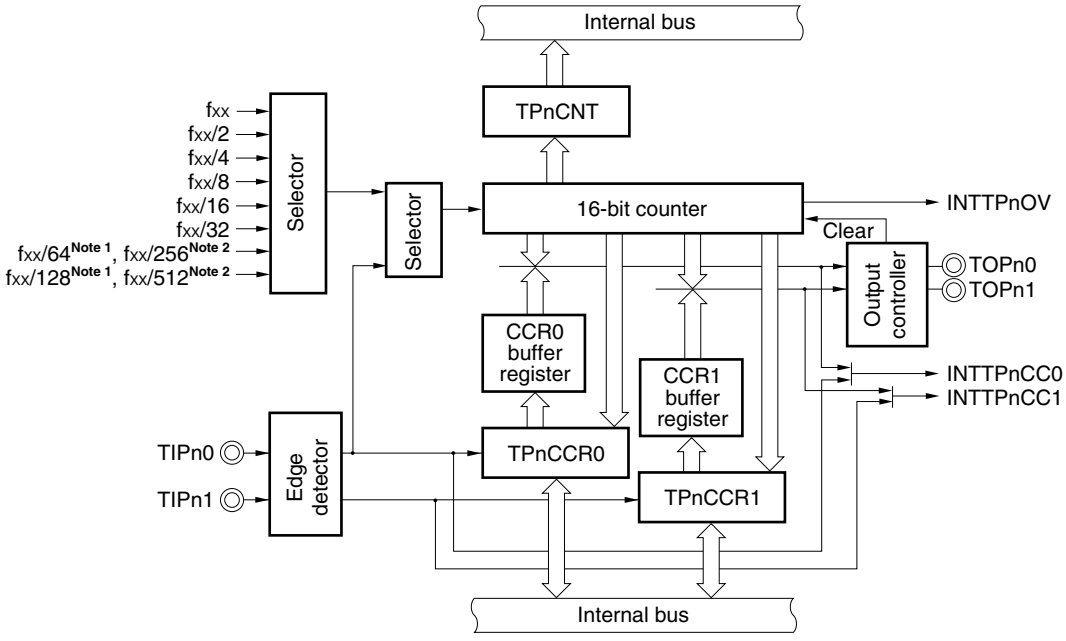
Notes 1. The TIPn0 pin functions alternately as a capture trigger input signal, external event count input signal, and external trigger input signal.

2. When using the functions of the TIPn0, TIPn1, TOPn0, and TOPn1 pins, see **Table 4-15 Using Port Pin as Alternate-Function Pin**.

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Figure 7-1. Block Diagram of TMPn



Notes 1. TMP0, TMP2, TMP4

<R> **2.** TMP1, TMP3, TMP5 (μPD703253Y only)

Remark fxx: Main clock frequency

(1) 16-bit counter

This 16-bit counter can count internal clocks or external events.

The count value of this counter can be read by using the TPnCNT register.

When the TPnCTL0.TPnCE bit = 0, the value of the 16-bit counter is FFFFH. If the TPnCNT register is read at this time, 0000H is read.

Reset sets the TPnCE bit to 0.

(2) CCR0 buffer register

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TPnCCR0 register is used as a compare register, the value written to the TPnCCR0 register is transferred to the CCR0 buffer register. When the count value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTPnCC0) is generated.

The CCR0 buffer register cannot be read or written directly.

The CCR0 buffer register is cleared to 0000H after reset, as the TPnCCR0 register is cleared to 0000H.

(3) CCR1 buffer register

This is a 16-bit compare register that compares the count value of the 16-bit counter.

When the TPnCCR1 register is used as a compare register, the value written to the TPnCCR1 register is transferred to the CCR1 buffer register. When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTPnCC1) is generated.

The CCR1 buffer register cannot be read or written directly.

The CCR1 buffer register is cleared to 0000H after reset, as the TPnCCR1 register is cleared to 0000H.

(4) Edge detector

This circuit detects the valid edges input to the TIPn0 and TIPn1 pins. No edge, rising edge, falling edge, or both the rising and falling edges can be selected as the valid edge by using the TPnIOC1 and TPnIOC2 registers.

(5) Output controller

This circuit controls the output of the TOPn0 and TOPn1 pins. The output controller is controlled by the TPnIOC0 register.

(6) Selector

This selector selects the count clock for the 16-bit counter. Eight types of internal clocks or an external event can be selected as the count clock.

7.4 Registers

The registers that control TMPn are as follows.

- TMPn control register 0 (TPnCTL0)
- TMPn control register 1 (TPnCTL1)
- TMPn I/O control register 0 (TPnIOC0)
- TMPn I/O control register 1 (TPnIOC1)
- TMPn I/O control register 2 (TPnIOC2)
- TMPn option register 0 (TPnOPT0)
- TMPn capture/compare register 0 (TPnCCR0)
- TMPn capture/compare register 1 (TPnCCR1)
- TMPn counter read buffer register (TPnCNT)

Remarks 1. When using the functions of the TIPn0, TIPn1,TOPn0, and TOPn1 pins, see **Table 4-15 Using Port Pin as Alternate-Function Pin**.

2. μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

(1) TMPn control register 0 (TPnCTL0)

The TPnCTL0 register is an 8-bit register that controls the operation of TMPn.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

The same value can always be written to the TPnCTL0 register by software.

After reset: 00H R/W Address: TP0CTL0 FFFFF590H, TP1CTL0 FFFFF5A0H,
TP2CTL0 FFFFF5B0H, TP3CTL0 FFFFF5C0H,
<R> TP4CTL0 FFFFF5D0H, TP5CTL0 FFFFF5E0H[Note 1]

	<7>	6	5	4	3	2	1	0
TPnCTL0	TPnCE	0	0	0	0	TPnCKS2	TPnCKS1	TPnCKS0

[μPD703249Y, μPD703252Y n = 0 to 4]

TPnCE	TMPn operation control
0	TMPn operation disabled (TMPn reset asynchronously[Note 2]).
1	TMPn operation enabled. TMPn operation started.

<R> [μPD703253Y n = 0 to 5]

<R>

TPnCKS2	TPnCKS1	TPnCKS0	Internal count clock selection	
			n = 0, 2, 4	n = 1, 3, 5[Note 1]
0	0	0	f_{XX}	
0	0	1	$f_{XX}/2$	
0	1	0	$f_{XX}/4$	
0	1	1	$f_{XX}/8$	
1	0	0	$f_{XX}/16$	
1	0	1	$f_{XX}/32$	
1	1	0	$f_{XX}/64$	$f_{XX}/256$
1	1	1	$f_{XX}/128$	$f_{XX}/512$

<R> **Notes 1.** μPD703253Y only

<R> **2.** The TPnOPT0.TPnOVF bit and 16-bit counter are reset at the same time. In addition, the timer output pins (TOPn0 and TOPn1 pins) are reset to the status set by the TPnIOC0 register when the 16-bit counter is reset.

Cautions 1. Set the TPnCKS2 to TPnCKS0 bits when the TPnCE bit = 0.
When the value of the TPnCE bit is changed from 0 to 1, the TPnCKS2 to TPnCKS0 bits can be set simultaneously.
2. Be sure to clear bits 3 to 6 to "0".

Remark f_{XX}: Main clock frequency

(2) TMPn control register 1 (TPnCTL1)

The TPnCTL1 register is an 8-bit register that controls the operation of TMPn.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: TP0CTL1 FFFFF591H, TP1CTL1 FFFFF5A1H,
TP2CTL1 FFFFF5B1H, TP3CTL1 FFFFF5C1H,
<R> TP4CTL1 FFFFF5D1H, TP5CTL1 FFFFF5E1H[Note]

	7	<6>	<5>	4	3	2	1	0
TPnCTL1	0	TPnEST	TPnEEE	0	0	TPnMD2	TPnMD1	TPnMD0

[μPD703249Y, μPD703252Y n = 0 to 4]

TPnEST	Software trigger control
0	–
1	Generate a valid signal for external trigger input. • In one-shot pulse output mode: A one-shot pulse is output with writing 1 to the TPnEST bit as the trigger. • In external trigger pulse output mode: A PWM waveform is output with writing 1 to the TPnEST bit as the trigger.
The read value of the TPnEST bit is always 0.	

<R> [μPD703253Y n = 0 to 5]

<R>

TPnEEE	Count clock selection
0	Disable operation with external event count input (TIPn0 pin). (Perform counting with the count clock selected by the TPnCTL0.TPnCKS0 to TPnCKS2 bits.)
1	Enable operation with external event count input (TIPn0 pin). (Perform counting at the valid edge of the external event count input signal (TIPn0 pin).)
The TPnEEE bit selects whether counting is performed with the internal count clock or the valid edge of the external event count input.	

TPnMD2	TPnMD1	TPnMD0	Timer mode selection
0	0	0	Interval timer mode
0	0	1	External event count mode
0	1	0	External trigger pulse output mode
0	1	1	One-shot pulse output mode
1	0	0	PWM output mode
1	0	1	Free-running timer mode
1	1	0	Pulse width measurement mode
1	1	1	Setting prohibited

<R> **Note** μPD703253Y only

Cautions 1. The TPnEST bit is valid only in the external trigger pulse output mode or the one-shot pulse output mode. In any other mode, writing 1 to this bit is ignored.

2. External event count input is selected in the external event count mode regardless of the value of the TPnEEE bit.

3. Set the TPnEEE and TPnMD2 to TPnMD0 bits when the TPnCTL0.TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) The operation is not guaranteed when rewriting is performed with the TPnCE bit = 1. If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.

4. Be sure to clear bits 3, 4, and 7 to "0".

(3) TMPn I/O control register 0 (TPnIOC0)

The TPnIOC0 register is an 8-bit register that controls the timer output (TOPn0, TOPn1 pins).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: TP0IOC0 FFFFF592H, TP1IOC0 FFFFF5A2H,
TP2IOC0 FFFFF5B2H, TP3IOC0 FFFFF5C2H,
<R> TP4IOC0 FFFFF5D2H, TP5IOC0 FFFFF5E2H[Note 1]

	7	6	5	4	3	<2>	1	<0>
TPnIOC0	0	0	0	0	TPnOL1	TPnOE1	TPnOL0	TPnOE0

<R> (μPD703249Y, μPD703252Y n = 0 to 4)

TPnOL1	TOPn1 pin output level setting[Note 2]
0	TOPn1 pin high level start
1	TOPn1 pin low level start

<R> (μPD703253Y n = 0 to 5)

TPnOE1	TOPn1 pin output setting
0	Timer output disabled • When TPnOL1 bit = 0: Low level is output from the TOPn1 pin • When TPnOL1 bit = 1: High level is output from the TOPn1 pin
1	Timer output enabled (a pulse is output from the TOPn1 pin).

<R>

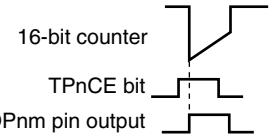
TPnOL0	TOPn0 pin output level setting[Note 2]
0	TOPn0 pin high level start
1	TOPn0 pin low level start

TPnOE0	TOPn0 pin output setting
0	Timer output disabled • When TPnOL0 bit = 0: Low level is output from the TOPn0 pin • When TPnOL0 bit = 1: High level is output from the TOPn0 pin
1	Timer output enabled (a pulse is output from the TOPn0 pin).

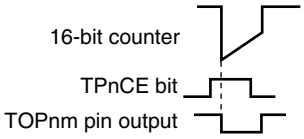
<R> **Notes 1.** μPD703253Y only

2. The output level of the timer output pins (TOPn0, TOPn1) specified by the TPnOLm bit is shown below (m = 0, 1).

• When TPnOLm bit = 0



• When TPnOLm bit = 1



<R> **Cautions 1. The pin output changes if the setting of the TPnIOC0 register is rewritten when the port is set to output TOPn0 and TOPn1. Therefore, note changes in the pin status by setting the port to the input mode and making the output status of the pins a high-impedance state.**

2. Rewrite the TPnOL1, TPnOE1, TPnOL0, and TPnOE0 bits when the TPnCTL0.TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.

3. Even if the TPnOLm bit is manipulated when the TPnCE and TPnOEm bits are 0, the TOPnm pin output level varies (m = 0, 1).

(4) TMPn I/O control register 1 (TPnIOC1)

The TPnIOC1 register is an 8-bit register that controls the valid edge of the capture trigger input signals (TIPn0, TIPn1 pins).

<R> This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: TP0IOC1 FFFFF593H, TP1IOC1 FFFFF5A3H,
TP2IOC1 FFFFF5B3H, TP3IOC1 FFFFF5C3H,
<R> TP4IOC1 FFFFF5D3H, TP5IOC1 FFFFF5E3H[Note]

	7	6	5	4	3	2	1	0
TPnIOC1	0	0	0	0	TPnIS3	TPnIS2	TPnIS1	TPnIS0

(μPD703249Y, μPD703252Y n = 0 to 4)

TPnIS3	TPnIS2	Capture trigger input signal (TIPn1 pin) valid edge setting
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

<R> (μPD703253Y n = 0 to 5)

TPnIS1	TPnIS0	Capture trigger input signal (TIPn0 pin) valid edge setting
0	0	No edge detection (capture operation invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

<R> **Note** μPD703253Y only

Cautions 1. Rewrite the TPnIS3 to TPnIS0 bits when the TPnCTL0.TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.

2. The TPnIS3 to TPnIS0 bits are valid only in the free-running timer mode (only when TPnOPT0.TPnCCS1, TPnCCS0 bits = 11) and the pulse width measurement mode. In all other modes, a capture operation is not possible.

(5) TMPn I/O control register 2 (TPnIOC2)

The TPnIOC2 register is an 8-bit register that controls the valid edge of the external event count input signal (TIPn0 pin) and external trigger input signal (TIPn0 pin).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: TP0IOC2 FFFFF594H, TP1IOC2 FFFFF5A4H,
TP2IOC2 FFFFF5B4H, TP3IOC2 FFFFF5C4H,
<R> TP4IOC2 FFFFF5D4H, TP5IOC2 FFFFF5E4H[Note]

	7	6	5	4	3	2	1	0
TPnIOC2	0	0	0	0	TPnEES1	TPnEES0	TPnETS1	TPnETS0

[μPD703249Y, μPD703252Y n = 0 to 4]

TPnEES1	TPnEES0	External event count input signal (TIPn0 pin) valid edge setting
0	0	No edge detection (external event count invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

<R> [μPD703253Y n = 0 to 5]

TPnETS1	TPnETS0	External trigger input signal (TIPn0 pin) valid edge setting
0	0	No edge detection (external trigger invalid)
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

<R> **Note** μPD703253Y only

Cautions 1. Rewrite the TPnEES1, TPnEES0, TPnETS1, and TPnETS0 bits when the TPnCTL0.TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.

2. The TPnEES1 and TPnEES0 bits are valid only when the TPnCTL1.TPnEEE bit = 1 or when the external event count mode (TPnCTL1.TPnMD2 to TPnCTL1.TPnMD0 bits = 001) has been set.

3. The TPnETS1 and TPnETS0 bits are valid only when the external trigger pulse output mode (TPnCTL1.TPnMD2 to TPnCTL1.TPnMD0 bits = 010) or the one-shot pulse output mode (TPnCTL1.TPnMD2 to TPnCTL1.TPnMD0 = 011) is set.

(6) TMPn option register 0 (TPnOPT0)

The TPnOPT0 register is an 8-bit register used to set the capture/compare operation and detect an overflow.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: TP0OPT0 FFFFF595H, TP1OPT0 FFFFF5A5H,
TP2OPT0 FFFFF5B5H, TP3OPT0 FFFFF5C5H,
<R> TP4OPT0 FFFFF5D5H, TP5OPT0 FFFFF5E5H[Note]

	7	6	5	4	3	2	1	<0>
TPnOPT0	0	0	TPnCCS1	TPnCCS0	0	0	0	TPnOVF

(μPD703249Y, μPD703252Y n = 0 to 4)

TPnCCS1	TPnCCR1 register capture/compare selection
0	Compare register selected
1	Capture register selected (cleared by setting TPnCTL0.TPnCE bit = 0)
The TPnCCS1 bit setting is valid only in the free-running timer mode.	

<R> <R> (μPD703253Y n = 0 to 5)

TPnCCS0	TPnCCR0 register capture/compare selection
0	Compare register selected
1	Capture register selected (cleared by setting TPnCTL0.TPnCE bit = 0)
The TPnCCS0 bit setting is valid only in the free-running timer mode.	

<R>

TPnOVF	TMPn overflow detection flag
Set (1)	Overflow occurred
Reset (0)	TPnOVF bit 0 written or TPnCTL0.TPnCE bit = 0

- The TPnOVF bit is set to 1 when the 16-bit counter count value overflows from FFFFH to 0000H in the free-running timer mode or the pulse width measurement mode.
- An overflow interrupt request signal (INTTPnOV) is generated at the same time that the TPnOVF bit is set to 1. The INTTPnOV signal is not generated in modes other than the free-running timer mode and the pulse width measurement mode.
- The TPnOVF bit is not cleared to 0 even when the TPnOVF bit or the TPnOPT0 register are read when the TPnOVF bit = 1.
- Before clearing the TPnOVF bit to 0 after the INTTPnOV signal has been generated, be sure to confirm (read) that the TPnOVF bit is set to 1.
- The TPnOVF bit can be both read and written, but the TPnOVF bit cannot be set to 1 by software. Writing 1 has no influence on the operation of TMPn.

<R> **Note** μPD703253Y only

Cautions 1. Rewrite the TPnCCS1 and TPnCCS0 bits when the TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.

2. Be sure to clear bits 1 to 3, 6, and 7 to "0".

(7) TMPn capture/compare register 0 (TPnCCR0)

The TPnCCR0 register is a 16-bit register that can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, depending on the setting of the TPnOPT0.TPnCCS0 bit. In the pulse width measurement mode, the TPnCCR0 register can be used only as a capture register. In any other mode, this register can be used only as a compare register.

The TPnCCR0 register can be read or written during operation.

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

Reset sets this register to 0000H.

Caution Accessing the TPnCCR0 register is prohibited in the following statuses. For details, see 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset: 0000H R/W Address: TP0CCR0 FFFFF596H, TP1CCR0 FFFFF5A6H,
TP2CCR0 FFFFF5B6H, TP3CCR0 FFFFF5C6H,
<R> TP4CCR0 FFFFF5D6H, TP5CCR0 FFFFF5E6H[Note]

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPnCCR0																

(μPD703249Y, μPD703252Y: n = 0 to 4)

<R> (μPD703253Y: n = 0 to 5)

<R> **Note** μPD703253Y only

(a) Function as compare register

The TPnCCR0 register can be rewritten even when the TPnCTL0.TPnCE bit = 1.

The set value of the TPnCCR0 register is transferred to the CCR0 buffer register. When the value of the 16-bit counter matches the value of the CCR0 buffer register, a compare match interrupt request signal (INTTPnCC0) is generated. If TOPn0 pin output is enabled at this time, the output of the TOPn0 pin is inverted.

When the TPnCCR0 register is used as a cycle register in the interval timer mode, external event count mode, external trigger pulse output mode, one-shot pulse output mode, or PWM output mode, the value of the 16-bit counter is cleared (0000H) if its count value matches the value of the CCR0 buffer register.

<R> The compare register is not cleared when the TPnCTL0.TPnCE bit = 0.

(b) Function as capture register

When the TPnCCR0 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TPnCCR0 register if the valid edge of the capture trigger input pin (TIPn0 pin) is detected. In the pulse-width measurement mode, the count value of the 16-bit counter is stored in the TPnCCR0 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIPn0) is detected.

Even if the capture operation and reading the TPnCCR0 register conflict, the correct value of the TPnCCR0 register can be read.

<R> The capture register is cleared when the TPnCTL0.TPnCE bit = 0.

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

Table 7-2. Function of Capture/Compare Register in Each Mode and How to Write Compare Register

Operation Mode	Capture/Compare Register	How to Write Compare Register
Interval timer	Compare register	Anytime write
External event counter	Compare register	Anytime write
<R> External trigger pulse output	Compare register	Batch write[Note]
One-shot pulse output	Compare register	Anytime write
<R> PWM output	Compare register	Batch write[Note]
Free-running timer	Capture/compare register	Anytime write
Pulse width measurement	Capture register	None

<R> **Note** Triggered by writing to the TPnCCR1 register

<R> **Remark** For details of anytime write and batch write, see **7.6 (2) Anytime write and batch write**.

(8) TMPn capture/compare register 1 (TPnCCR1)

The TPnCCR1 register is a 16-bit register that can be used as a capture register or a compare register depending on the mode.

This register can be used as a capture register or a compare register only in the free-running timer mode, depending on the setting of the TPnOPT0.TPnCCS1 bit. In the pulse width measurement mode, the TPnCCR1 register can be used only as a capture register. In any other mode, this register can be used only as a compare register.

The TPnCCR1 register can be read or written during operation.

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

Reset sets this register to 0000H.

Caution Accessing the TPnCCR1 register is prohibited in the following statuses. For details, see 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

<R> After reset: 0000H R/W Address: TP0CCR1 FFFFF598H, TP1CCR1 FFFFF5A8H, TP2CCR1 FFFFF5B8H, TP3CCR1 FFFFF5C8H, TP4CCR1 FFFFF5D8H, TP5CCR1 FFFFF5E8H[Note]

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPnCCR1																

[μPD703249Y, μPD703252Y n = 0 to 4]

<R> [μPD703253Y n = 0 to 5]

<R> **Note** μPD703253Y only

(a) Function as compare register

The TPnCCR1 register can be rewritten even when the TPnCTL0.TPnCE bit = 1.

The set value of the TPnCCR1 register is transferred to the CCR1 buffer register. When the value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTPnCC1) is generated. If TOPn1 pin output is enabled at this time, the output of the TOPn1 pin is inverted.

<R> The compare register is not cleared when the TPnCTL0.TPnCE bit = 0.

(b) Function as capture register

When the TPnCCR1 register is used as a capture register in the free-running timer mode, the count value of the 16-bit counter is stored in the TPnCCR1 register if the valid edge of the capture trigger input pin (TIPn1 pin) is detected. In the pulse-width measurement mode, the count value of the 16-bit counter is stored in the TPnCCR1 register and the 16-bit counter is cleared (0000H) if the valid edge of the capture trigger input pin (TIPn1) is detected.

Even if the capture operation and reading the TPnCCR1 register conflict, the correct value of the TPnCCR1 register can be read.

<R> The capture register is cleared when the TPnCTL0.TPnCE bit = 0.

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

The following table shows the functions of the capture/compare register in each mode, and how to write data to the compare register.

Table 7-3. Function of Capture/Compare Register in Each Mode and How to Write Compare Register

Operation Mode	Capture/Compare Register	How to Write Compare Register
Interval timer	Compare register	Anytime write
External event counter	Compare register	Anytime write
<R> External trigger pulse output	Compare register	Batch write[Note]
One-shot pulse output	Compare register	Anytime write
<R> PWM output	Compare register	Batch write[Note]
Free-running timer	Capture/compare register	Anytime write
Pulse width measurement	Capture register	None

<R> **Note** Triggered by writing to the TPnCCR1 register

<R> **Remark** For anytime write and batch write, see **7.6 (2) Anytime write and batch write**.

(9) TMPn counter read buffer register (TPnCNT)

The TPnCNT register is a read buffer register that can read the count value of the 16-bit counter.

If this register is read when the TPnCTL0.TPnCE bit = 1, the count value of the 16-bit timer can be read.

This register is read-only, in 16-bit units.

The value of the TPnCNT register is cleared to 0000H when the TPnCE bit = 0. If the TPnCNT register is read at this time, the value of the 16-bit counter (FFFFH) is not read, but 0000H is read.

The value of the TPnCNT register is cleared to 0000H after reset, as the TPnCE bit is cleared to 0.

Caution Accessing the TPnCNT register is prohibited in the following statuses. For details, see 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset: 0000H R Address: TP0CNT FFFFF59AH, TP1CNT FFFFF5AAH,
TP2CNT FFFFF5BAH, TP3CNT FFFFF5CAH,
<R> TP4CNT FFFFF5DAH, TP5CNT FFFFF5EAH[Note]

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPnCNT																

[μPD703249Y, μPD703252Y n = 0 to 4]

<R> [μPD703253Y n = 0 to 5]

<R> **Note** μPD703253Y only

7.5 Timer Output Operations

The following table shows the operations and output levels of the TOPn0 and TOPn1 pins.

Table 7-4. Timer Output Control in Each Mode

Operation Mode	TOPn1 Pin	TOPn0 Pin
Interval timer mode	Square wave output	
<R> External event count mode	None	
External trigger pulse output mode	External trigger pulse output	Square wave output
One-shot pulse output mode	One-shot pulse output	
PWM output mode	PWM output	
Free-running timer mode	Square wave output (only when compare function is used)	
Pulse width measurement mode	None	

Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

Table 7-5. Truth Table of TOPn0 and TOPn1 Pins Under Control of Timer Output Control Bits

TPnIOC0.TPnOLm Bit	TPnIOC0.TPnOEm Bit	TPnCTL0.TPnCE Bit	Level of TOPnm Pin
0	0	×	Low-level output
	1	0	Low-level output
		1	Low level immediately before counting, high level after counting is started
1	0	×	High-level output
	1	0	High-level output
		1	High level immediately before counting, low level after counting is started

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1
<R> μPD703253Y: n = 0 to 5, m = 0 or 1

7.6 Operation

TMPn can perform the following operations.

Operation	TPnCTL1.TPnEST Bit (Software Trigger Bit)	TIPn0 Pin (External Trigger Input)	Capture/Compare Register Setting	Compare Register Write
Interval timer mode	Invalid	Invalid	Compare only	Anytime write
External event count mode[Note 1]	Invalid	Invalid	Compare only	Anytime write
External trigger pulse output mode[Note 2]	Valid	Valid	Compare only	Batch write
One-shot pulse output mode[Note 2]	Valid	Valid	Compare only	Anytime write
PWM output mode	Invalid	Invalid	Compare only	Batch write
Free-running timer mode	Invalid	Invalid	Switching enabled	Anytime write
Pulse width measurement mode[Note 2]	Invalid	Invalid	Capture only	Not applicable

Notes 1. To use the external event count mode, specify that the valid edge of the TIPn0 pin capture trigger input is not detected (by clearing the TPnIOC1.TPnIS1 and TPnIOC1.TPnIS0 bits to “00”).

2. To use the external trigger pulse output mode, one-shot pulse output mode, or pulse width measurement mode, select the internal clock (by setting the TPnCTL1.TPnEEE bit to 0) as the count clock.

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

<R> **(1) Counter basic operation**

This section explains the basic operation of the 16-bit counter. For details, refer to the description of the operation in each mode.

Remark μPD703249Y and 703252Y: n = 0 to 4
μPD703253Y: n = 0 to 5

(a) Counter start operation

The 16-bit counter of TMPn starts counting from the default value FFFFH in all modes.
It counts up from FFFFH to 0000H, 0001H, 0002H, 0003H, and so on.

(b) Clear operation

The 16-bit counter is cleared to 0000H when its value matches the value of the compare register and is cleared, and when its value is captured and cleared. The counting operation from FFFFH to 0000H that takes place immediately after the counter has started counting or when the counter overflows is not a clearing operation. Therefore, the INTTPnCC0 and INTTPnCC1 interrupt signals are not generated.

(c) Overflow operation

The 16-bit counter overflows when the counter counts up from FFFFH to 0000H in the free-running timer mode or pulse width measurement mode. If the counter overflows, the TPnOPT0.TPnOVF bit is set to 1 and an interrupt request signal (INTTPnOV) is generated. Note that the INTTPnOV signal is not generated under the following conditions.

- Immediately after a counting operation has been started
- If the counter value matches the compare value FFFFH and is cleared
- When FFFFH is captured and cleared in the pulse width measurement mode and the counter counts up from FFFFH to 0000H

Caution After the overflow interrupt request signal (INTTPnOV) has been generated, be sure to check that the overflow flag (TPnOVF bit) is set to 1.

(d) Counter read operation during counting operation

The value of the 16-bit counter of TMPn can be read by using the TPnCNT register during the count operation. When the TPnCTL0.TPnCE bit = 1, the value of the 16-bit counter can be read by reading the TPnCNT register. When the TPnCTL0.TPnCE bit = 0, the 16-bit counter is FFFFH and the TPnCNT register is 0000H.

(e) Interrupt operation

TMPn generates the following three types of interrupt request signals.

- INTTPnCC0 interrupt: This signal functions as a match interrupt request signal of the CCR0 buffer register and as a capture interrupt request signal to the TPnCCR0 register.
- INTTPnCC1 interrupt: This signal functions as a match interrupt request signal of the CCR1 buffer register and as a capture interrupt request signal to the TPnCCR1 register.
- INTTPnOV interrupt: This signal functions as an overflow interrupt request signal.

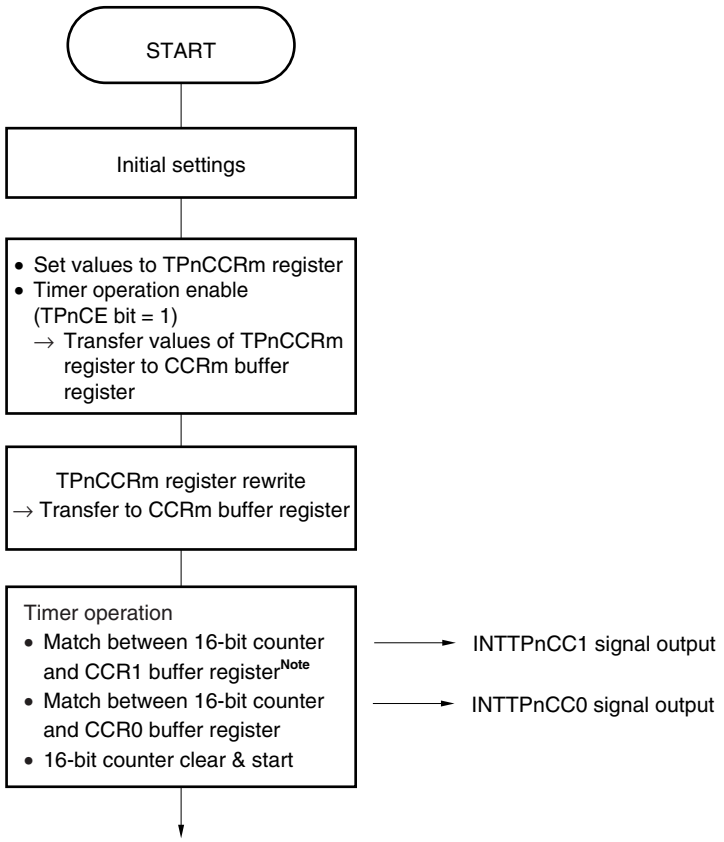
<R> **(2) Anytime write and batch write**

The TPnCCR0 and TPnCCR1 registers in TMPn can be rewritten during timer operation (TPnCTL0.TPnCE bit = 1), but the write method (anytime write, batch write) of the CCR0 and CCR1 buffer registers differs depending on the mode.

(a) Anytime write

In this mode, data is transferred at any time from the TPnCCR0 and TPnCCR1 registers to the CCR0 and CCR1 buffer registers during timer operation. (μPD703249Y and 703252Y: n = 0 to 4, μPD703253Y: n = 0 to 5)

Figure 7-2. Flowchart of Basic Operation for Anytime Write

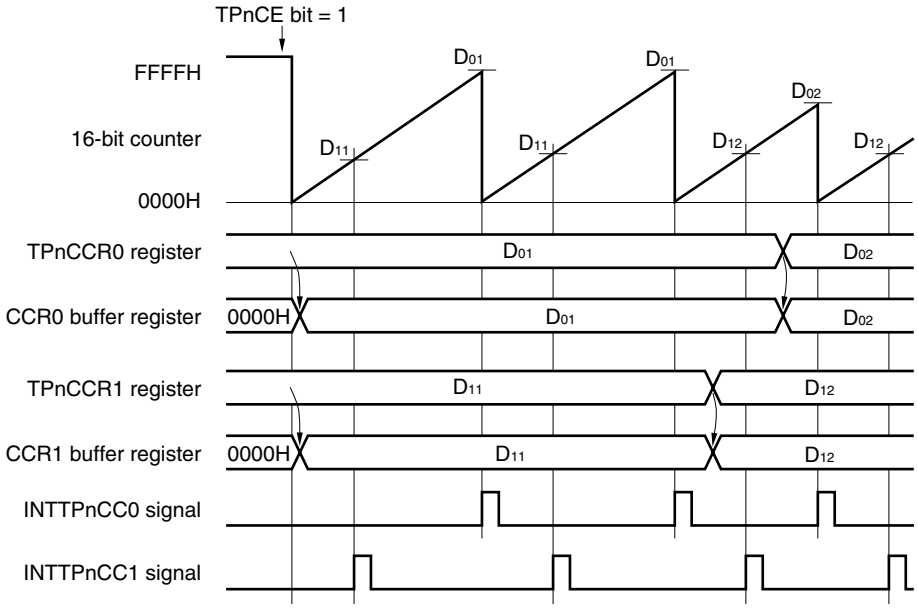


Note The 16-bit counter is not cleared upon a match between the 16-bit counter value and the CCR1 buffer register value. It is cleared upon a match between the 16-bit counter value and the CCR0 buffer register value.

Remarks 1. The above flowchart illustrates an example of the operation in the interval timer mode.

2. μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1
μPD703253Y: n = 0 to 5, m = 0 or 1

Figure 7-3. Timing of Anytime Write



Remarks 1. D_{01}, D_{02}: Setting values of TPnCCR0 register

D_{11}, D_{12}: Setting values of TPnCCR1 register

2. The above timing chart illustrates an example of the operation in the interval timer mode.

3. μPD703249Y and 703252Y: n = 0 to 4

μPD703253Y: n = 0 to 5

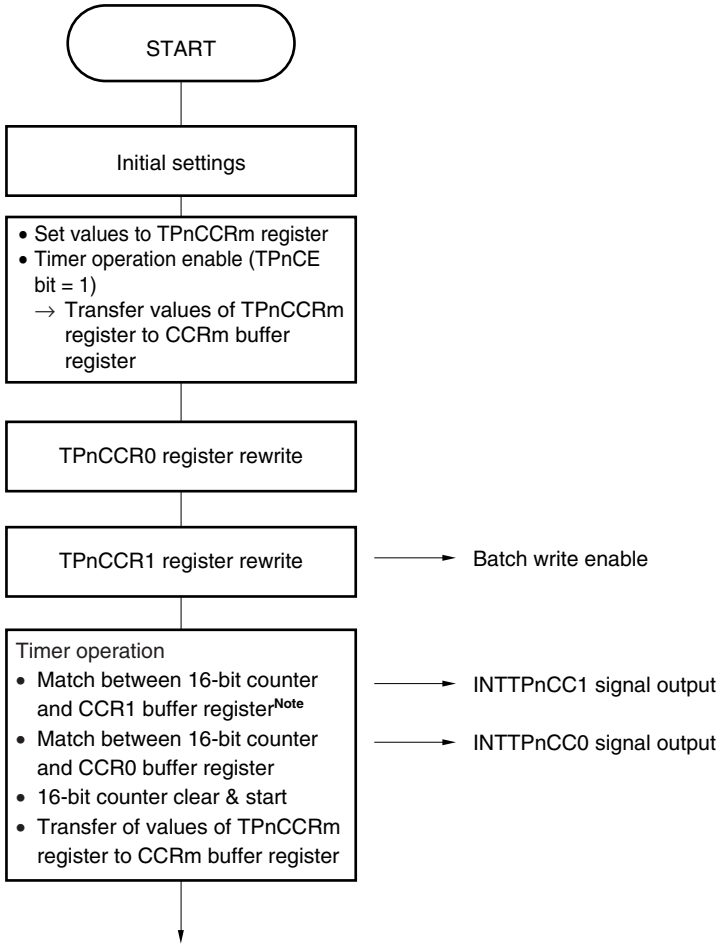
(b) Batch write

In this mode, data is transferred all at once from the TPnCCR0 and TPnCCR1 registers to the CCR0 and CCR1 buffer registers during timer operation. This data is transferred upon a match between the value of the CCR0 buffer register and the value of the 16-bit counter. Transfer is enabled by writing to the TPnCCR1 register.

Whether to enable or disable the next transfer timing is controlled by writing or not writing to the TPnCCR1 register.

In order for the setting value when the TPnCCR0 and TPnCCR1 registers are rewritten to become the 16-bit counter comparison value (in other words, in order for this value to be transferred to the CCR0 and CCR1 buffer registers), it is necessary to rewrite the TPnCCR0 register and then write to the TPnCCR1 register before the 16-bit counter value and the CCR0 buffer register value match. Therefore, the values of the TPnCCR0 and TPnCCR1 registers are transferred to the CCR0 and CCR1 buffer registers upon a match between the count value of the 16-bit counter and the value of the CCR0 buffer register. Thus even when wishing only to rewrite the value of the TPnCCR0 register, also write the same value (same as preset value of the TPnCCR1 register) to the TPnCCR1 register.

Figure 7-4. Flowchart of Basic Operation for Batch Write



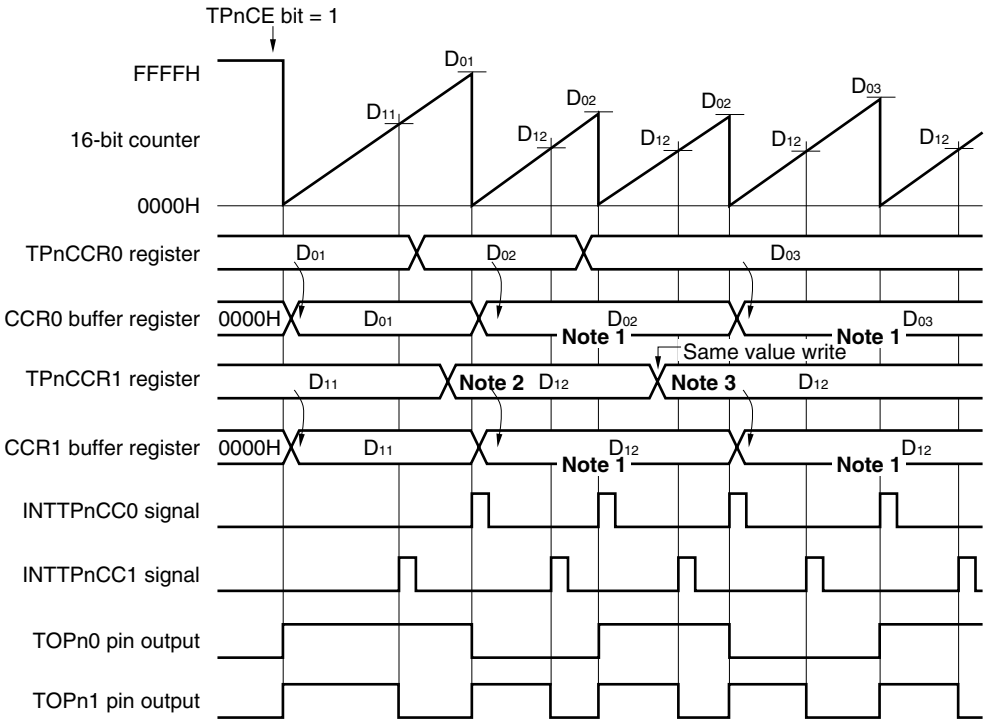
Note The 16-bit counter is not cleared upon a match between the 16-bit counter value and the CCR1 buffer register value. It is cleared upon a match between the 16-bit counter value and the CCR0 buffer register value.

Caution Writing to the TPnCCR1 register includes enabling of batch write. Thus, rewrite the TPnCCR1 register after rewriting the TPnCCR0 register.

Remarks 1. The above flowchart illustrates an example of the operation in the PWM output mode.

2. μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1
μPD703253Y: n = 0 to 5, m = 0 or 1

Figure 7-5. Timing of Batch Write



Notes 1. Because the TPnCCR1 register was not rewritten, D_{03} is not transferred.

2. Because the TPnCCR1 register has been written (D_{12}), data is transferred to the CCR1 buffer register upon a match between the value of the 16-bit counter and the value of the TPnCCR0 register (D_{01}).

3. Because the TPnCCR1 register has been written (D_{12}), data is transferred to the CCR1 buffer register upon a match between the value of the 16-bit counter and the value of the TPnCCR0 register (D_{02}).

Remarks 1. D_{01}, D_{02}, D_{03}: Setting values of TPnCCR0 register
D_{11}, D_{12}: Setting values of TPnCCR1 register

2. The above timing chart illustrates the operation in the PWM output mode as an example.

3. μPD703249Y and 703252Y: n = 0 to 4
μPD703253Y: n = 0 to 5

<R>

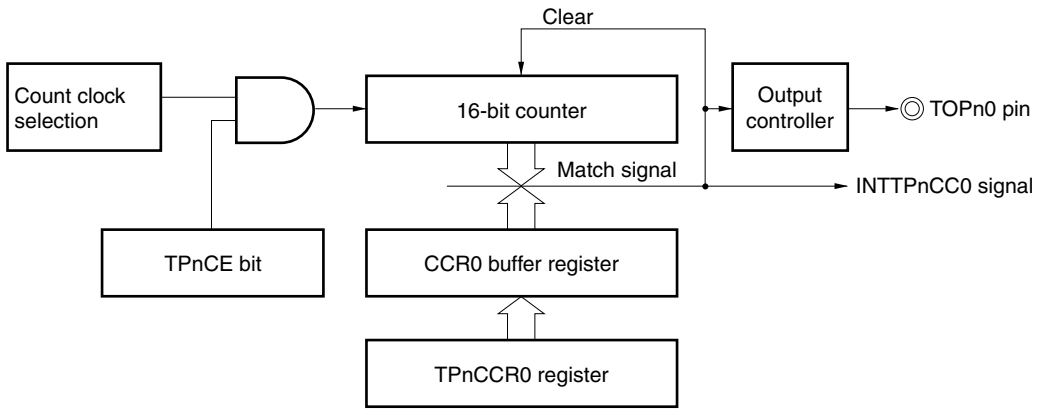
7.6.1 Interval timer mode (TPnMD2 to TPnMD0 bits = 000)

In the interval timer mode, an interrupt request signal (INTTPnCC0) is generated at the interval set by the TPnCCR0 register if the TPnCTL0.TPnCE bit is set to 1. A square wave with a duty factor of 50% whose half cycle is equal to the interval can be output from the TOPn0 pin.

The TPnCCR1 register is not used in the interval timer mode. However, the set value of the TPnCCR1 register is transferred to the CCR1 buffer register, and when the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTPnCC1) is generated. In addition, a square wave with a duty factor of 50%, which is inverted when the INTTPnCC1 signal is generated, can be output from the TOPn1 pin.

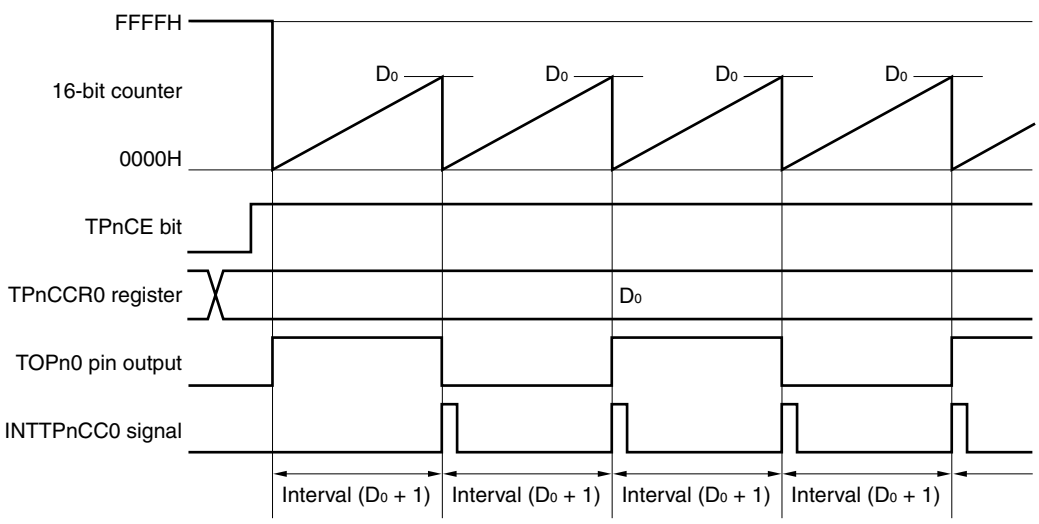
The value of the TPnCCR0 and TPnCCR1 registers can be rewritten even while the timer is operating.

Figure 7-6. Configuration of Interval Timer



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

Figure 7-7. Basic Timing of Operation in Interval Timer Mode



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

When the TPnCE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H in synchronization with the count clock, and the counter starts counting. At this time, the output of the TOPn0 pin is inverted. Additionally, the set value of the TPnCCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, the output of the TOPn0 pin is inverted, and a compare match interrupt request signal (INTTPnCC0) is generated.

The interval can be calculated by the following expression.

Interval = (Set value of TPnCCR0 register + 1) × Count clock cycle

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Figure 7-8. Register Setting for Interval Timer Mode Operation (1/3)

(a) TMPn control register 0 (TPnCTL0)

	TPnCE					TPnCKS2	TPnCKS1	TPnCKS0
TPnCTL0	0/1	0	0	0	0	0/1	0/1	0/1

Select count clock

0: Stop counting
1: Enable counting

(b) TMPn control register 1 (TPnCTL1)

		TPnEST	TPnEEE			TPnMD2	TPnMD1	TPnMD0
TPnCTL1	0	0	0/1[Note]	0	0	0	0	0

0, 0, 0:
Interval timer mode

0: Operate on count clock selected by TPnCKS0 to TPnCKS2 bits
1: Count with external event count input signal

<R> **Note** The TPnEEE bit can be set to 1 only when the timer output (TOPn1) is used. However, set the TPnCCR0 and TPnCCR1 registers to the same value.

Figure 7-8. Register Setting for Interval Timer Mode Operation (2/3)

(c) TMPn I/O control register 0 (TPnIOC0)

	TPnOL1	TPnOE1	TPnOL0	TPnOE0
TPnIOC0	0/1	0/1	0/1	0/1

- TPnOE0: 0: Disable TOPn0 pin output; 1: Enable TOPn0 pin output
- TPnOL0: Setting to output level of TOPn0 pin before count operation; 0: Low level; 1: High level
- TPnOE1: 0: Disable TOPn1 pin output; 1: Enable TOPn1 pin output
- TPnOL1: Setting to output level of TOPn1 pin before count operation; 0: Low level; 1: High level

<R> **(d) TMPn I/O control register 2 (TPnIOC2)**

	TPnEES1	TPnEES0	TPnETS1	TPnETS0
TPnIOC2	0/1 Note	0/1 Note	0	0

- TPnEES1, TPnEES0: Select valid edge of external event count input (TIPn0 pin).

Note The TPnEES1 and TPnEES0 bits can be set only when the timer output (TOPn1) is used. However, set the TPnCCR0 and TPnCCR1 registers to the same value.

(e) TMPn counter read buffer register (TPnCNT)

By reading the TPnCNT register, the count value of the 16-bit counter can be read.

(f) TMPn capture/compare register 0 (TPnCCR0)

If the TPnCCR0 register is set to D_0, the interval is as follows.

Interval = $(D_0 + 1) \times$ Count clock cycle

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Figure 7-8. Register Setting for Interval Timer Mode Operation (3/3)

<R> **(g) TMPn capture/compare register 1 (TPnCCR1)**

The TPnCCR1 register is not used in the interval timer mode. However, the set value of the TPnCCR1 register is transferred to the CCR1 buffer register. When the count value of the 16-bit counter matches the value of the CCR1 buffer register, the TOPn1 pin output is inverted and a compare match interrupt request signal (INTTPnCC1) is generated.

By setting this register to the same value as the value set in the TPnCCR0 register, a square wave with a duty factor of 50% can be output from the TOPn1 pin.

When the TPnCCR1 register is not used, it is recommended to set its value to FFFFH. Also mask the register by the interrupt mask flag (TPnCCIC1.TPnCCMK1).

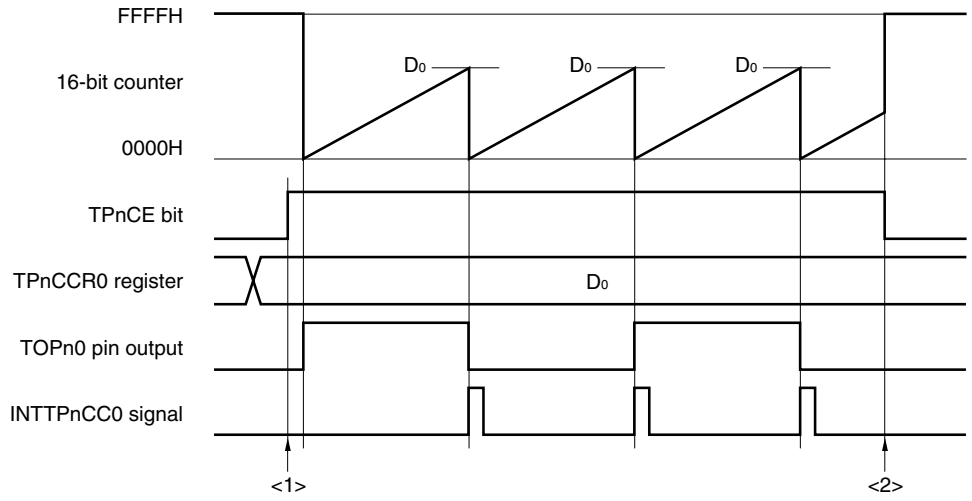
<R> **Remarks 1.** TMPn I/O control register 1 (TPnIOC1) and TMPn option register 0 (TPnOPT0) are not used in the interval timer mode.

2. μPD703249Y and 703252Y: n = 0 to 4

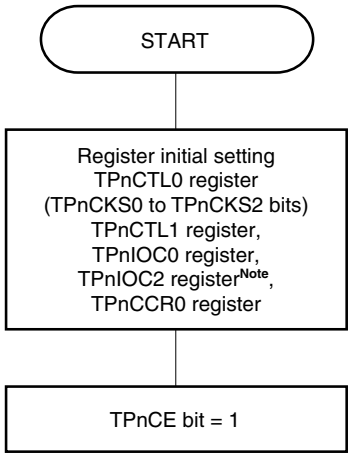
<R> μPD703253Y: n = 0 to 5

(1) Interval timer mode operation flow

Figure 7-9. Software Processing Flow in Interval Timer Mode



<1> Count operation start flow



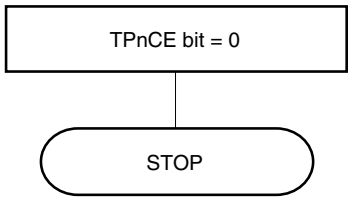
Initial setting of these registers is performed before setting the TPnCE bit to 1.

The TPnCKS0 to TPnCKS2 bits can be set at the same time when counting has been started (TPnCE bit = 1).

<R>

<R> **Note** The TPnEES1 and TPnEES0 bits can be set only when timer output (TOPn1) is used. However, set the TPnCCR0 and TPnCCR1 registers to the same value.

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TPnCE bit to 0. The output level of the TOPn0 pin is as specified by the TPnIOC0 register.

Remark μPD703249Y and 703252Y: n = 0 to 4

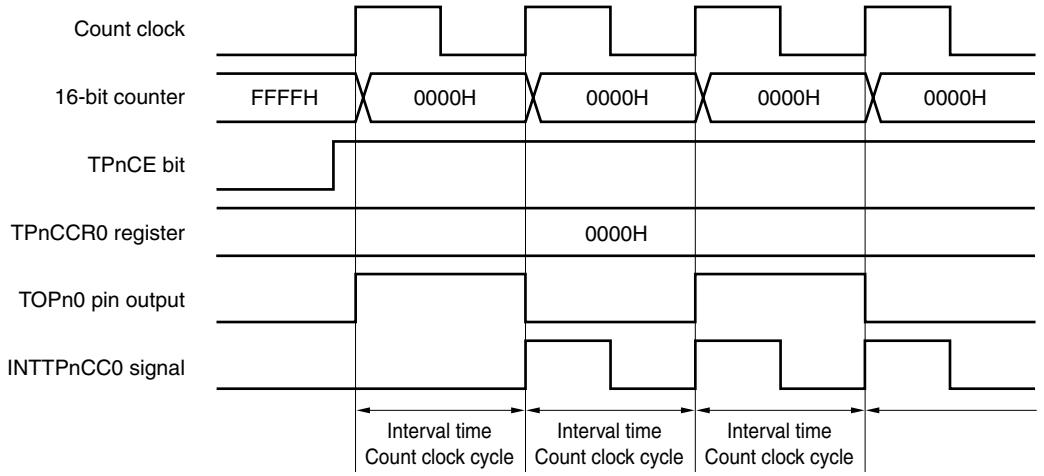
<R> μPD703253Y: n = 0 to 5

(2) Interval timer mode operation timing

<R> **(a) Operation if TPnCCR0 register is set to 0000H**

If the TPnCCR0 register is set to 0000H, the INTTPnCC0 signal is generated at each count clock, and the output of the TOPn0 pin is inverted.

The value of the 16-bit counter is always 0000H.

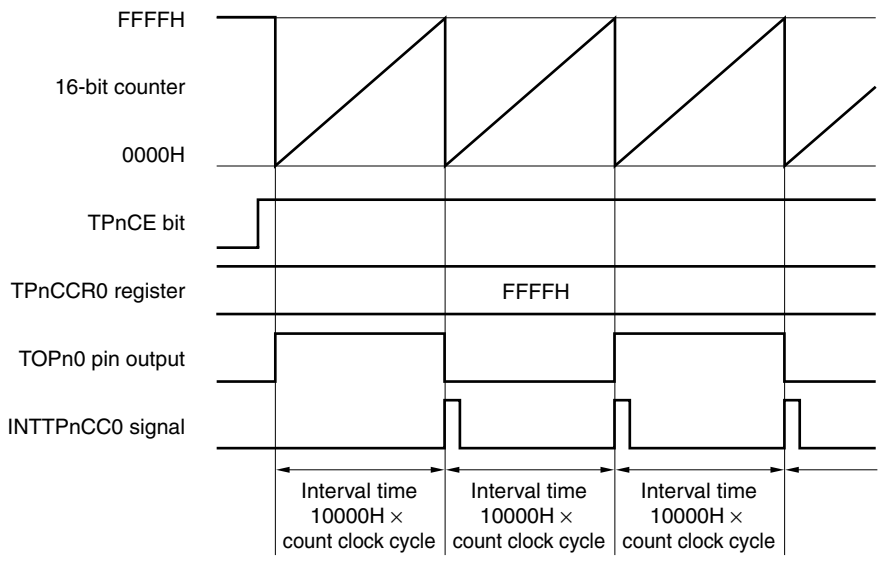


Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

(b) Operation if TPnCCR0 register is set to FFFFH

If the TPnCCR0 register is set to FFFFH, the 16-bit counter counts up to FFFFH. The counter is cleared to 0000H in synchronization with the next count-up timing. The INTTPnCC0 signal is generated and the output of the TOPn0 pin is inverted. At this time, an overflow interrupt request signal (INTTPnOV) is not generated, nor is the overflow flag (TPnOPT0.TPnOVF bit) set to 1.

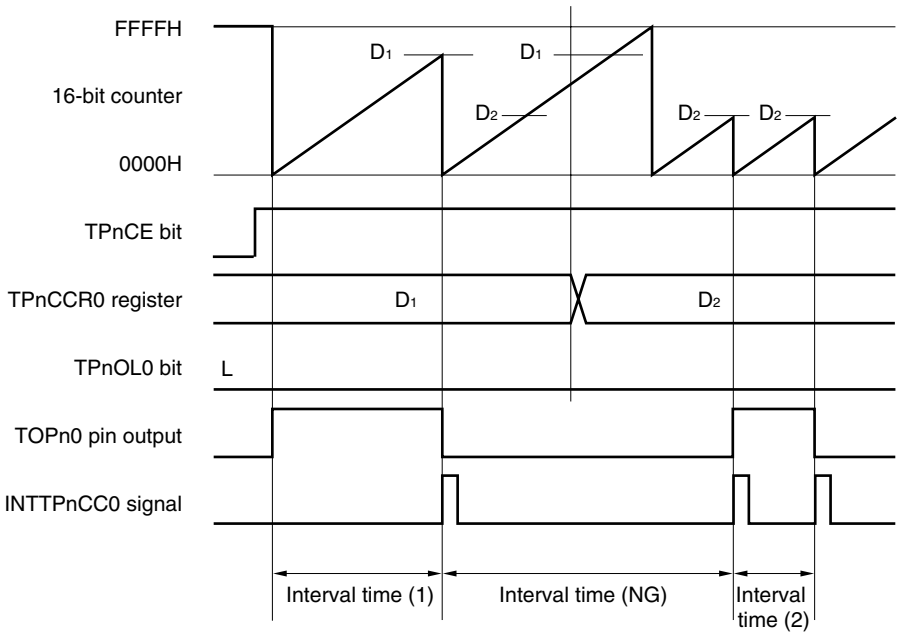


Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

(c) Notes on rewriting TPnCCR0 register

When the value of the TPnCCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow. If there is a possibility of overflow, stop counting and then change the set value.



Remarks 1. Interval time (1): $(D_1 + 1) \times$ Count clock cycle
Interval time (NG): $(10000H + D_2 + 1) \times$ Count clock cycle
Interval time (2): $(D_2 + 1) \times$ Count clock cycle

2. μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

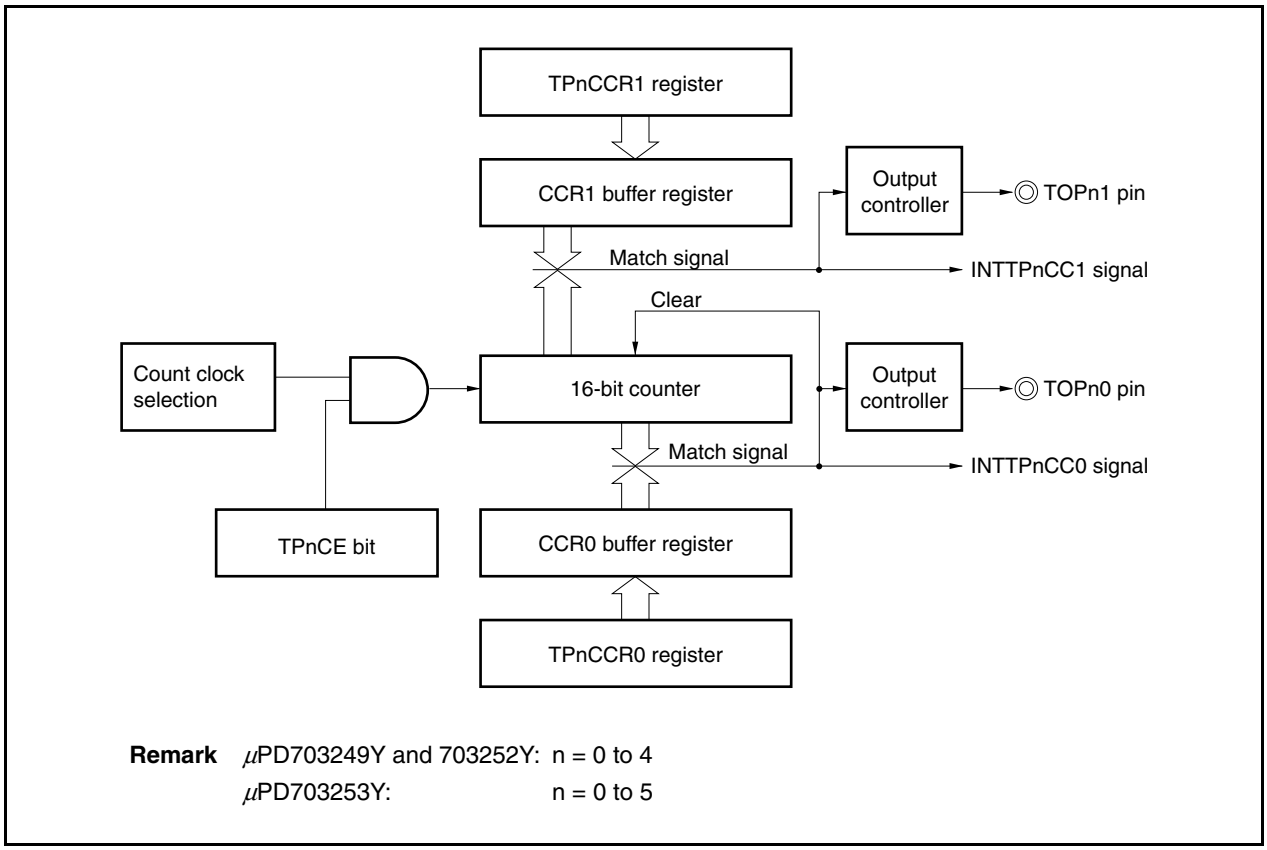
If the value of the TPnCCR0 register is changed from D_1 to D_2 while the count value is greater than D_2 but less than D_1, the count value is transferred to the CCR0 buffer register as soon as the TPnCCR0 register has been rewritten. Consequently, the value of the 16-bit counter that is compared is D_2.

Because the count value has already exceeded D_2, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches D_2, the INTTPnCC0 signal is generated and the output of the TOPn0 pin is inverted.

Therefore, the INTTPnCC0 signal may not be generated at the interval time "$(D_1 + 1) \times$ Count clock cycle" or "$(D_2 + 1) \times$ Count clock cycle" originally expected, but may be generated at an interval of "$(10000H + D_2 + 1) \times$ Count clock period".

(d) Operation of TPnCCR1 register

Figure 7-10. Configuration of TPnCCR1 Register



Remark μPD703249Y and 703252Y: n = 0 to 4

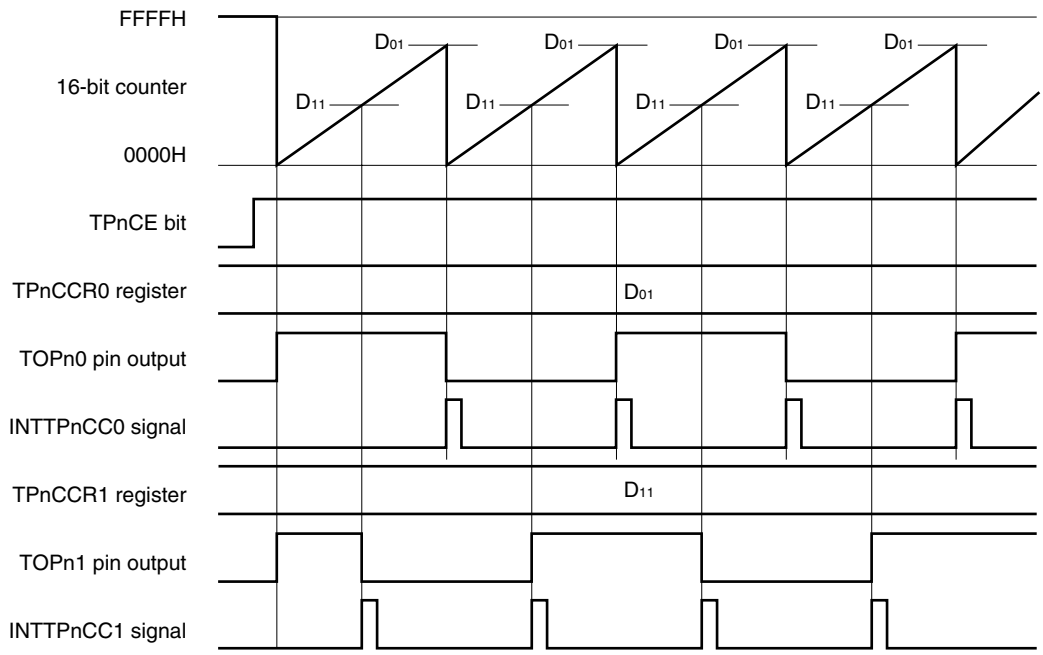
<R> μPD703253Y: n = 0 to 5

<R> When the TPnCCR1 register is set to the same value as the TPnCCR0 register, the INTTPnCC1 signal is generated at the same timing as the INTTPnCC0 signal and the TOPn1 pin output is inverted. In other words, a square wave with a duty factor of 50% can be output from the TOPn1 pin.

The following shows the operation when the TPnCCR1 register is set to other than the value set in the TPnCCR0 register.

If the set value of the TPnCCR1 register is less than the set value of the TPnCCR0 register, the INTTPnCC1 signal is generated once per cycle. At the same time, the output of the TOPn1 pin is inverted. The TOPn1 pin outputs a square wave with a duty factor of 50% after outputting a short-width pulse.

Figure 7-11. Timing Chart When $D_{01} \geq D_{11}$



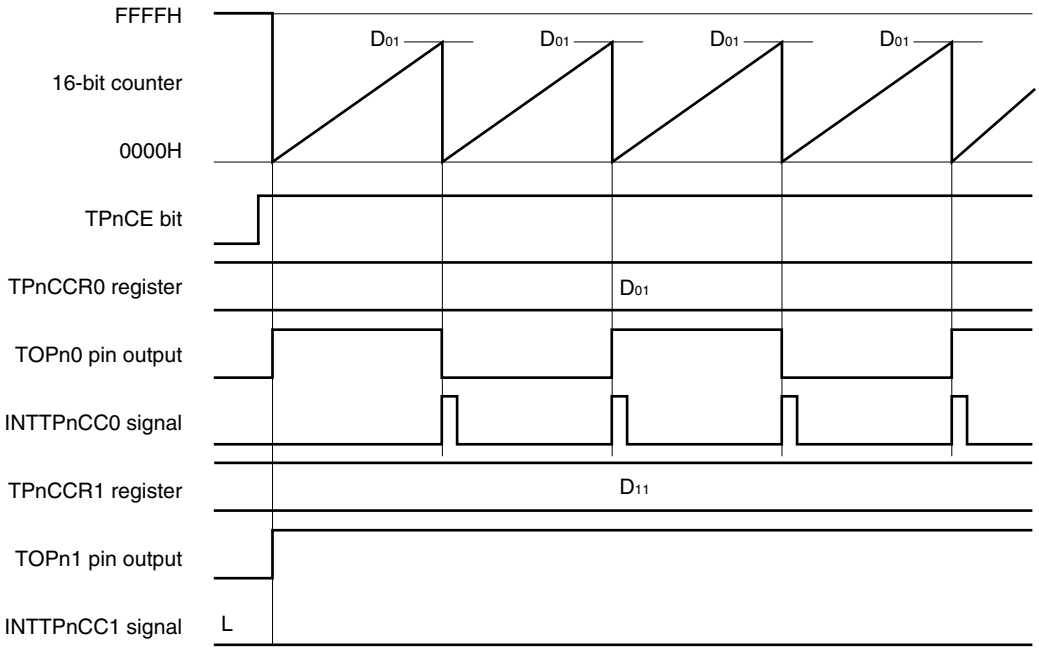
Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

If the set value of the TPnCCR1 register is greater than the set value of the TPnCCR0 register, the count value of the 16-bit counter does not match the value of the TPnCCR1 register. Consequently, the INTTPnCC1 signal is not generated, nor is the output of the TOPn1 pin changed.

<R> When the TPnCCR1 register is not used, it is recommended to set its value to FFFFH.

Figure 7-12. Timing Chart When $D_{01} < D_{11}$



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

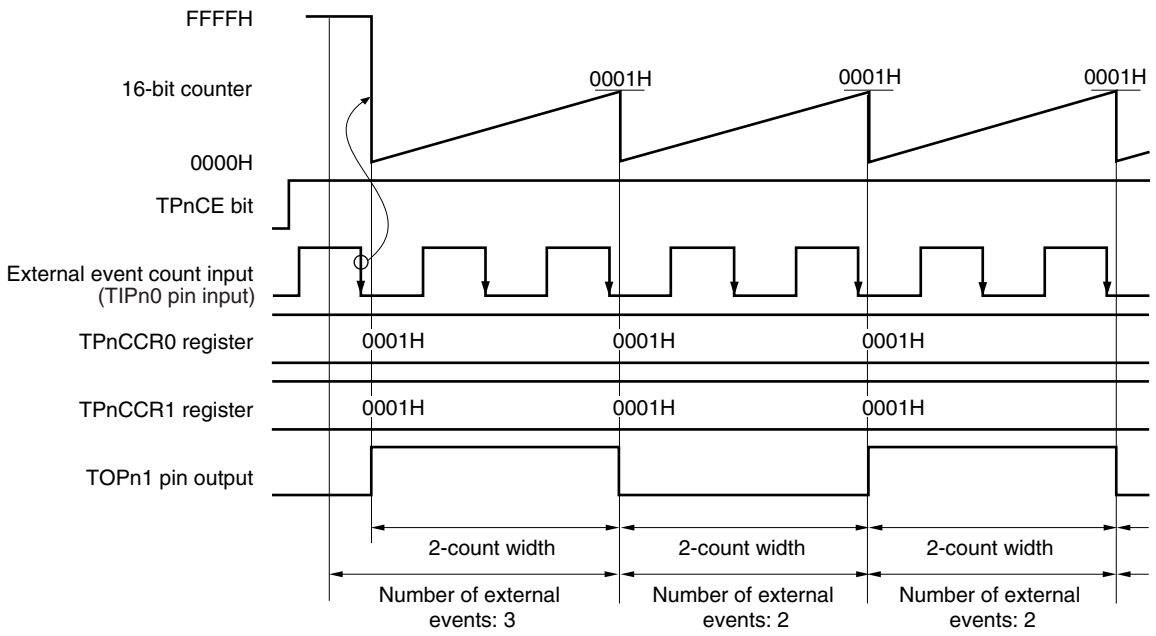
<R> **(3) Operation by external event count input (TIPn0)**

(a) Operation

To count the 16-bit counter at the valid edge of external event count input (TIPn0) in the interval timer mode, the valid edge of the external event count input is necessary once because the 16-bit counter is cleared from FFFFH to 0000H immediately after the TPnCE bit is set from 0 to 1.

When 0001H is set to both the TPnCCR0 and TPnCCR1 registers, the TOPn1 pin output is inverted each time the 16-bit counter counts twice.

The TPnCTL1.TPnEEE bit can be set to 1 in the interval timer mode only when the timer output (TOPn1) is used with the external event count input.



Remark μPD703249Y and 703252Y: n = 0 to 4
μPD703253Y: n = 0 to 5

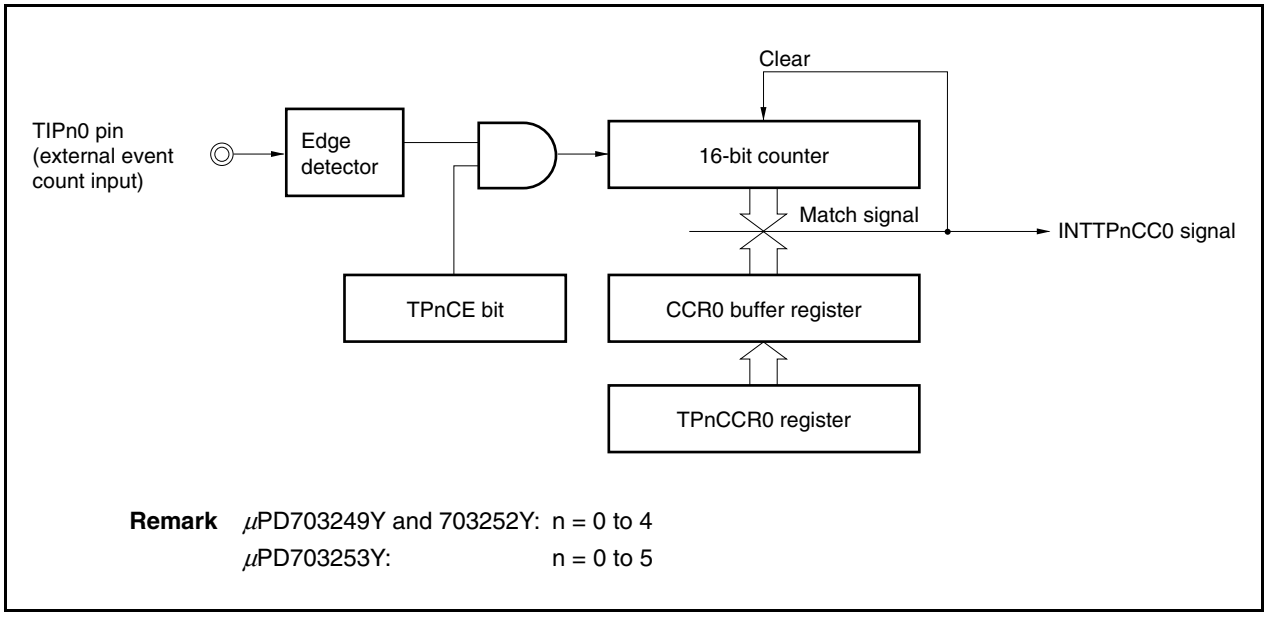
7.6.2 External event count mode (TPnMD2 to TPnMD0 bits = 001)

In the external event count mode, the valid edge of the external event count input (TIPn0) is counted when the
<R> TPnCTL0.TPnCE bit is set to 1, and an interrupt request signal (INTTPnCC0) is generated each time the number of edges set by the TPnCCR0 register have been counted. The TOPn0 and TOPn1 pins cannot be used. When using the TOPn1 pin for external event count input, set the TPnCTL1.TPnEEE bit to 1 in the interval timer mode (see **7.6.1 (3) Operation by external event count input (TIPn0)**).

The TPnCCR1 register is not used in the external event count mode.

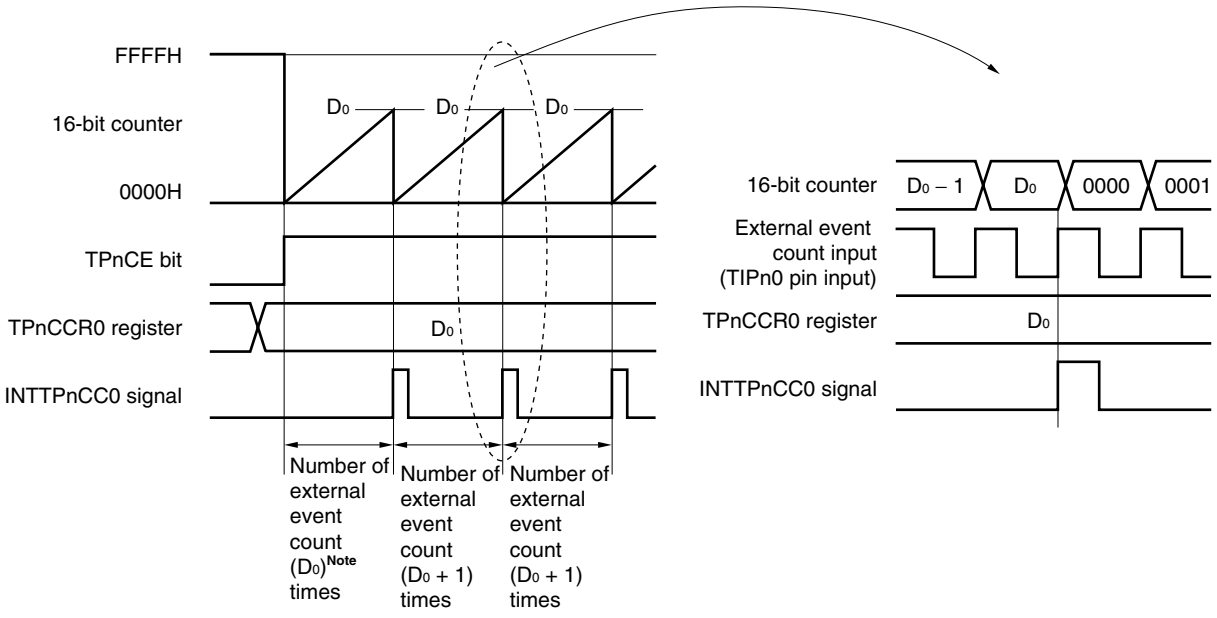
<R> **Caution In the external event count mode, the TPnCCR0 and TPnCCR1 registers must not be cleared to 0000H.**

Figure 7-13. Configuration in External Event Count Mode



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

Figure 7-14. Basic Timing in External Event Count Mode



<R>

<R>

Note In the external event count mode, when the TPnCTL0.TPnCE bit is set to 1 (operation starts), the 16-bit counter is cleared from FFFFH to 0000H at the same time. The first count operation starts from 0001H each time the valid edge of the external event count input is detected. Therefore, the count of the first count operation is one number smaller than the count of the second or subsequent count operation.

Remarks 1. This figure shows the basic timing when the rising edge is specified as the valid edge of the external event count input.

2. μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

When the TPnCE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H. The counter counts each time the valid edge of external event count input is detected. Additionally, the set value of the TPnCCR0 register is transferred to the CCR0 buffer register.

When the count value of the 16-bit counter matches the value of the CCR0 buffer register, the 16-bit counter is cleared to 0000H, and a compare match interrupt request signal (INTTPnCC0) is generated.

<R> The INTTPnCC0 signal is generated for the first time when the valid edge of the external event count input has been detected "value set to TPnCCR0 register" times. After that, the INTTPnCC0 signal is generated each time the valid edge of the external event count input has been detected "value set to TPnCCR0 register + 1" times.

Figure 7-15. Register Setting for Operation in External Event Count Mode (1/2)

(a) TMPn control register 0 (TPnCTL0)

	TPnCE					TPnCKS2	TPnCKS1	TPnCKS0
TPnCTL0	0/1	0	0	0	0	0	0	0

0: Stop counting
1: Enable counting

(b) TMPn control register 1 (TPnCTL1)

		TPnEST	TPnEEE			TPnMD2	TPnMD1	TPnMD0
TPnCTL1	0	0	0	0	0	0	0	1

0, 0, 1:
External event count mode

(c) TMPn I/O control register 2 (TPnIOC2)

					TPnEES1	TPnEES0	TPnETS1	TPnETS0
TPnIOC2	0	0	0	0	0/1	0/1	0	0

Select valid edge
of external event
count input (TIPn0 pin)

Figure 7-15. Register Setting for Operation in External Event Count Mode (2/2)

(d) TMPn counter read buffer register (TPnCNT)

The count value of the 16-bit counter can be read by reading the TPnCNT register.

<R> **(e) TMPn capture/compare register 0 (TPnCCR0)**

If the TPnCCR0 register is set to D_0, the count is cleared when the number of external events has reached (D_0) and the first compare match interrupt request signal (INTTPnCC0) is generated. The second compare match interrupt request signal (INTTPnCC0) is generated when the number of external events has reached (D_0 + 1).

<R> **(f) TMPn capture/compare register 1 (TPnCCR1)**

The TPnCCR1 register is not used in the external event count mode. However, the set value of the TPnCCR1 register is transferred to the CCR1 buffer register. When the count value of the 16-bit counter matches the value of the CCR1 buffer register, a compare match interrupt request signal (INTTPnCC1) is generated.

When the TPnCCR1 registers are not used, it is recommended to set their value to FFFFH. Also mask the register by the interrupt mask flag (TPnCCIC1.TPnCCMK1).

Cautions 1. Set 00H to the TPnIOC0 register.

<R> **2. When the external clock is used as the count clock, the external clock can be input only from the TIPn0 pin. At this time, clear the TPnIOC1.TPnIS1 and TPnIS0 bits to 00 (capture trigger input (TIPn0 pin): No edge detected).**

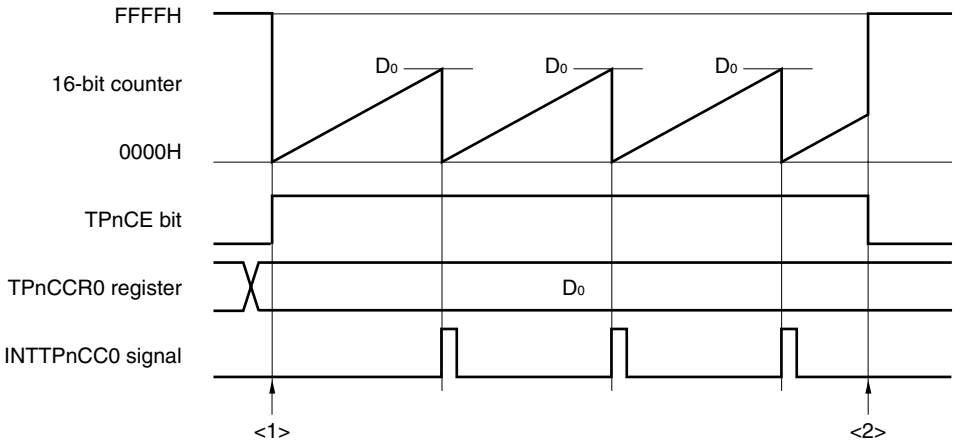
Remarks 1. TMPn I/O control register 1 (TPnIOC1) and TMPn option register 0 (TPnOPT0) are not used in the external event count mode.

2. μPD703249Y and 703252Y: n = 0 to 4

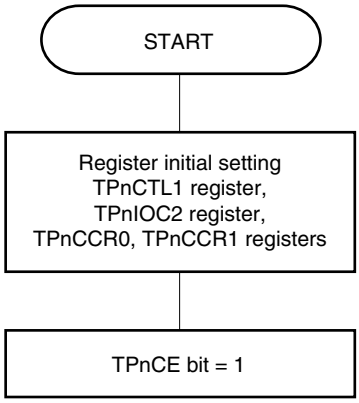
<R> μPD703253Y: n = 0 to 5

(1) External event count mode operation flow

Figure 7-16. Flow of Software Processing in External Event Count Mode

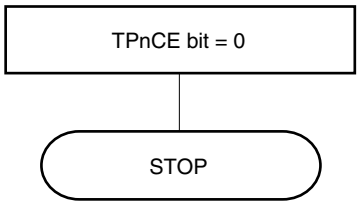


<1> Count operation start flow



Initial setting of these registers is performed before the TPnCE bit is set to 1.

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TPnCE bit to 0.

Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

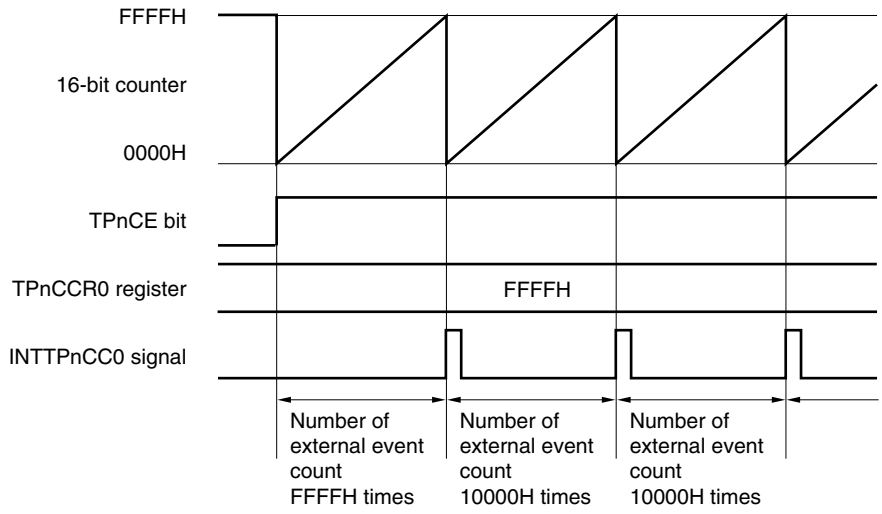
(2) Operation timing in external event count mode

Cautions 1. In the external event count mode, do not set the TPnCCR0 and TPnCCR1 registers to 0000H.

2. In the external event count mode, use of the timer output (TOPn0, TOPn1) is disabled. If performing timer output (TOPn1) using external event count input (TIPn0), set the interval timer mode, and set the operation enabled (TPnCTL1.TPnEEE bit = 1) by the external event count input for the count clock (refer to 7.6.1 (3) Operation by external event count input (TIPn0)).

(a) Operation if TPnCCR0 register is set to FFFFH

If the TPnCCR0 register is set to FFFFH, the 16-bit counter counts to FFFFH each time the valid edge of the external event count signal has been detected. The 16-bit counter is cleared to 0000H in synchronization with the next count-up timing, and the INTTPnCC0 signal is generated. At this time, the TPnOPT0.TPnOVF bit is not set.



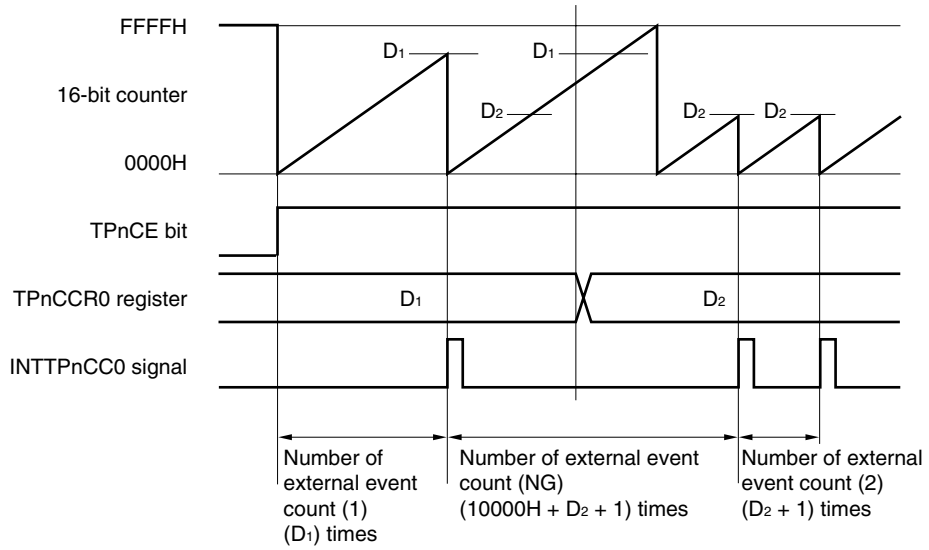
<R>

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

(b) Notes on rewriting the TPnCCR0 register

When the value of the TPnCCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow. If there is a possibility of overflow, stop counting and then change the set value.



<R>

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

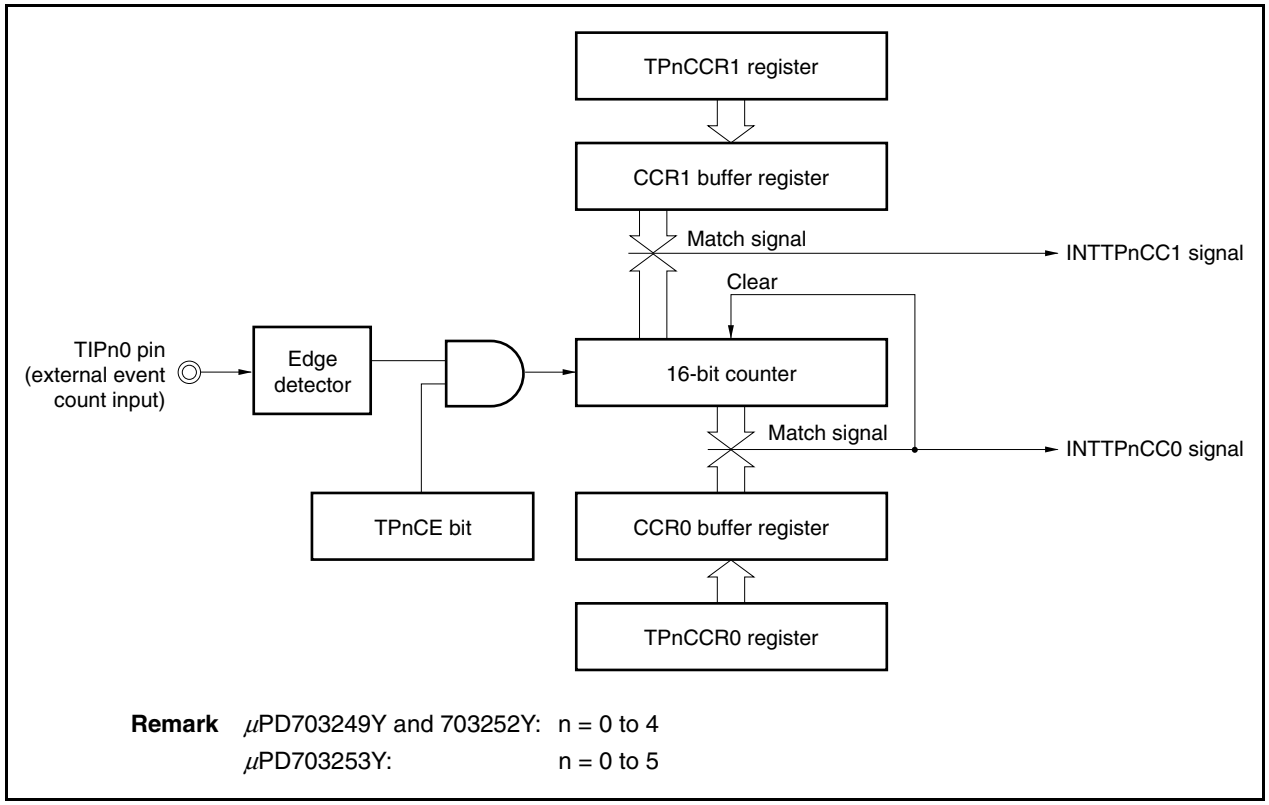
If the value of the TPnCCR0 register is changed from D_1 to D_2 while the count value is greater than D_2 but less than D_1, the count value is transferred to the CCR0 buffer register as soon as the TPnCCR0 register has been rewritten. Consequently, the value that is compared with the 16-bit counter is D_2.

Because the count value has already exceeded D_2, however, the 16-bit counter counts up to FFFFH, overflows, and then counts up again from 0000H. When the count value matches D_2, the INTTPnCC0 signal is generated.

Therefore, the INTTPnCC0 signal may not be generated at the valid edge count of "(D_1 + 1) times" or "(D_2 + 1) times" originally expected, but may be generated at the valid edge count of "(10000H + D_2 + 1) times".

(c) Operation of TPnCCR1 register

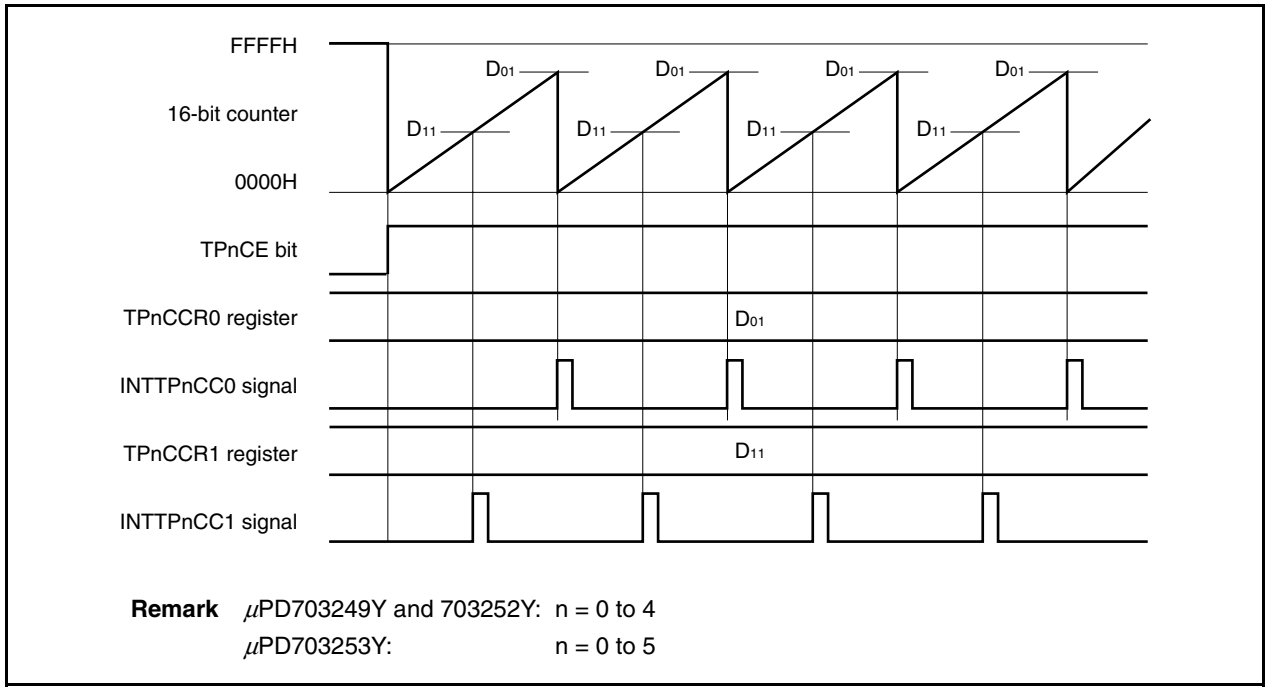
Figure 7-17. Configuration of TPnCCR1 Register



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

If the set value of the TPnCCR1 register is smaller than the set value of the TPnCCR0 register, the INTTPnCC1 signal is generated once per cycle.

Figure 7-18. Timing Chart When $D_{01} \geq D_{11}$

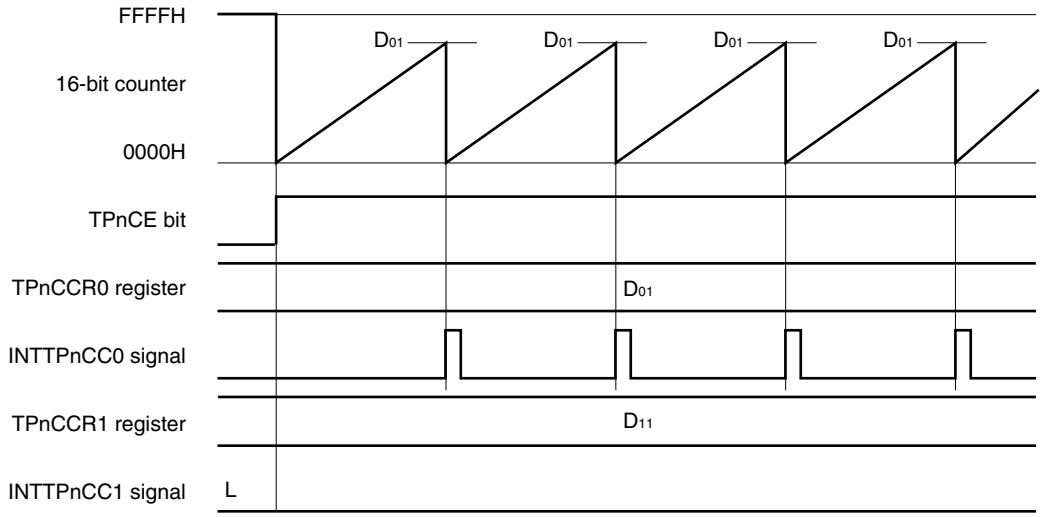


Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

If the set value of the TPnCCR1 register is greater than the set value of the TPnCCR0 register, the INTTPnCC1 signal is not generated because the count value of the 16-bit counter and the value of the TPnCCR1 register do not match.

<R> It is recommended to set FFFFH to the TPnCCR1 register when the TPnCCR1 register is not used.

Figure 7-19. Timing Chart When $D_{01} < D_{11}$



Remark μPD703249Y and 703252Y: n = 0 to 4

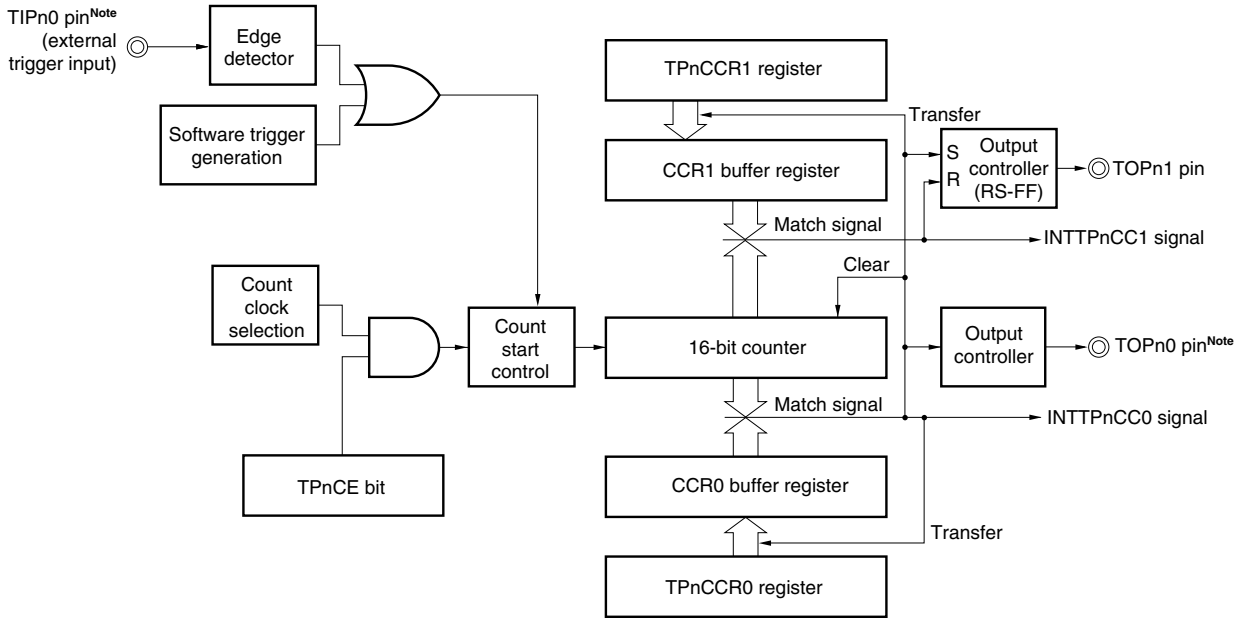
<R> μPD703253Y: n = 0 to 5

7.6.3 External trigger pulse output mode (TPnMD2 to TPnMD0 bits = 010)

In the external trigger pulse output mode, 16-bit timer/event counter P waits for a trigger when the TPnCTL0.TPnCE bit is set to 1. When the valid edge of an external trigger input (TIPn0) is detected, 16-bit timer/event counter P starts counting, and outputs a PWM waveform from the TOPn1 pin.

Pulses can also be output by generating a software trigger instead of using the external trigger. When using a software trigger, a square wave with a duty factor of 50% that has the set value of the TPnCCR0 register + 1 as half its cycle can also be output from the TOPn0 pin.

Figure 7-20. Configuration in External Trigger Pulse Output Mode

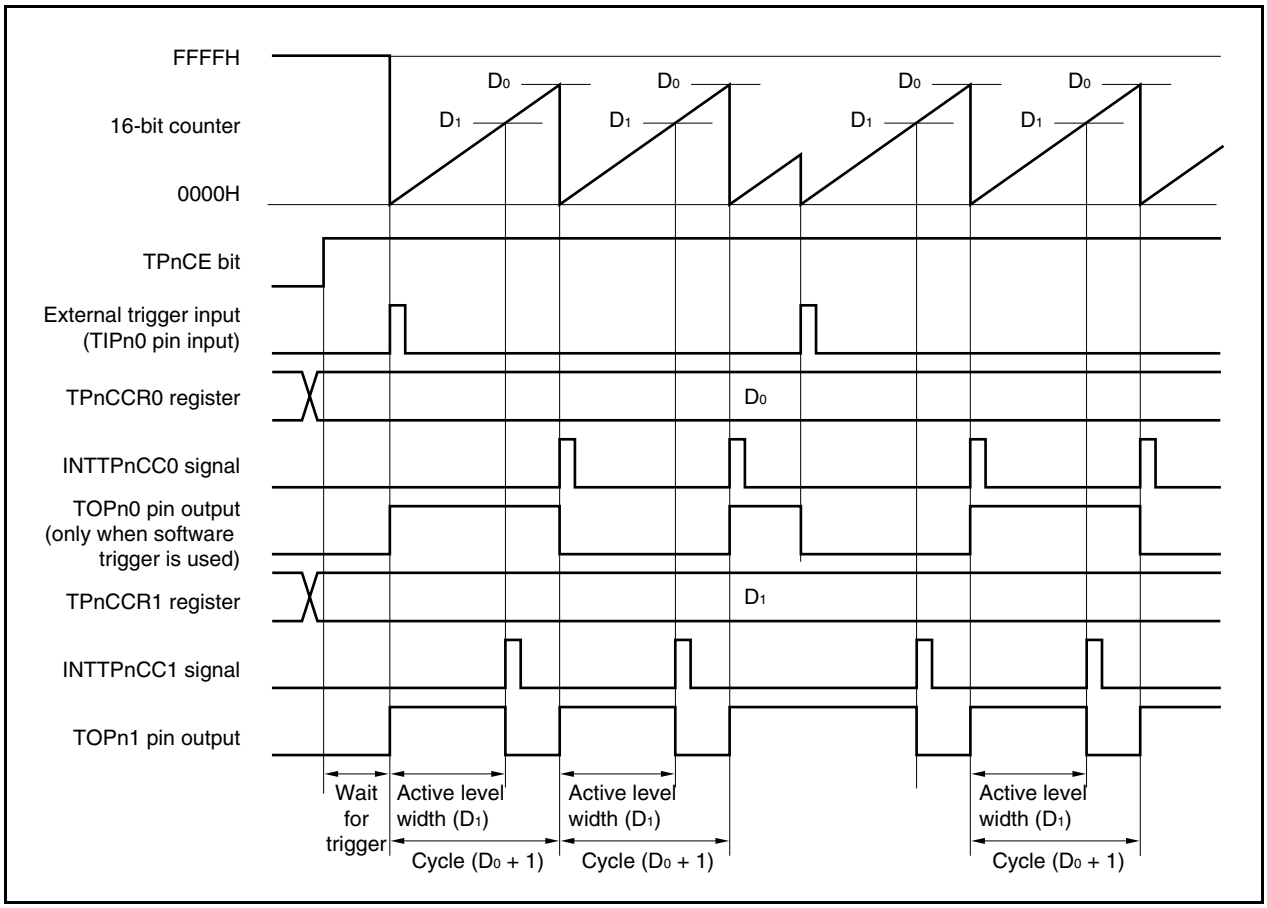


<R> **Note** Because the external trigger input pin (TIPn0) and timer output pin (TOPn0) share the same alternate-function pin, two functions cannot be used at the same time.

<R> **Caution In external trigger pulse output mode, select the internal clock (set TPnCTL1.TPnEEE bit = 0) as the count clock.**

Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

Figure 7-21. Basic Timing in External Trigger Pulse Output Mode



16-bit timer/event counter P waits for a trigger when the TPnCE bit is set to 1. When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting at the same time, and outputs a PWM wave\form from the TOPn1 pin. If the trigger is generated again while the counter is operating, the counter is cleared to 0000H and restarted. (The output of the TOPn0 pin is inverted. The TOPn1 pin outputs a high-level regardless of the status (high/low) when a trigger occurs.)

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

Active level width = (Set value of TPnCCR1 register) × Count clock cycle
Cycle = (Set value of TPnCCR0 register + 1) × Count clock cycle
Duty factor = (Set value of TPnCCR1 register)/(Set value of TPnCCR0 register + 1)

The compare match request signal INTTPnCC0 is generated when the 16-bit counter counts next time after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H. The compare match interrupt request signal INTTPnCC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The value set to the TPnCCRm register is transferred to the CCRm buffer register when the count value of the 16-bit counter matches the value of the CCRm buffer register and the 16-bit counter is cleared to 0000H.

The valid edge of an external trigger input (TIPn0) signal, or setting the software trigger (TPnCTL1.TPnEST bit) to 1 is used as the trigger.

<R> **Remark** μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1, μPD703253Y: n = 0 to 5, m = 0 or 1

Figure 7-22. Setting of Registers in External Trigger Pulse Output Mode (1/2)

(a) TMPn control register 0 (TPnCTL0)

<R>

	TPnCE					TPnCKS2	TPnCKS1	TPnCKS0
TPnCTL0	0/1	0	0	0	0	0/1	0/1	0/1

Select count clock

0: Stop counting
1: Enable counting

(b) TMPn control register 1 (TPnCTL1)

<R>

	TPnEST	TPnEEE				TPnMD2	TPnMD1	TPnMD0
TPnCTL1	0	0/1	0	0	0	0	1	0

0, 1, 0:
External trigger pulse output mode

0: Operate on count clock selected by TPnCKS0 to TPnCKS2 bits

Generate software trigger when 1 is written

(c) TMPn I/O control register 0 (TPnIOC0)

					TPnOL1	TPnOE1	TPnOL0	TPnOE0
TPnIOC0	0	0	0	0	0/1	0/1	0/1	0/1[Note]

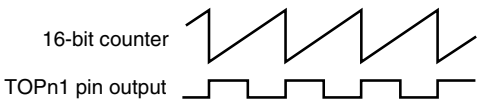
0: Disable TOPn0 pin output
1: Enable TOPn0 pin output

Setting level of TOPn1 pin in status of waiting for external trigger
0: Low level
1: High level

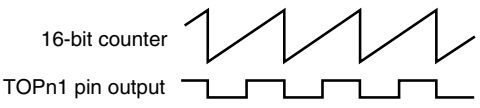
0: Disable TOPn1 pin output
1: Enable TOPn1 pin output

Setting level of TOPn1 pin in status of waiting for external trigger
0: Low level
1: High level

- When TPnOL1 bit = 0



- When TPnOL1 bit = 1



Note Clear this bit to 0 when the TOPn0 pin is not used in the external trigger pulse output mode.

Figure 7-22. Setting of Registers in External Trigger Pulse Output Mode (2/2)

(d) TMPn I/O control register 2 (TPnIOC2)

<R>

							TPnEES1	TPnEES0	TPnETS1	TPnETS0
TPnIOC2	0	0	0	0	0	0	0/1	0/1		

Select valid edge of external trigger input (TIPn0 pin)

(e) TMPn counter read buffer register (TPnCNT)

The value of the 16-bit counter can be read by reading the TPnCNT register.

(f) TMPn capture/compare registers 0 and 1 (TPnCCR0 and TPnCCR1)

If D_0 is set to the TPnCCR0 register and D_1 to the TPnCCR1 register, the cycle and active level of the PWM waveform are as follows.

Cycle = $(D_0 + 1) \times$ Count clock cycle

Active level width = $D_1 \times$ Count clock cycle

Remarks 1. TMPn I/O control register 1 (TPnIOC1) and TMPn option register 0 (TPnOPT0) are not used in the external trigger pulse output mode.

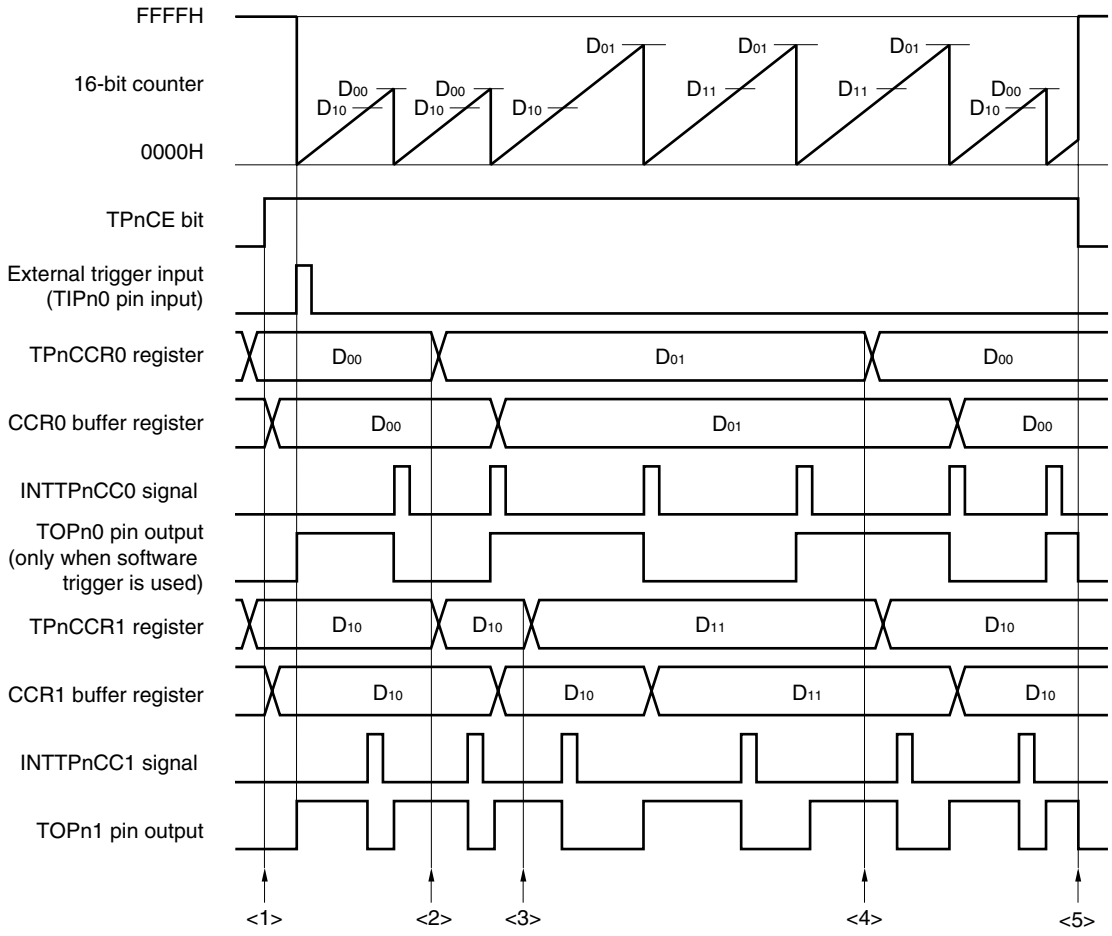
<R>

2. μPD703249Y and 703252Y: n = 0 to 4

μPD703253Y: n = 0 to 5

(1) Operation flow in external trigger pulse output mode

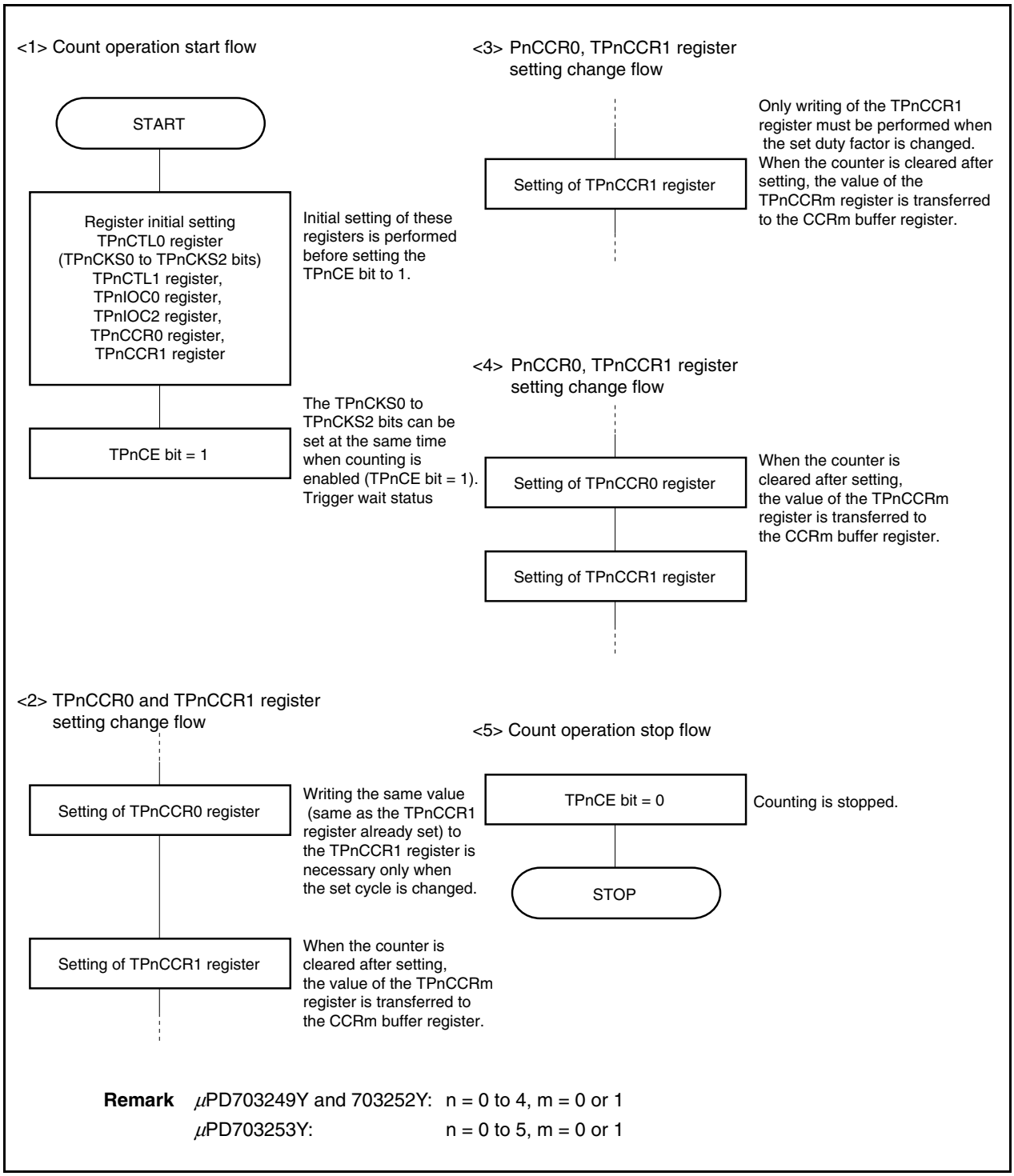
Figure 7-23. Software Processing Flow in External Trigger Pulse Output Mode (1/2)



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Figure 7-23. Software Processing Flow in External Trigger Pulse Output Mode (2/2)



Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

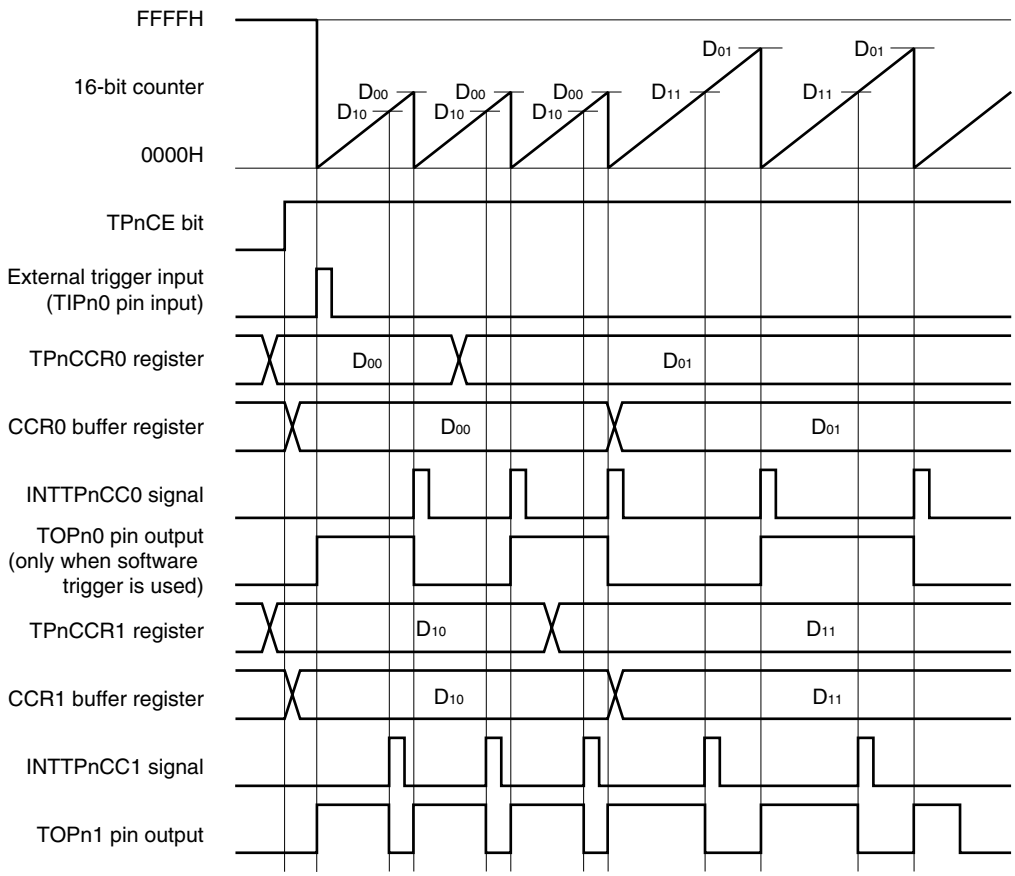
<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(2) External trigger pulse output mode operation timing

(a) Note on changing pulse width during operation

To change the PWM waveform while the counter is operating, write the TPnCCR1 register last.

Rewrite the TPnCCRm register after writing the TPnCCR1 register after the INTTPnCC0 signal is detected.



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

In order to transfer data from the TPnCCRm register to the CCRm buffer register, the TPnCCR1 register must be written.

To change both the cycle and active level width of the PWM waveform at this time, first set the cycle to the TPnCCR0 register and then set the active level width to the TPnCCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TPnCCR0 register, and then write the same value (same as the TPnCCR1 register already set) to the TPnCCR1 register.

To change only the active level width (duty factor) of the PWM waveform, only the TPnCCR1 register has to be set.

After data is written to the TPnCCR1 register, the value written to the TPnCCRm register is transferred to the CCRm buffer register in synchronization with clearing of the 16-bit counter, and is used as the value compared with the 16-bit counter.

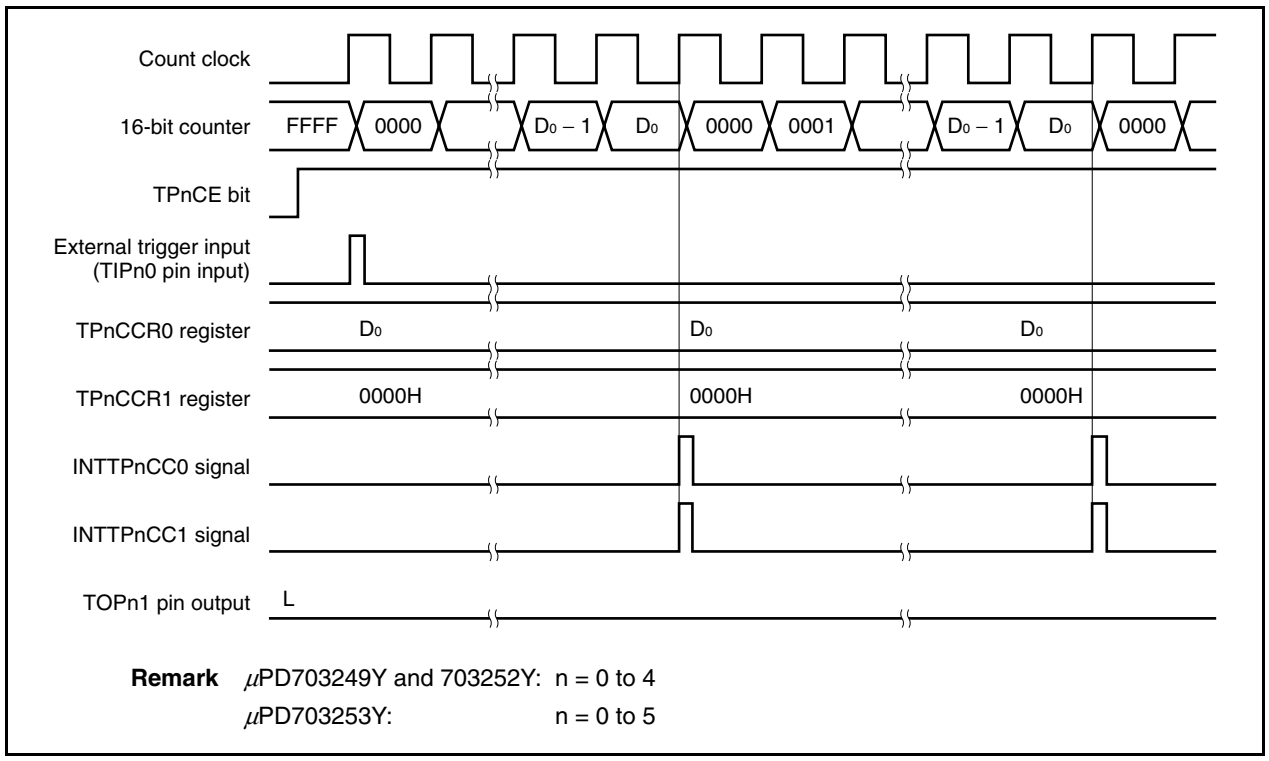
To write the TPnCCR0 or TPnCCR1 register again after writing the TPnCCR1 register once, do so after the INTTPnCC0 signal is generated. Otherwise, the value of the CCRm buffer register may become undefined because the timing of transferring data from the TPnCCRm register to the CCRm buffer register conflicts with writing the TPnCCRm register.

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

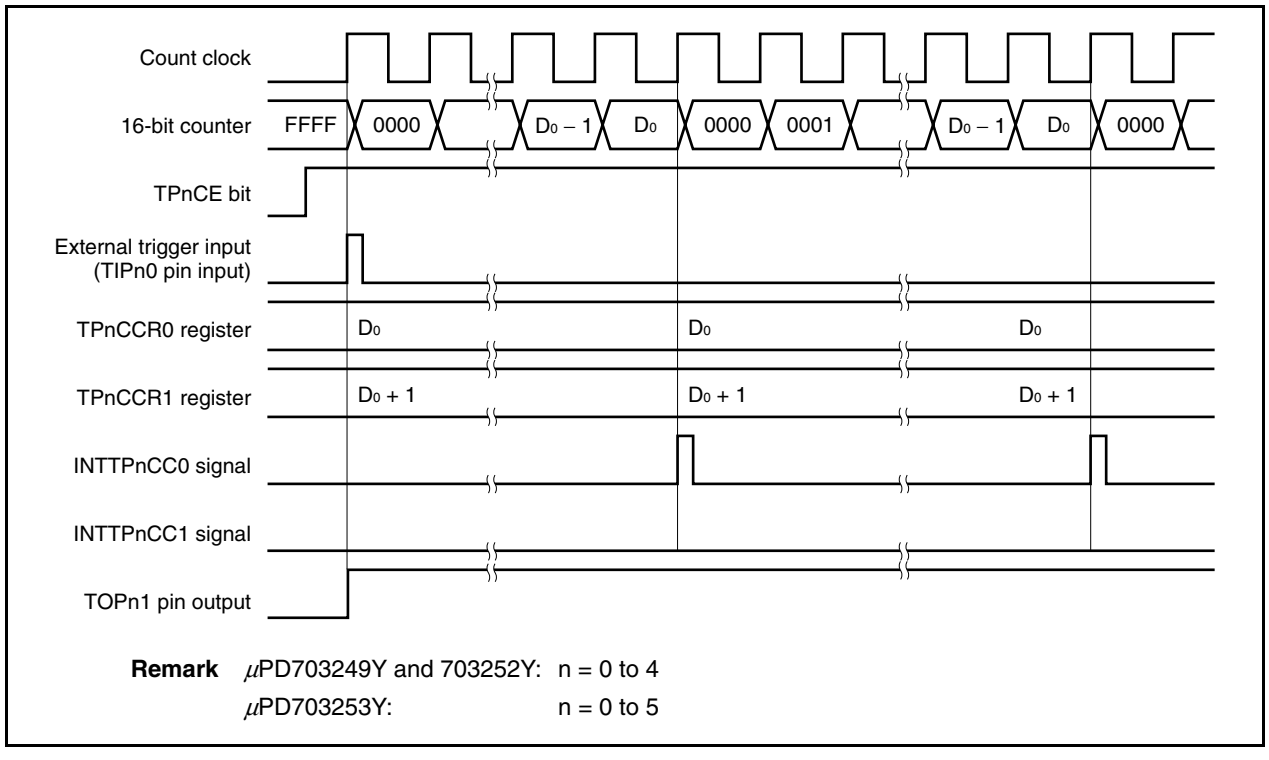
(b) 0%/100% output of PWM waveform

<R> To output a 0% waveform, set the TPnCCR1 register to 0000H. The 16-bit counter is cleared to 0000H and the INTTPnCO0 and INTTPnCC1 signals are generated at the next timing after a match between the count value of the 16-bit counter and the value of the CCR0 buffer register.



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

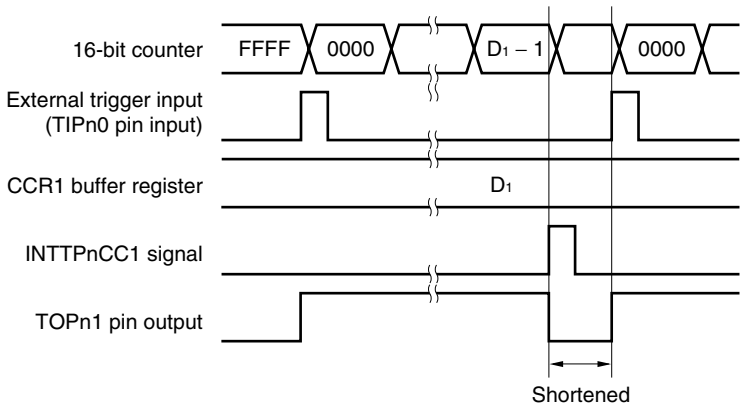
To output a 100% waveform, set a value of (set value of TPnCCR0 register + 1) to the TPnCCR1 register. If the set value of the TPnCCR0 register is FFFFH, 100% output cannot be produced.



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

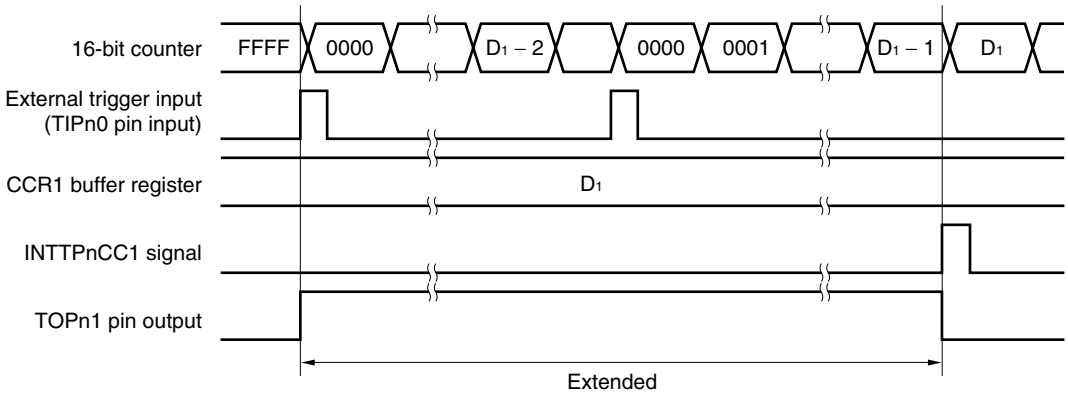
(c) Conflict between trigger detection and match with CCR1 buffer register

If the trigger is detected immediately after the INTTPnCC1 signal is generated, the 16-bit counter is immediately cleared to 0000H, the output signal of the TOPn1 pin is asserted, and the counter continues counting. Consequently, the inactive period of the PWM waveform is shortened.



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

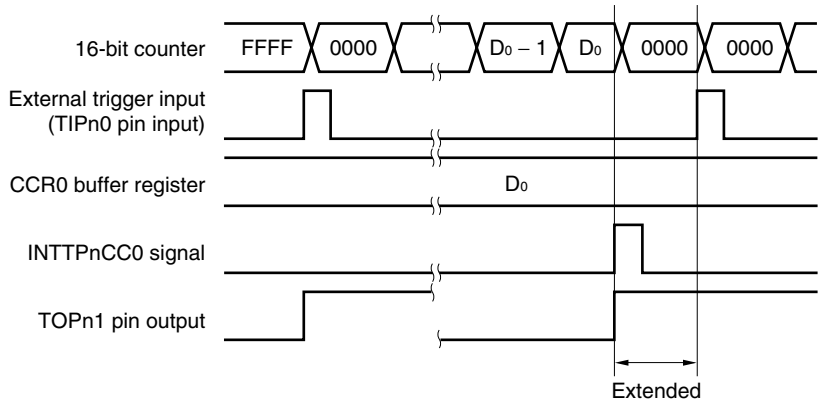
If the trigger is detected immediately before the INTTPnCC1 signal is generated, the INTTPnCC1 signal is not generated, and the 16-bit counter is cleared to 0000H and continues counting. The output signal of the TOPn1 pin remains active. Consequently, the active period of the PWM waveform is extended.



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

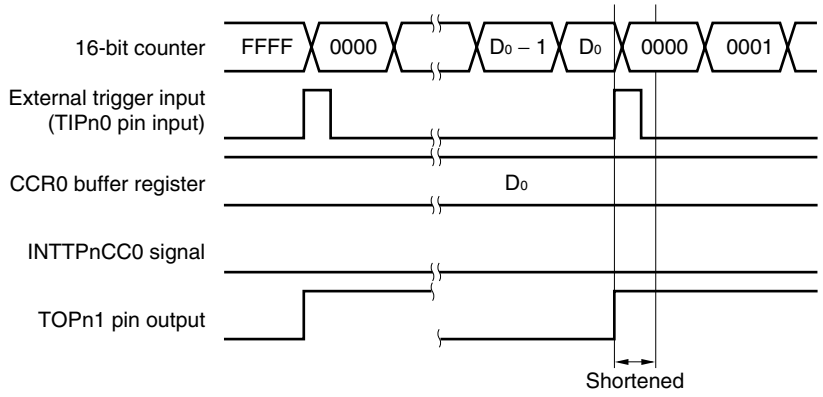
(d) Conflict between trigger detection and match with CCR0 buffer register

If the trigger is detected immediately after the INTTPnCC0 signal is generated, the 16-bit counter is cleared to 0000H and continues counting up. Therefore, the active period of the TOPn1 pin is extended by time from generation of the INTTPnCC0 signal to trigger detection.



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

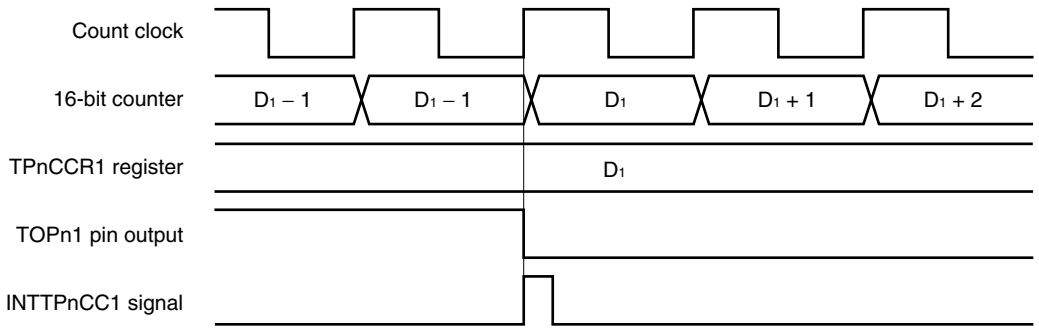
If the trigger is detected immediately before the INTTPnCC0 signal is generated, the INTTPnCC0 signal is not generated. The 16-bit counter is cleared to 0000H, the TOPn1 pin is asserted, and the counter continues counting. Consequently, the inactive period of the PWM waveform is shortened.



Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

(e) Generation timing of compare match interrupt request signal (INTTPnCC1)

The timing of generation of the INTTPnCC1 signal in the external trigger pulse output mode differs from the timing of other mode INTTPnCC1 signals; the INTTPnCC1 signal is generated when the count value of the 16-bit counter matches the value of the TPnCCR1 register.



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Usually, the INTTPnCC1 signal is generated in synchronization with the next count up, after the count value of the 16-bit counter matches the value of the TPnCCR1 register.

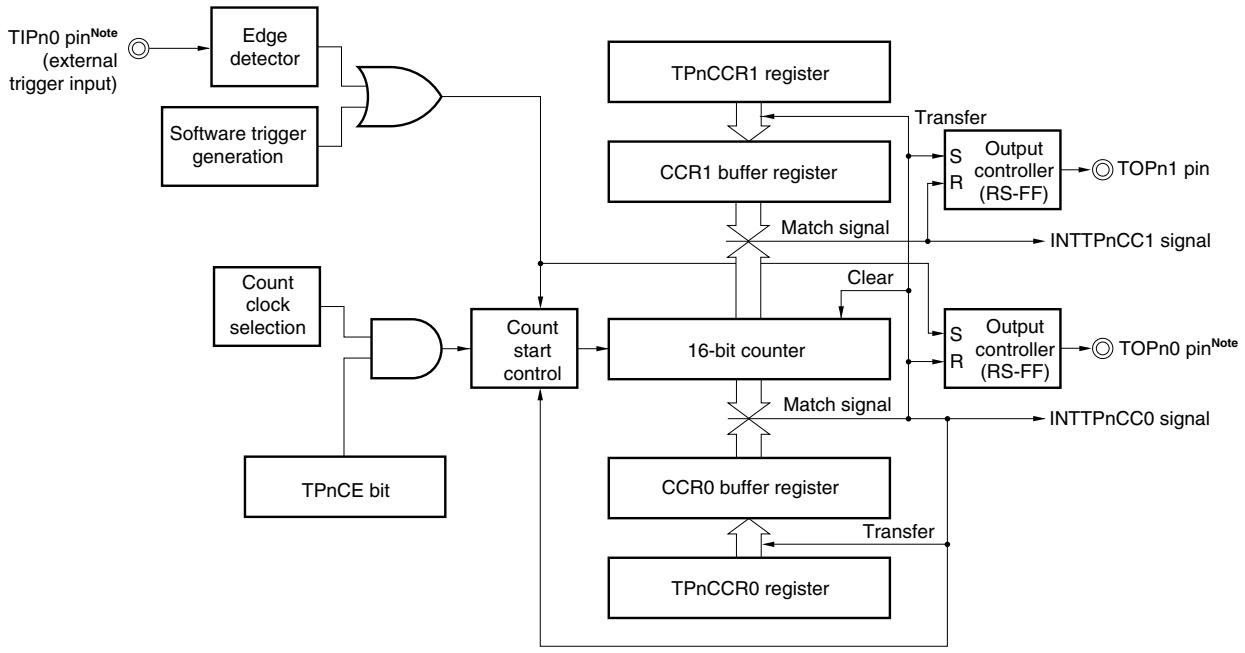
In the external trigger pulse output mode, however, it is generated one clock earlier. This is because the timing is changed to match the timing of changing the output signal of the TOPn1 pin.

7.6.4 One-shot pulse output mode (TPnMD2 to TPnMD0 bits = 011)

In the one-shot pulse output mode, 16-bit timer/event counter P waits for a trigger when the TPnCTL0.TPnCE bit is set to 1. When the valid edge of an external trigger input (TIPn0) is detected, 16-bit timer/event counter P starts counting, and outputs a one-shot pulse from the TOPn1 pin.

Instead of the external trigger input (TIPn0), a software trigger can also be generated to output the pulse. When the software trigger is used, the TOPn0 pin outputs the active level while the 16-bit counter is counting, and the inactive level when the counter is stopped (waiting for a trigger).

Figure 7-24. Configuration in One-Shot Pulse Output Mode



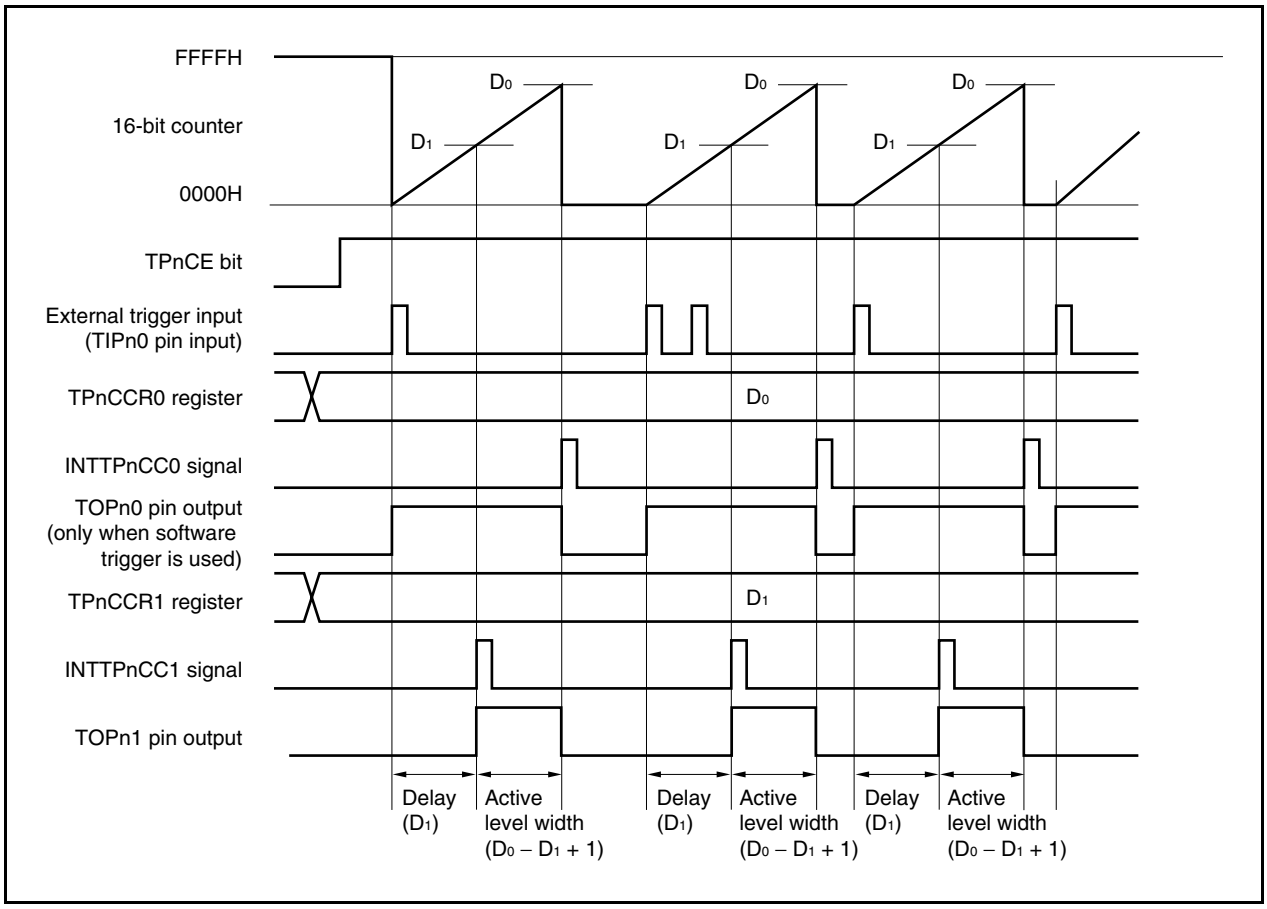
<R> **Note** Because the external trigger input pin (TIPn0) and timer output pin (TOPn0) share the same alternate-function pin, two functions cannot be used at the same time.

<R> **Caution In one-shot pulse output mode, select the internal clock (set TPnCTL1.TPnEEE bit = 0) as the count clock.**

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Figure 7-25. Basic Timing in One-Shot Pulse Output Mode



When the TPnCE bit is set to 1, 16-bit timer/event counter P waits for a trigger. When the trigger is generated, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a one-shot pulse from the TOPn1 pin. <R> After the one-shot pulse is output, the 16-bit counter is set to 0000H, stops counting, and waits for a trigger. When the trigger is generated again, the 16-bit counter starts counting from 0000H. If a trigger is generated again while the one-shot pulse is being output, it is ignored.

The output delay period and active level width of the one-shot pulse can be calculated as follows.

Output delay period = (Set value of TPnCCR1 register) × Count clock cycle

Active level width = (Set value of TPnCCR0 register − Set value of TPnCCR1 register + 1) × Count clock cycle

The compare match interrupt request signal INTTPnCC0 is generated when the 16-bit counter counts after its count value matches the value of the CCR0 buffer register. The compare match interrupt request signal INTTPnCC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The valid edge of an external trigger input (TIPn0 pin) or setting the software trigger (TPnCTL1.TPnEST bit) to 1 is used as the trigger.

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

Figure 7-26. Setting of Registers in One-Shot Pulse Output Mode (1/2)

(a) TMPn control register 0 (TPnCTL0)

<R>

	TPnCE					TPnCKS2	TPnCKS1	TPnCKS0
TPnCTL0	0/1	0	0	0	0	0/1	0/1	0/1

Select count clock

0: Stop counting
1: Enable counting

(b) TMPn control register 1 (TPnCTL1)

<R>

	TPnEST	TPnEEE				TPnMD2	TPnMD1	TPnMD0
TPnCTL1	0	0/1	0	0	0	0	1	1

0, 1, 1:
One-shot pulse output mode

0: Operate on count clock selected by TPnCKS0 to TPnCKS2 bits

Generate software trigger when 1 is written

(c) TMPn I/O control register 0 (TPnIOC0)

					TPnOL1	TPnOE1	TPnOL0	TPnOE0
TPnIOC0	0	0	0	0	0/1	0/1	0/1	0/1 Note

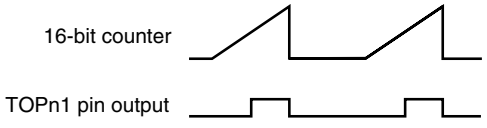
0: Disable TOPn0 pin output
1: Enable TOPn0 pin output

Setting of TOPn0 pin output level in status of waiting for external trigger
0: Low level
1: High level

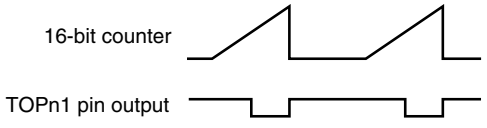
0: Disable TOPn1 pin output
1: Enable TOPn1 pin output

Setting of TOPn1 pin output level in status of waiting for external trigger
0: Low level
1: High level

• When TPnOL1 bit = 0



• When TPnOL1 bit = 1



Note Clear this bit to 0 when the TOPn0 pin is not used in the one-shot pulse output mode.

Figure 7-26. Setting of Registers in One-Shot Pulse Output Mode (2/2)

(d) TMPn I/O control register 2 (TPnIOC2)

<R>

							TPnEES1	TPnEES0	TPnETS1	TPnETS0
TPnIOC2	0	0	0	0	0	0	0/1	0/1		

Select valid edge of external trigger input (TIPn0 pin)

(e) TMPn counter read buffer register (TPnCNT)

The value of the 16-bit counter can be read by reading the TPnCNT register.

(f) TMPn capture/compare registers 0 and 1 (TPnCCR0 and TPnCCR1)

If D_0 is set to the TPnCCR0 register and D_1 to the TPnCCR1 register, the active level width and output delay period of the one-shot pulse are as follows.

Active level width = $(D_0 - D_1 + 1) \times$ Count clock cycle

Output delay period = $D_1 \times$ Count clock cycle

<R>

Caution The one-shot pulse is not output if the set value of the TPnCCR1 register is greater than the set value of the TPnCCR0 register in the one-shot pulse output mode.

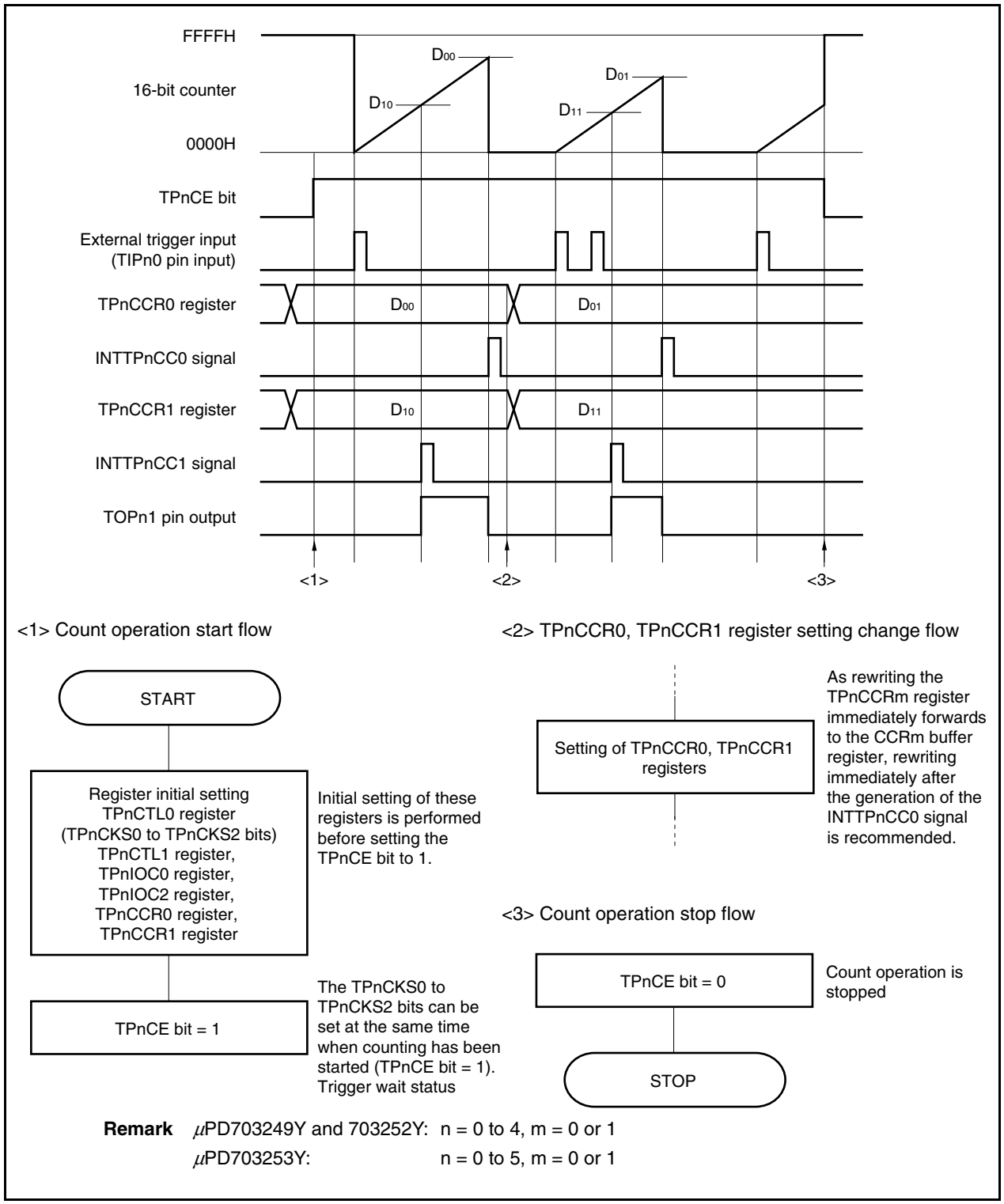
Remarks 1. TMPn I/O control register 1 (TPnIOC1) and TMPn option register 0 (TPnOPT0) are not used in the one-shot pulse output mode.

2. μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

(1) Operation flow in one-shot pulse output mode

Figure 7-27. Software Processing Flow in One-Shot Pulse Output Mode



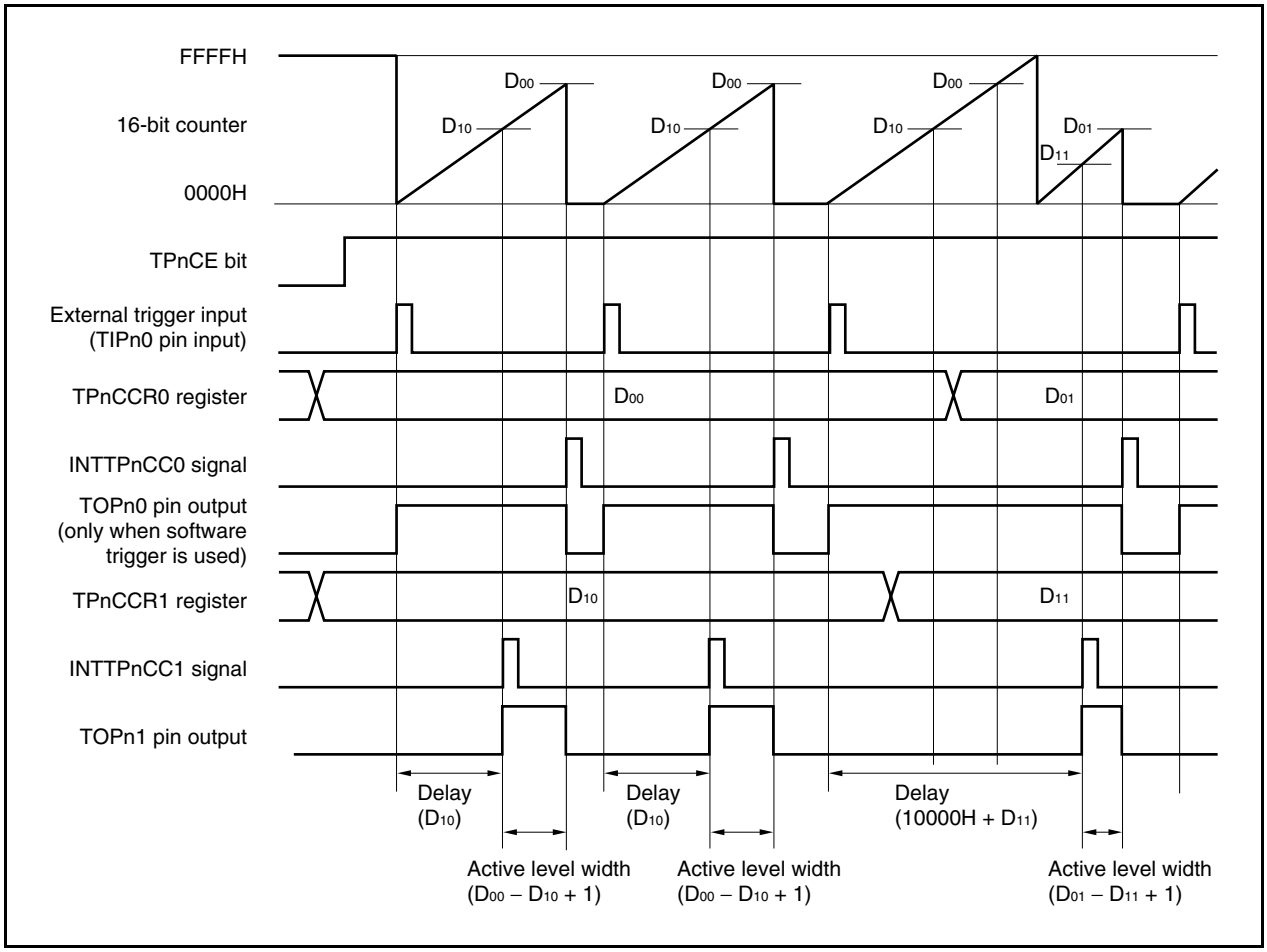
Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(2) Operation timing in one-shot pulse output mode

(a) Note on rewriting TPnCCRm register

When the value of the TPnCCRm register is rewritten to a smaller value during counting, the 16-bit counter may overflow. If there is a possibility of overflow, stop counting and then change the set value.



When the TPnCCR0 register is rewritten from D_{00} to D_{01} and the TPnCCR1 register from D_{10} to D_{11} where $D_{00} > D_{01}$ and $D_{10} > D_{11}$, if the TPnCCR1 register is rewritten when the count value of the 16-bit counter is greater than D_{11} and less than D_{10} and if the TPnCCR0 register is rewritten when the count value is greater than D_{01} and less than D_{00}, each set value is reflected as soon as the register has been rewritten and compared with the count value. The counter counts up to FFFFH and then counts up again from 0000H. When the count value matches D_{11}, the counter generates the INTTPnCC1 signal and asserts the TOPn1 pin. When the count value matches D_{01}, the counter generates the INTTPnCC0 signal, deasserts the TOPn1 pin, and stops counting.

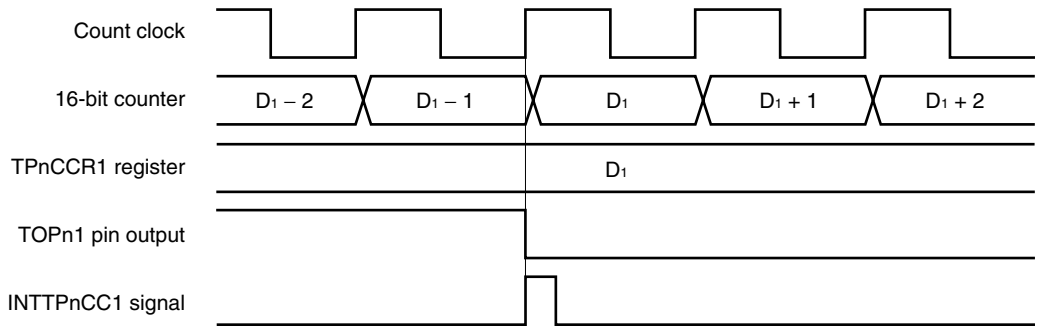
Therefore, the counter may output a pulse with a delay period or active period different from that of the one-shot pulse that is originally expected.

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(b) Generation timing of compare match interrupt request signal (INTTPnCC1)

The generation timing of the INTTPnCC1 signal in the one-shot pulse output mode is different from other mode INTTPnCC1 signals; the INTTPnCC1 signal is generated when the count value of the 16-bit counter matches the value of the TPnCCR1 register.



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Usually, the INTTPnCC1 signal is generated when the 16-bit counter counts up next time after its count value matches the value of the TPnCCR1 register.

In the one-shot pulse output mode, however, it is generated one clock earlier. This is because the timing is changed to match the change timing of the TOPn1 pin.

Remark μPD703249Y and 703252Y: n = 0 to 4

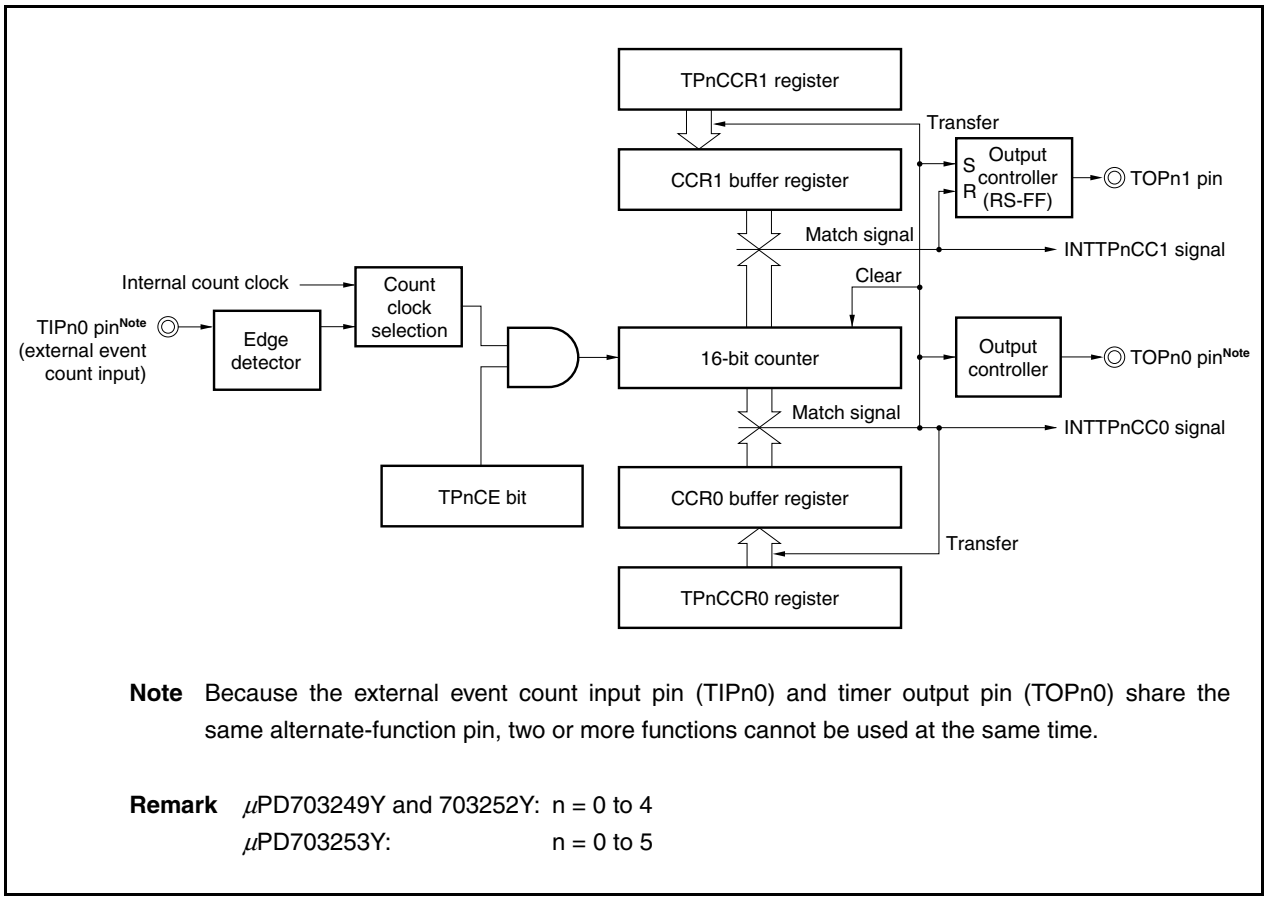
<R> μPD703253Y: n = 0 to 5

7.6.5 PWM output mode (TPnMD2 to TPnMD0 bits = 100)

In the PWM output mode, a PWM waveform is output from the TOPn1 pin when the TPnCTL0.TPnCE bit is set to 1.

In addition, a square wave with a duty factor of 50% with the set value of the TPnCCR0 register + 1 as half its cycle is output from the TOPn0 pin.

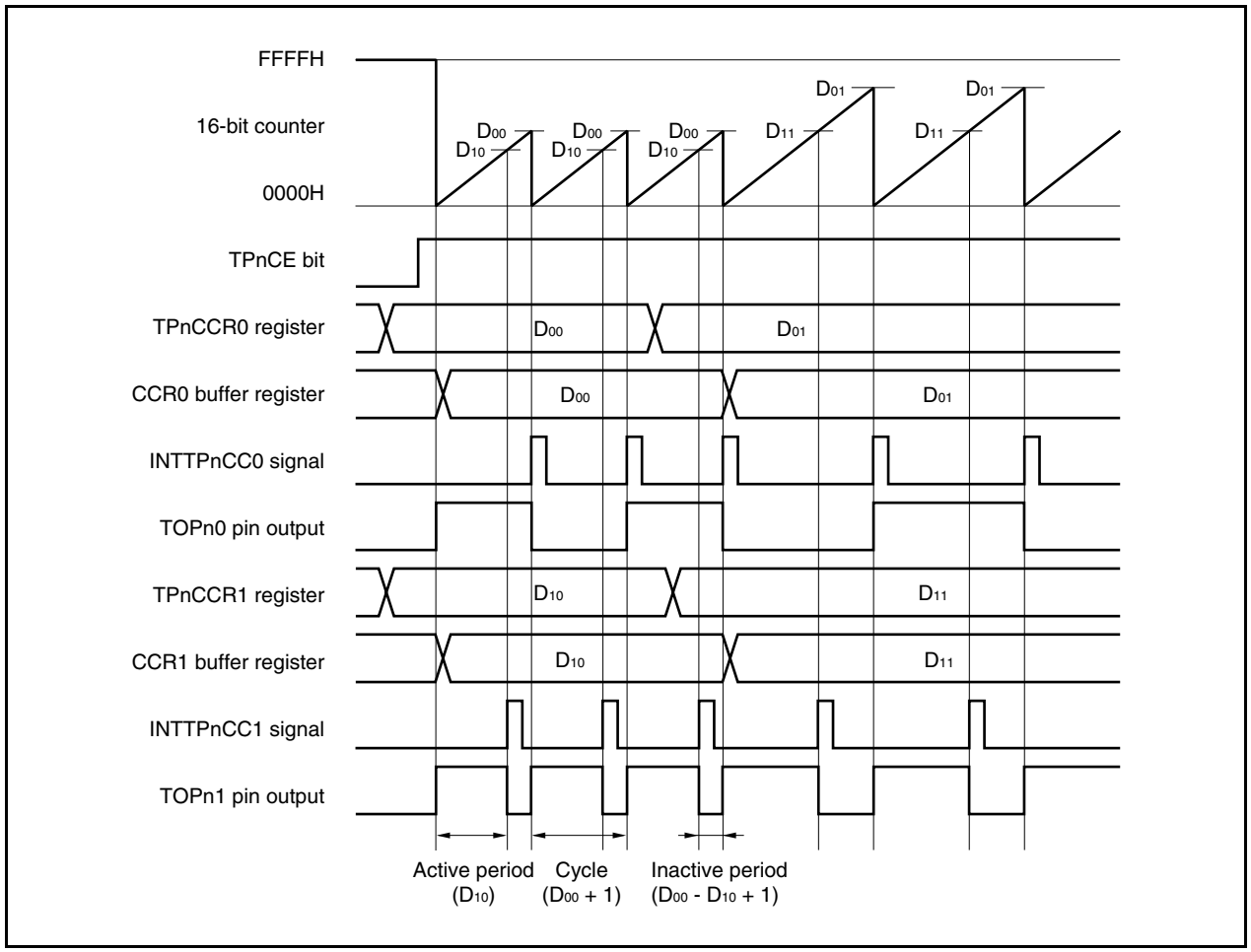
<R> **Figure 7-28. Configuration in PWM Output Mode**



Note Because the external event count input pin (TIPn0) and timer output pin (TOPn0) share the same alternate-function pin, two or more functions cannot be used at the same time.

Remark μPD703249Y and 703252Y: n = 0 to 4
μPD703253Y: n = 0 to 5

Figure 7-29. Basic Timing in PWM Output Mode



When the TPnCE bit is set to 1, the 16-bit counter is cleared from FFFFH to 0000H, starts counting, and outputs a PWM waveform from the TOPn1 pin.

The active level width, cycle, and duty factor of the PWM waveform can be calculated as follows.

Active level width = (Set value of TPnCCR1 register) × Count clock cycle
Cycle = (Set value of TPnCCR0 register + 1) × Count clock cycle
Duty factor = (Set value of TPnCCR1 register)/(Set value of TPnCCR0 register + 1)

The PWM waveform can be changed by rewriting the TPnCCRm register while the counter is operating. The newly written value is reflected when the count value of the 16-bit counter matches the value of the CCR0 buffer register and the 16-bit counter is cleared to 0000H.

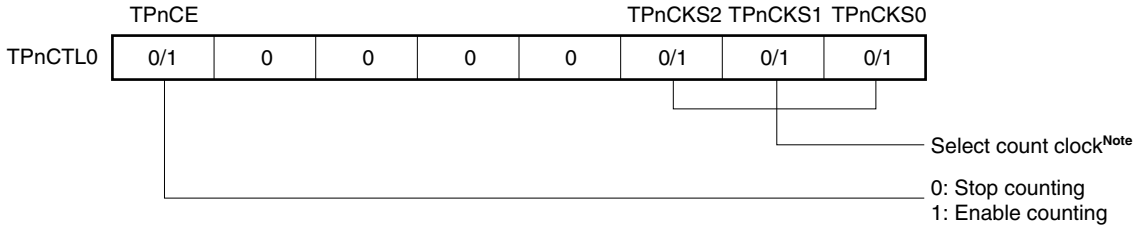
The compare match interrupt request signal INTTPnCC0 is generated when the 16-bit counter counts next time after its count value matches the value of the CCR0 buffer register, and the 16-bit counter is cleared to 0000H. The compare match interrupt request signal INTTPnCC1 is generated when the count value of the 16-bit counter matches the value of the CCR1 buffer register.

The value set to the TPnCCRm register is transferred to the CCRm buffer register when the count value of the 16-bit counter matches the value of the CCRm buffer register and the 16-bit counter is cleared to 0000H.

<R> **Remark** μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1, μPD703253Y: n = 0 to 5, m = 0 or 1

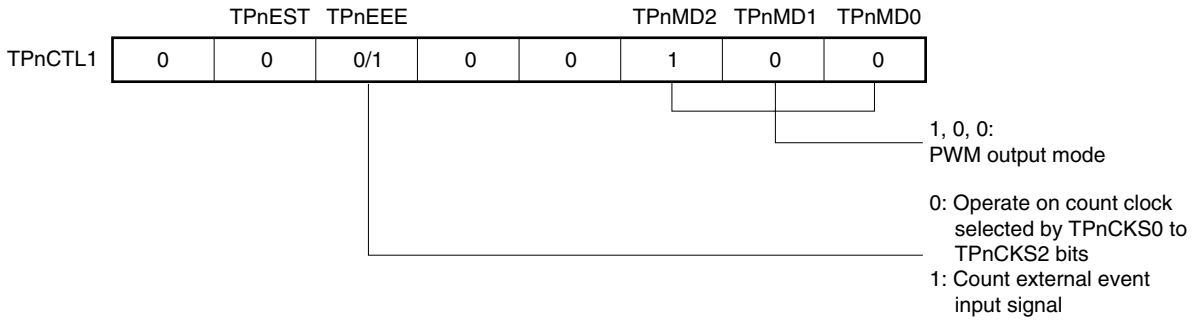
Figure 7-30. Setting of Registers in PWM Output Mode (1/2)

(a) TMPn control register 0 (TPnCTL0)

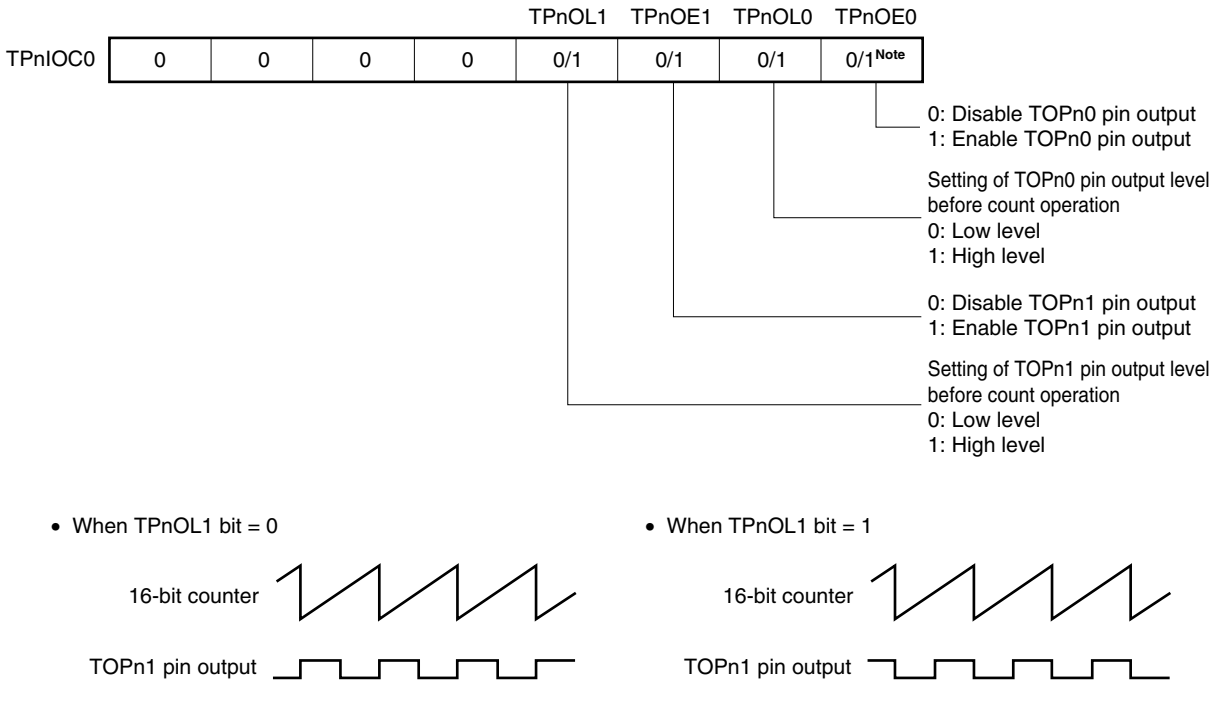


Note The setting is invalid when the TPnCTL1.TPnEEE bit = 1.

(b) TMPn control register 1 (TPnCTL1)



(c) TMPn I/O control register 0 (TPnIOC0)



Note Clear this bit to 0 when the TOPn0 pin is not used in the PWM output mode.

Figure 7-30. Register Setting in PWM Output Mode (2/2)

(d) TMPn I/O control register 2 (TPnIOC2)

					TPnEES1	TPnEES0	TPnETS1	TPnETS0
TPnIOC2	0	0	0	0	0/1	0/1	0	0

Select valid edge of external event count input (TIPn0 pin).

(e) TMPn counter read buffer register (TPnCNT)

The value of the 16-bit counter can be read by reading the TPnCNT register.

(f) TMPn capture/compare registers 0 and 1 (TPnCCR0 and TPnCCR1)

If D_0 is set to the TPnCCR0 register and D_1 to the TPnCCR1 register, the cycle and active level of the PWM waveform are as follows.

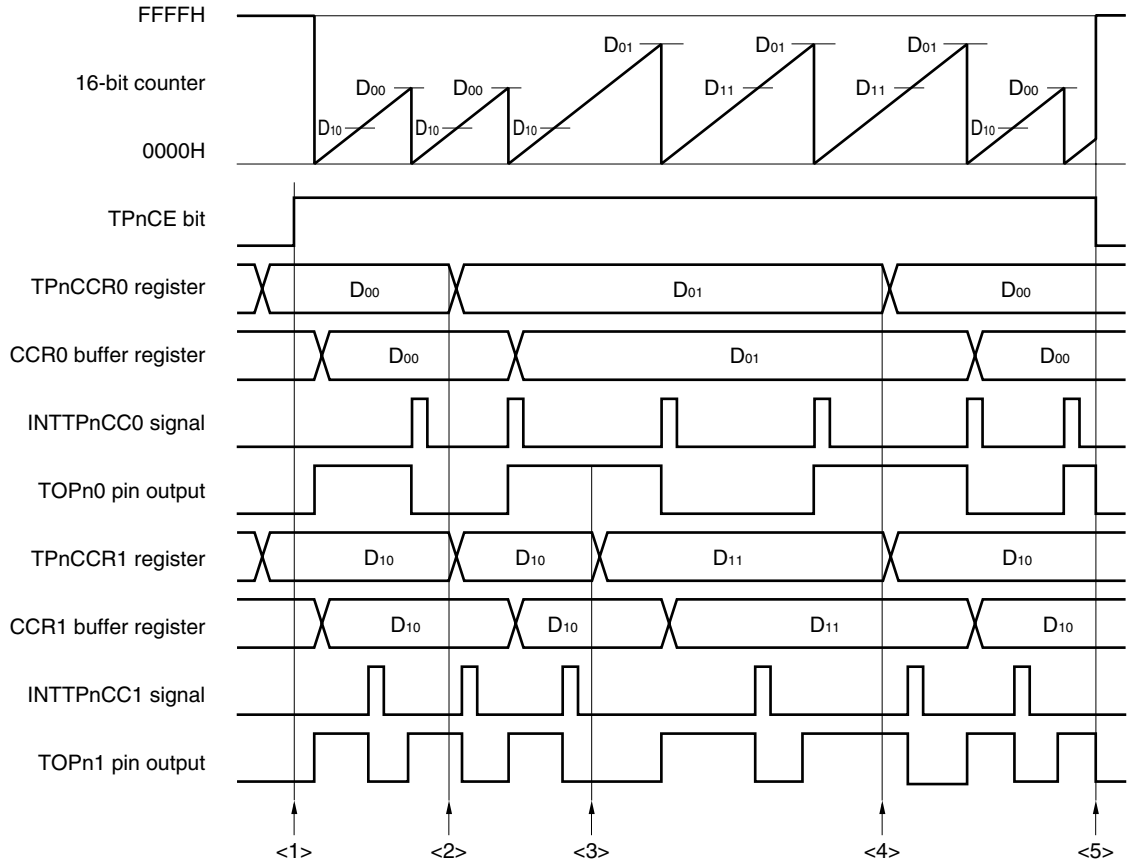
Cycle = $(D_0 + 1) \times$ Count clock cycle
Active level width = $D_1 \times$ Count clock cycle

Remarks 1. TMPn I/O control register 1 (TPnIOC1) and TMPn option register 0 (TPnOPT0) are not used in the PWM output mode.

2. μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

(1) Operation flow in PWM output mode

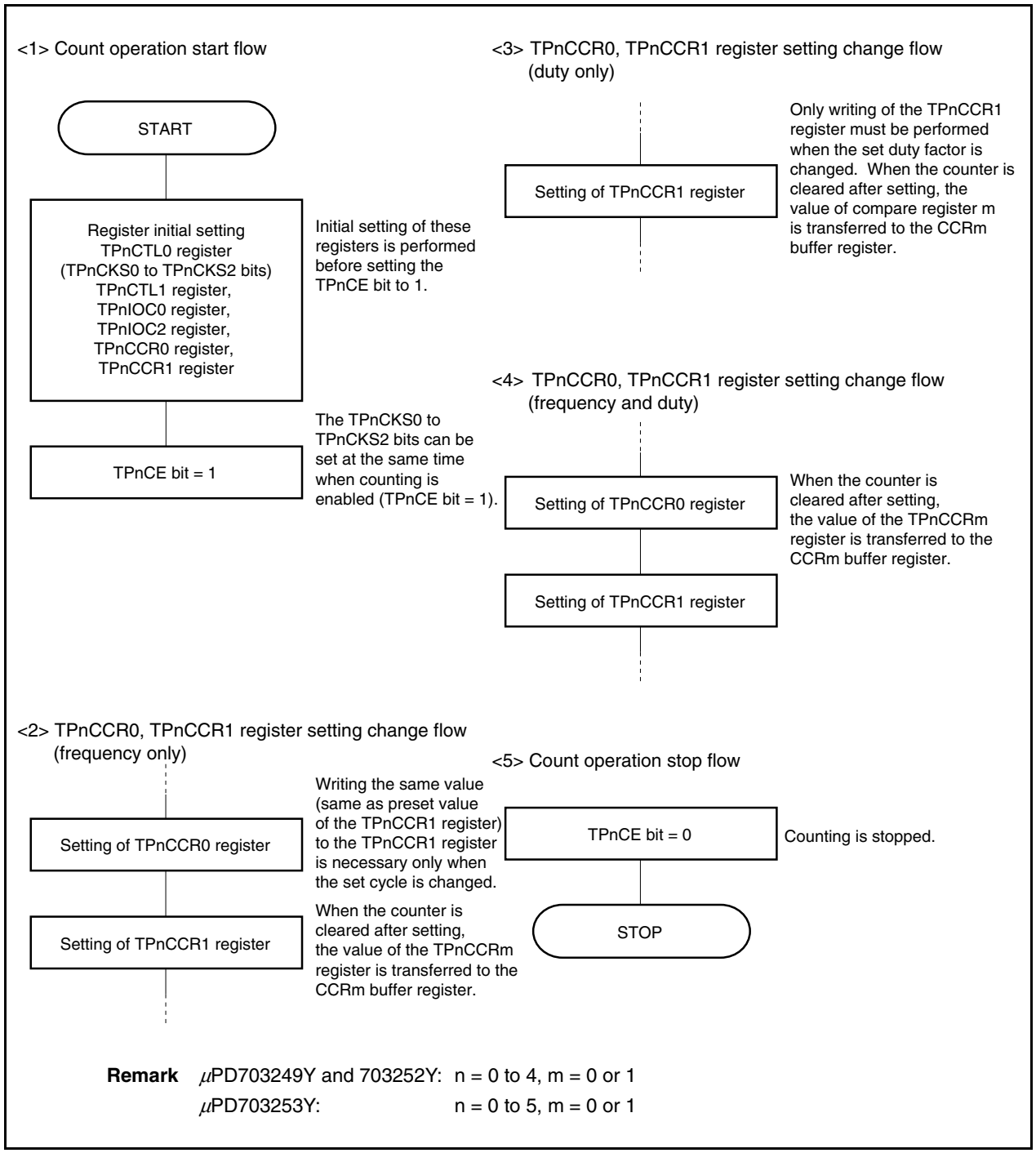
Figure 7-31. Software Processing Flow in PWM Output Mode (1/2)



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Figure 7-31. Software Processing Flow in PWM Output Mode (2/2)



Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

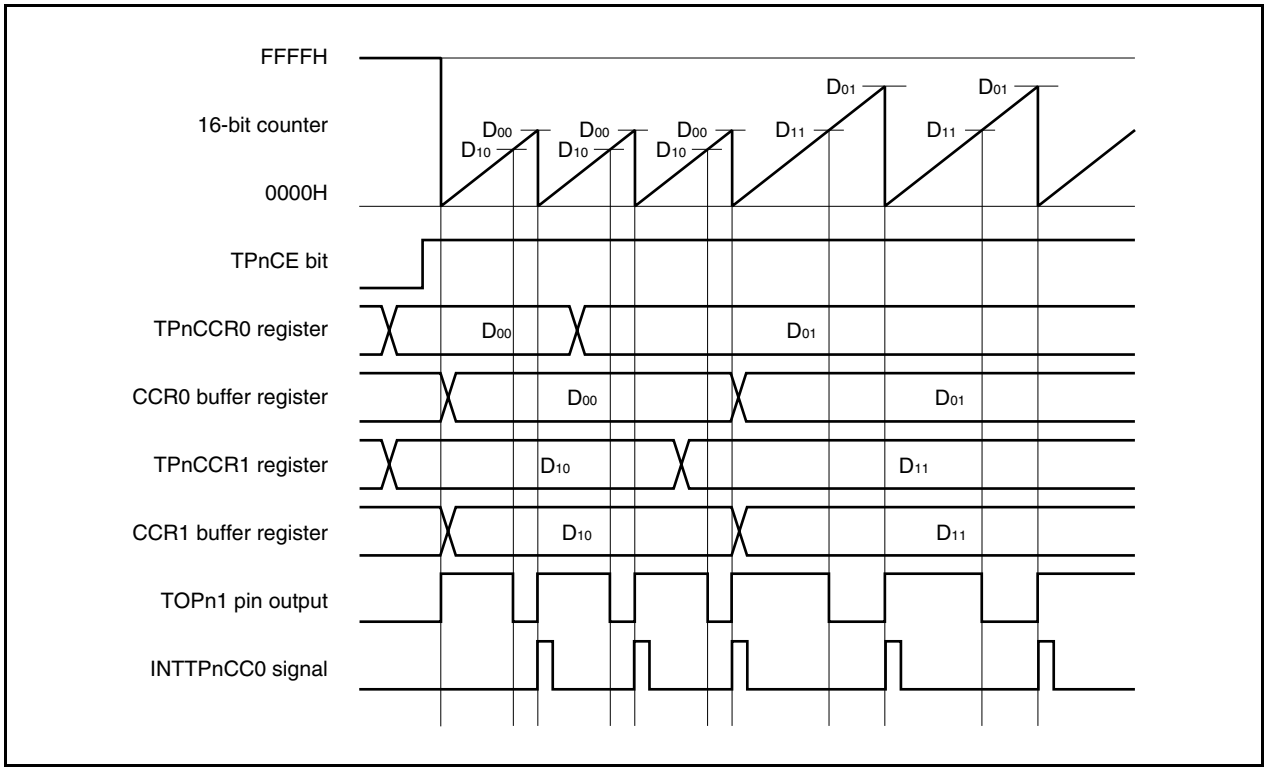
<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(2) PWM output mode operation timing

(a) Changing pulse width during operation

To change the PWM waveform while the counter is operating, write the TPnCCR1 register last.

Rewrite the TPnCCRm register after writing the TPnCCR1 register after the INTTPnCC1 signal is detected.



To transfer data from the TPnCCRm register to the CCRm buffer register, the TPnCCR1 register must be written.

To change both the cycle and active level of the PWM waveform at this time, first set the cycle to the TPnCCR0 register and then set the active level to the TPnCCR1 register.

To change only the cycle of the PWM waveform, first set the cycle to the TPnCCR0 register, and then write the same value (same as preset value of the TPnCCR1 register) to the TPnCCR1 register.

To change only the active level width (duty factor) of the PWM waveform, only the TPnCCR1 register has to be set.

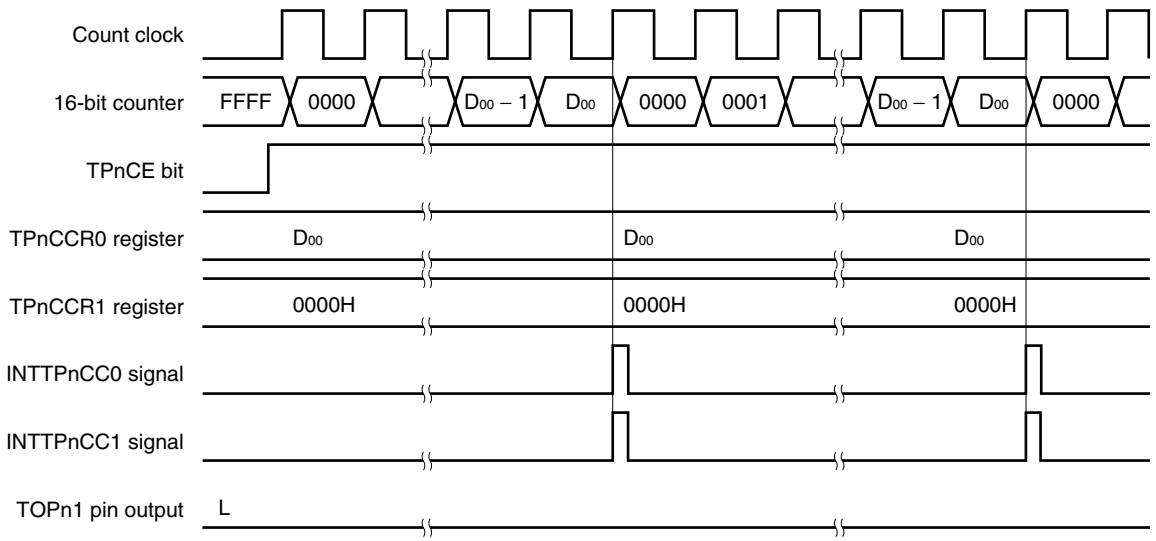
After data is written to the TPnCCR1 register, the value written to the TPnCCRm register is transferred to the CCRm buffer register in synchronization with clearing of the 16-bit counter, and is used as the value compared with the 16-bit counter.

To write the TPnCCR0 or TPnCCR1 register again after writing the TPnCCR1 register once, do so after the INTTPnCC0 signal is generated. Otherwise, the value of the CCRm buffer register may become undefined because the timing of transferring data from the TPnCCRm register to the CCRm buffer register conflicts with writing the TPnCCRm register.

<R> **Remark** μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1, μPD703253Y: n = 0 to 5, m = 0 or 1

(b) 0%/100% output of PWM waveform

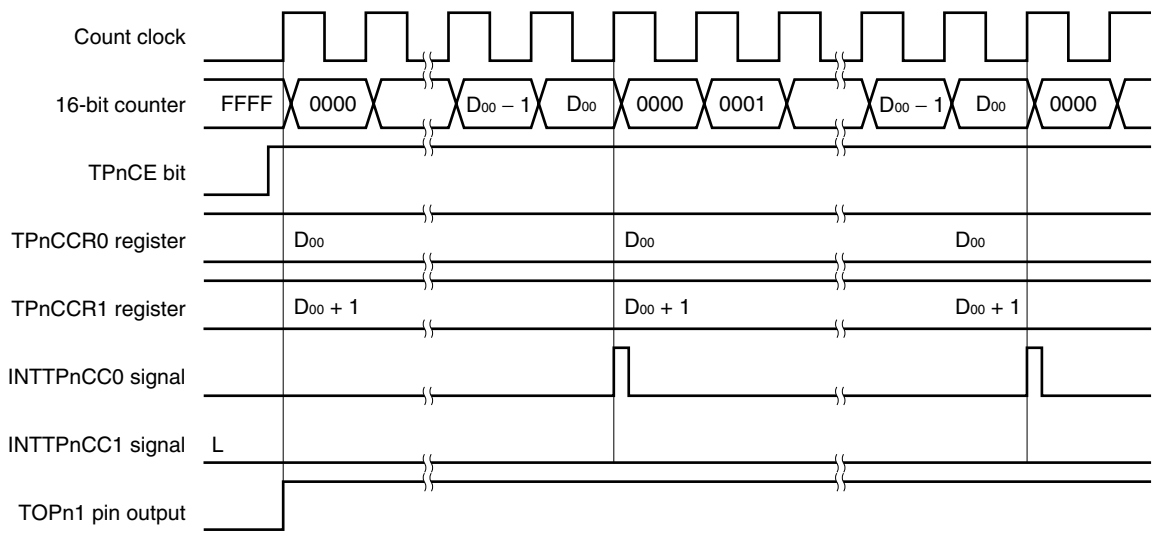
<R> To output a 0% waveform, set the TPnCCR1 register to 0000H. The 16-bit counter is cleared to 0000H and the INTTPnCC0 and INTTPnCC1 signals are generated at the next timing after a match between the count value of the 16-bit counter and the value of the CCR0 buffer register.



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

To output a 100% waveform, set a value of (set value of TPnCCR0 register + 1) to the TPnCCR1 register. If the set value of the TPnCCR0 register is FFFFH, 100% output cannot be produced.

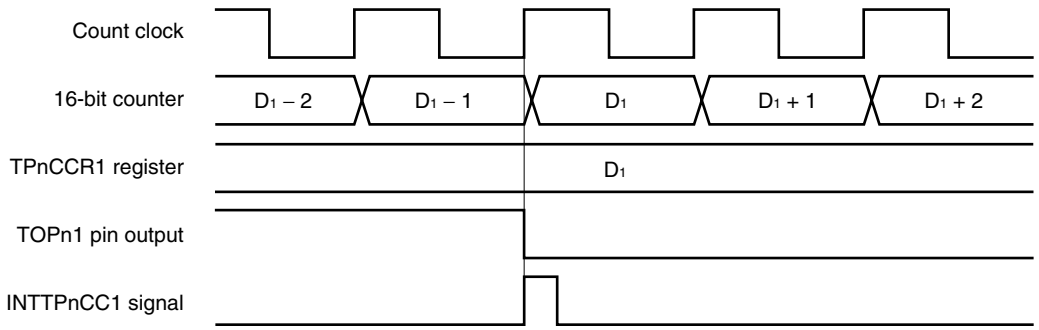


Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

(c) Generation timing of compare match interrupt request signal (INTTPnCC1)

The timing of generation of the INTTPnCC1 signal in the PWM output mode differs from the timing of other mode INTTPnCC1 signals; the INTTPnCC1 signal is generated when the count value of the 16-bit counter matches the value of the TPnCCR1 register.



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

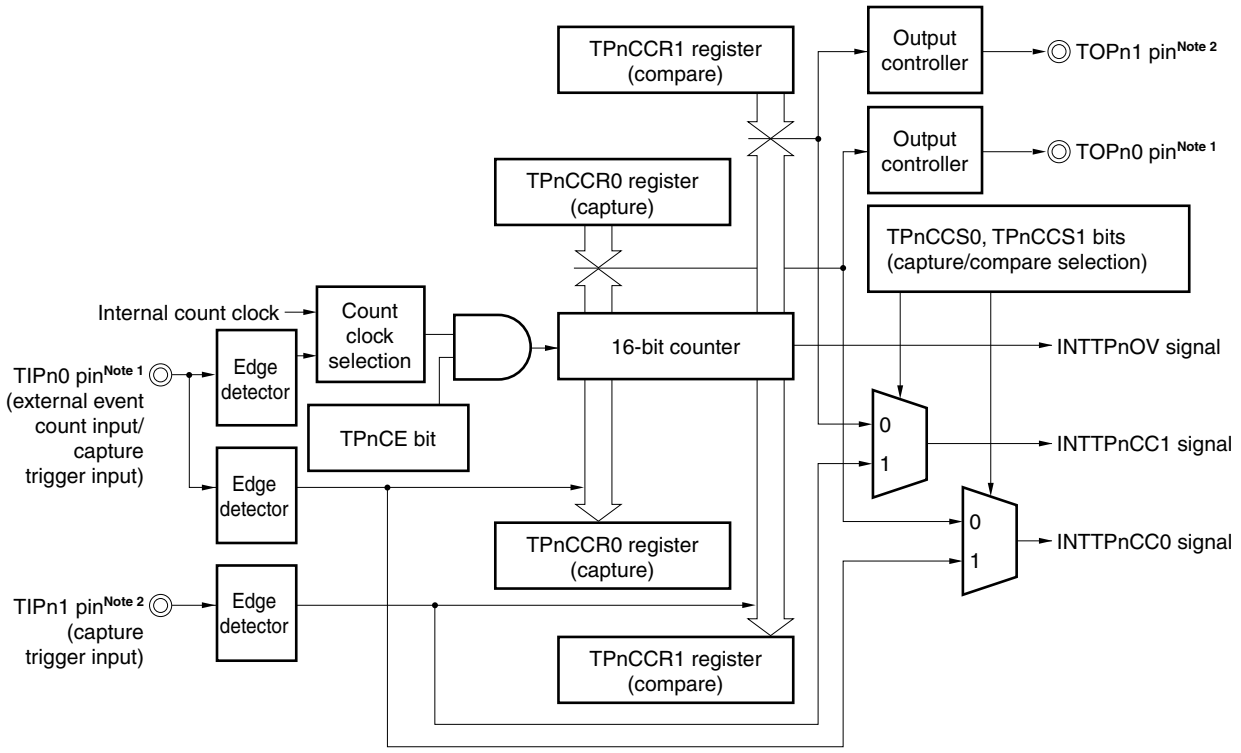
Usually, the INTTPnCC1 signal is generated in synchronization with the next counting up after the count value of the 16-bit counter matches the value of the TPnCCR1 register.

In the PWM output mode, however, it is generated one clock earlier. This is because the timing is changed to match the change timing of the output signal of the TOPn1 pin.

7.6.6 Free-running timer mode (TPnMD2 to TPnMD0 bits = 101)

In the free-running timer mode, 16-bit timer/event counter P starts counting when the TPnCTL0.TPnCE bit is set to 1. At this time, the TPnCCRm register can be used as a compare register or a capture register, depending on the setting of the TPnOPT0.TPnCCS0 and TPnOPT0.TPnCCS1 bits.

Figure 7-32. Configuration in Free-Running Timer Mode



<R> **Notes 1.** Because the external event count input pin (TIPn0), capture trigger input pin (TIPn0), and timer output pin (TOPn0) share the same alternate-function pin, two or more functions cannot be used at the same time.

<R> **2.** Because the capture trigger input pin (TIPn1) and timer output pin (TOPn1) are the same alternate-function pin, two or more functions cannot be used at the same time.

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

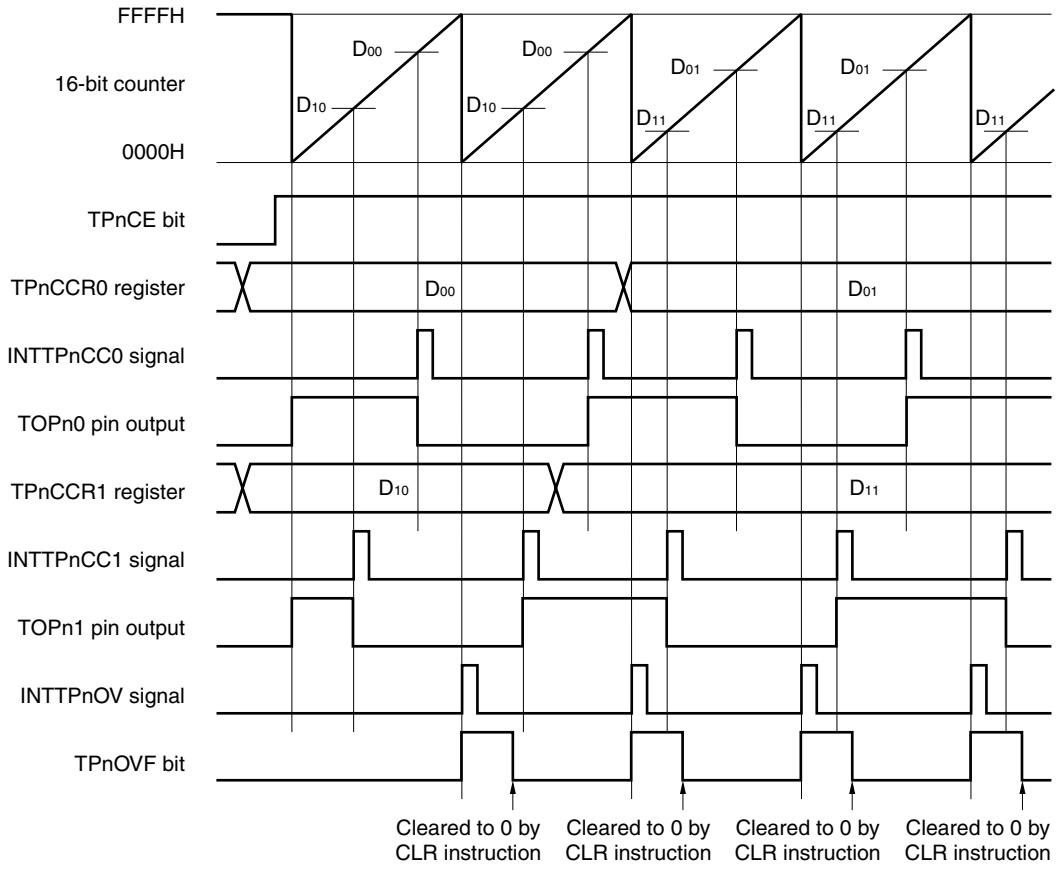
- Compare operation

  When the TPnCE bit is set to 1, 16-bit timer/event counter P starts counting, and the output signals of the TOPn0 and TOPn1 pins are inverted. When the count value of the 16-bit counter later matches the set value of the TPnCCRm register, a compare match interrupt request signal (INTTPnCCm) is generated, and the output signals of the TOPnm pins are inverted.

  The 16-bit counter continues counting in synchronization with the count clock. When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTPnOV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TPnOPT0.TPnOVF bit) is also set to 1. Confirm that the overflow flag is set to 1 and then clear it to 0 by executing the CLR instruction via software.

  The TPnCCRm register can be rewritten while the counter is operating. If it is rewritten, the new value is reflected at that time by anytime write, and compared with the count value.

Figure 7-33. Basic Timing in Free-Running Timer Mode (Compare Function)



Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

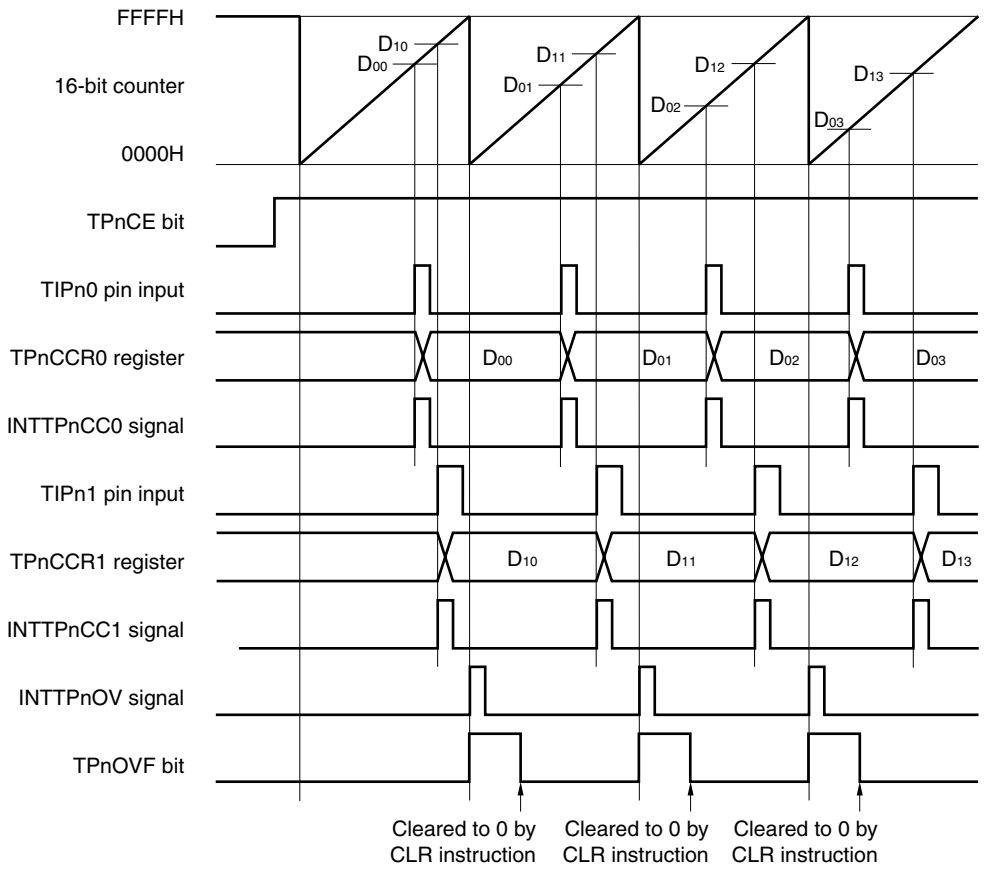
<R> μPD703253Y: n = 0 to 5, m = 0 or 1

- Capture operation

  When the TPnCE bit is set to 1, the 16-bit counter starts counting. When the valid edge input to the TIPnm pin is detected, the count value of the 16-bit counter is stored in the TPnCCRm register, and a capture interrupt request signal (INTTPnCCm) is generated.

  The 16-bit counter continues counting in synchronization with the count clock. When it counts up to FFFFH, it generates an overflow interrupt request signal (INTTPnOV) at the next clock, is cleared to 0000H, and continues counting. At this time, the overflow flag (TPnOPT0.TPnOVF bit) is also set to 1. Confirm that the overflow flag is set to 1 and then clear it to 0 by executing the CLR instruction via software.

Figure 7-34. Basic Timing in Free-Running Timer Mode (Capture Function)

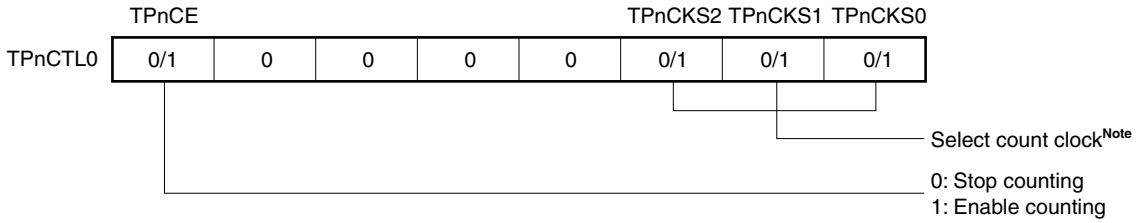


Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Figure 7-35. Register Setting in Free-Running Timer Mode (1/3)

(a) TMPn control register 0 (TPnCTL0)



Note The setting is invalid when the TPnCTL1.TPnEEE bit = 1

(b) TMPn control register 1 (TPnCTL1)

(c) TMPn I/O control register 0 (TPnIOC0)

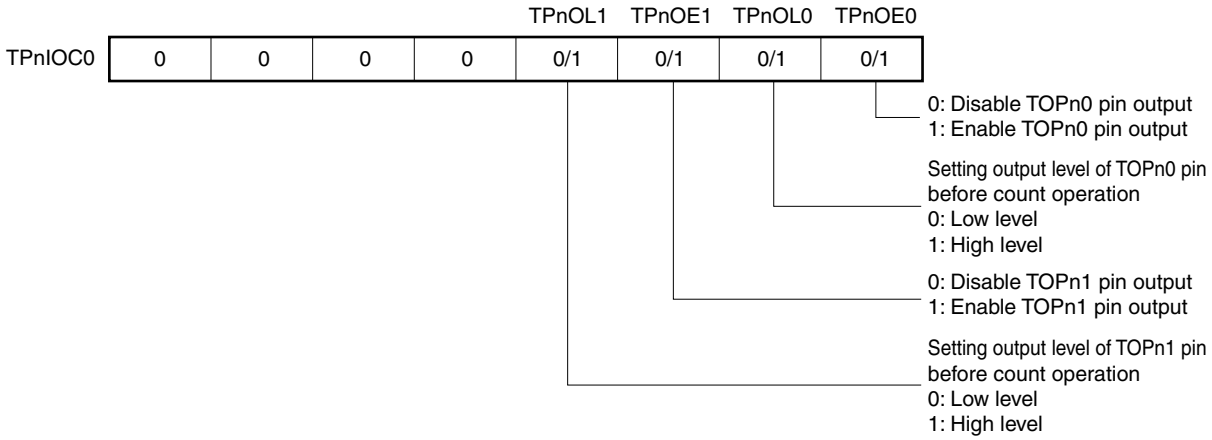
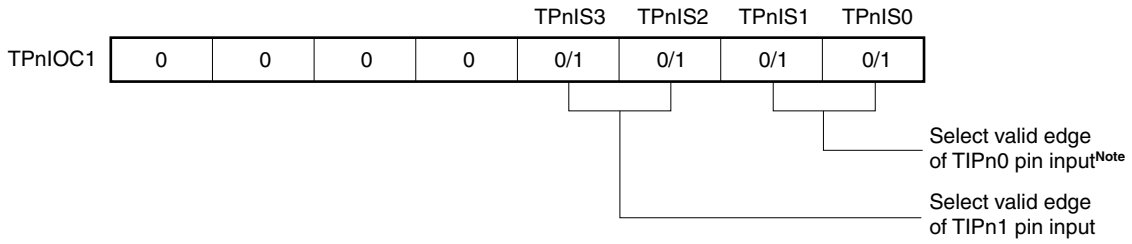


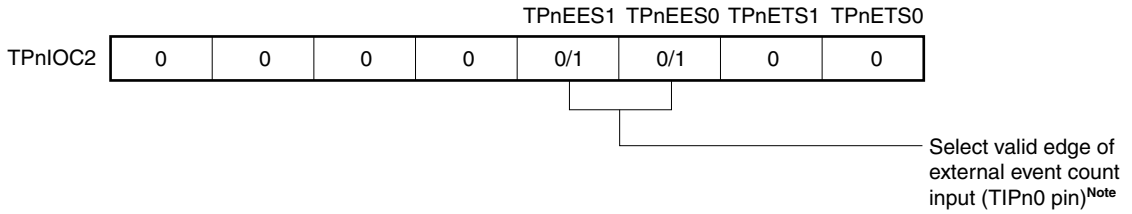
Figure 7-35. Register Setting in Free-Running Timer Mode (2/3)

(d) TMPn I/O control register 1 (TPnIOC1)



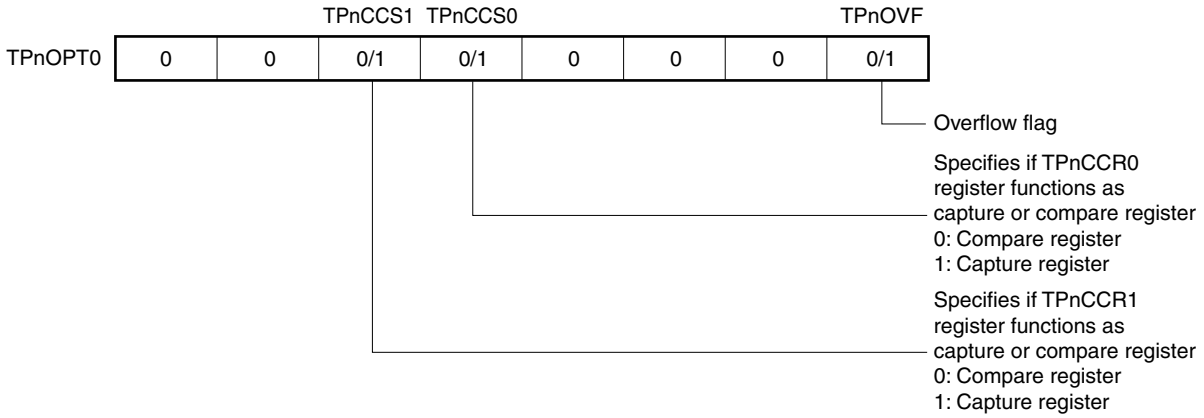
<R> **Note** Set the valid edge selection of the unused alternate external input signals to “No edge detection”.

(e) TMPn I/O control register 2 (TPnIOC2)



<R> **Note** Set the valid edge selection of the unused alternate external input signals to “No edge detection”.

(f) TMPn option register 0 (TPnOPT0)



(g) TMPn counter read buffer register (TPnCNT)

The value of the 16-bit counter can be read by reading the TPnCNT register.

Figure 7-35. Register Setting in Free-Running Timer Mode (3/3)

(h) TMPn capture/compare registers 0 and 1 (TPnCCR0 and TPnCCR1)

These registers function as capture registers or compare registers depending on the setting of the TPnOPT0.TPnCCSm bit.

When the registers function as capture registers, they store the count value of the 16-bit counter when the valid edge input to the TIPnm pin is detected.

When the registers function as compare registers and when D_m is set to the TPnCCRm register, the INTTPnCCm signal is generated when the counter reaches (D_m + 1), and the output signal of the TOPnm pin is inverted.

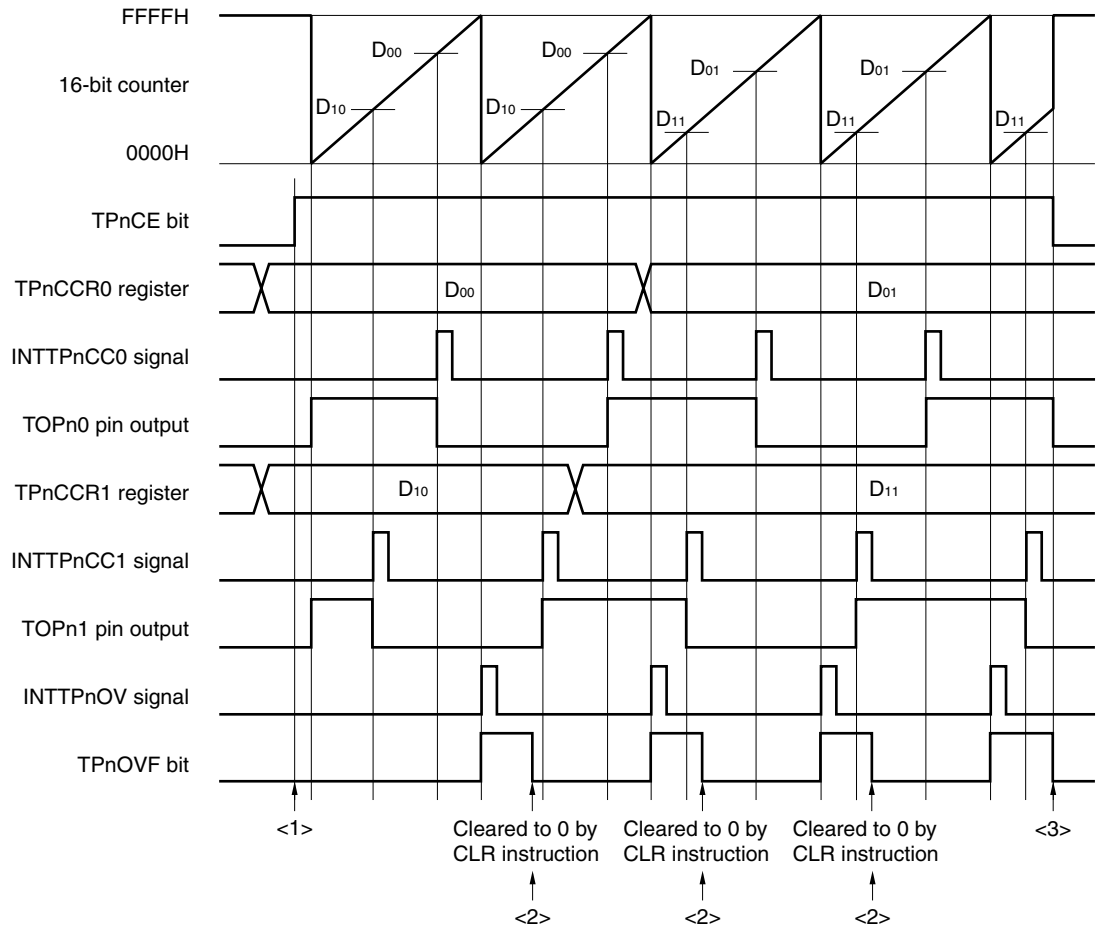
Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(1) Operation flow in free-running timer mode

(a) When using capture/compare register as compare register

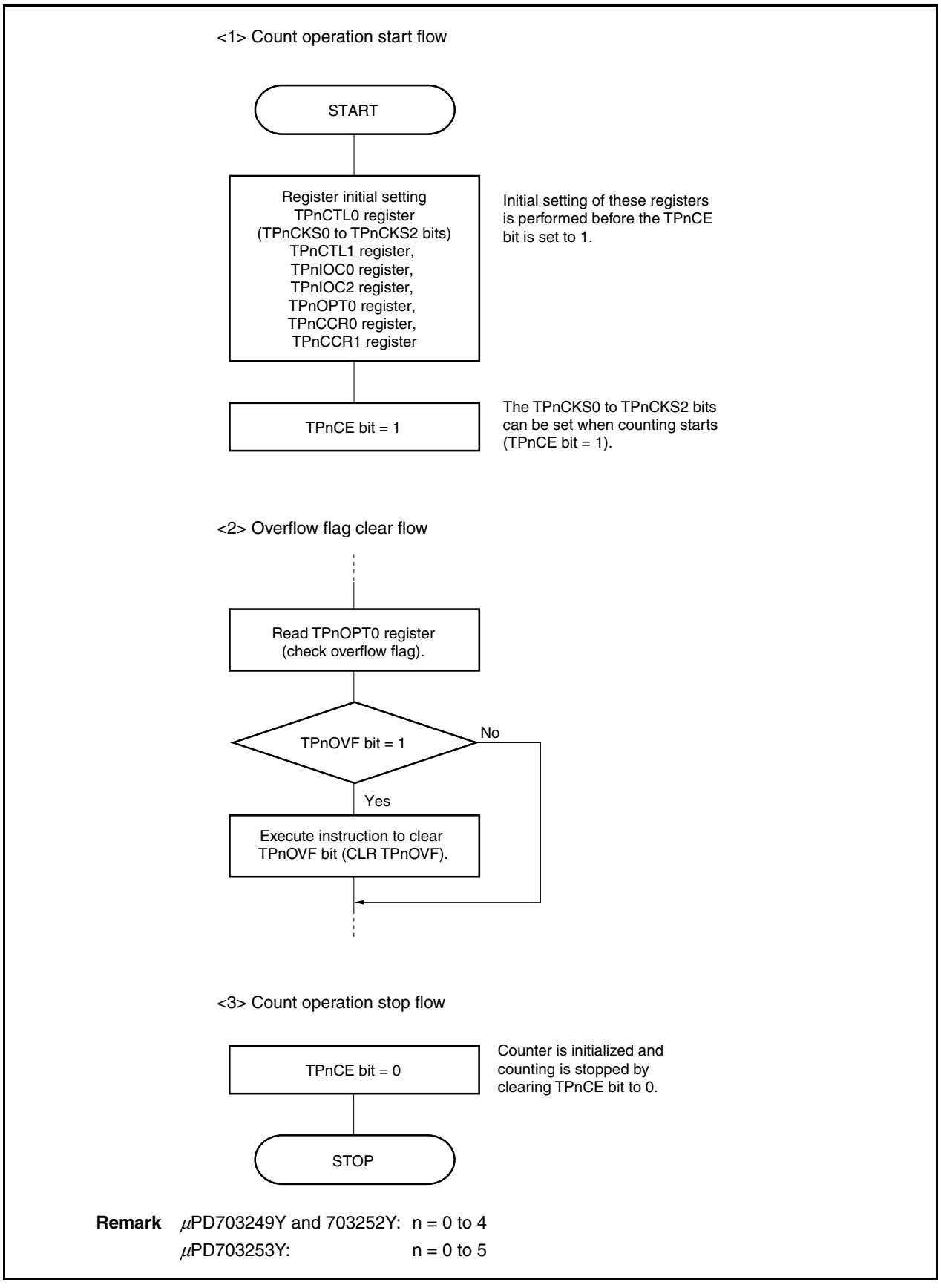
Figure 7-36. Software Processing Flow in Free-Running Timer Mode (Compare Function) (1/2)



Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

Figure 7-36. Software Processing Flow in Free-Running Timer Mode (Compare Function) (2/2)

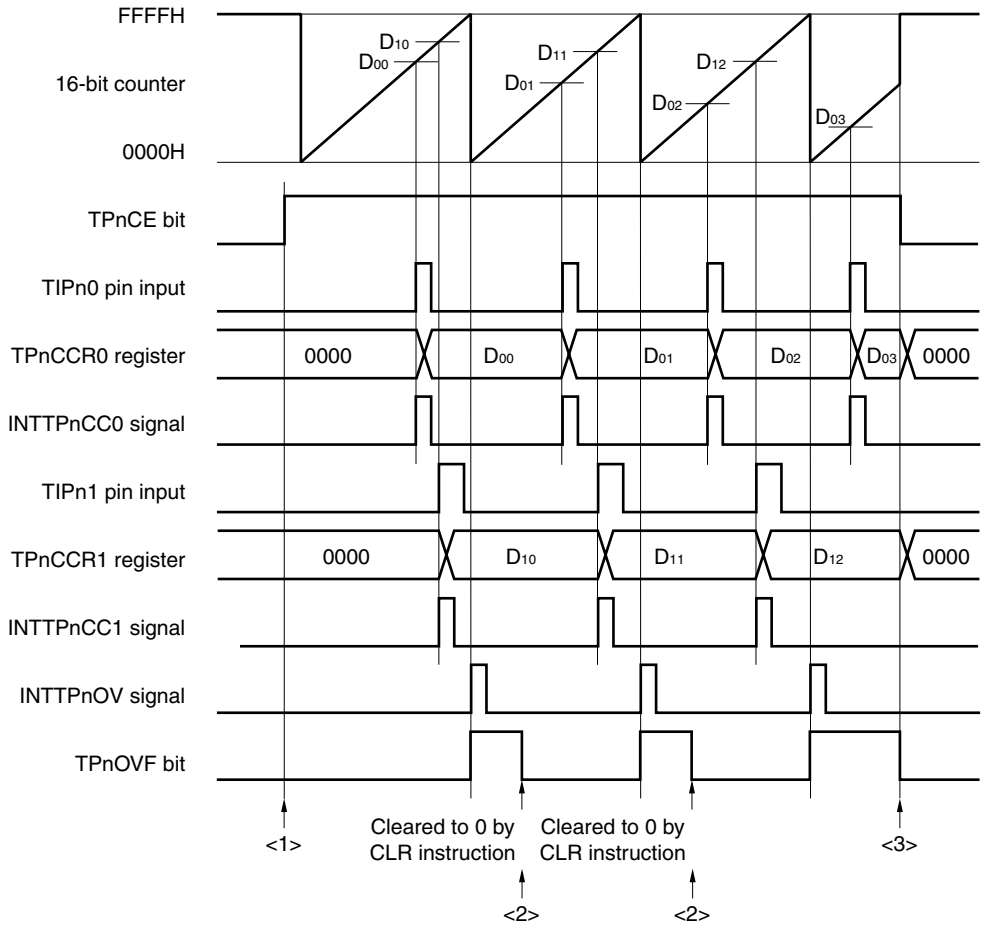


Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

(b) When using capture/compare register as capture register

Figure 7-37. Software Processing Flow in Free-Running Timer Mode (Capture Function) (1/2)

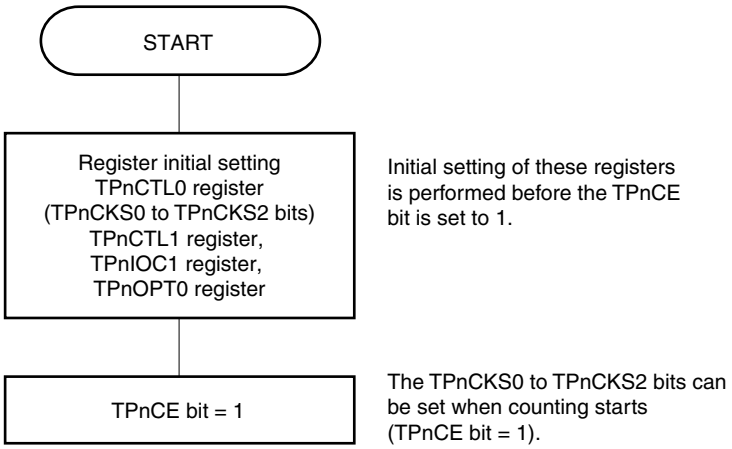


Remark μPD703249Y and 703252Y: n = 0 to 4

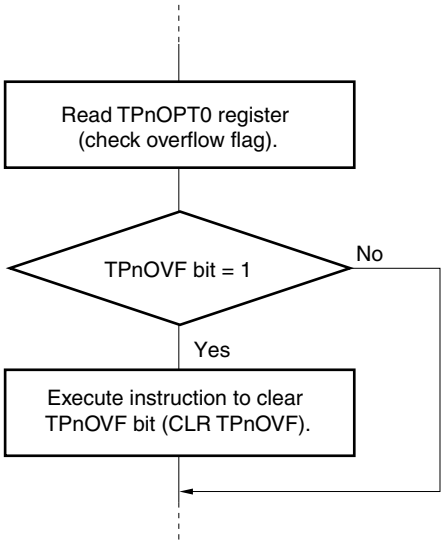
<R> μPD703253Y: n = 0 to 5

Figure 7-37. Software Processing Flow in Free-Running Timer Mode (Capture Function) (2/2)

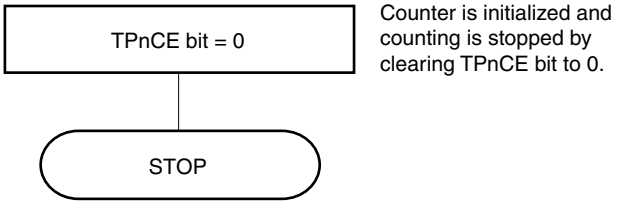
<1> Count operation start flow



<2> Overflow flag clear flow



<3> Count operation stop flow



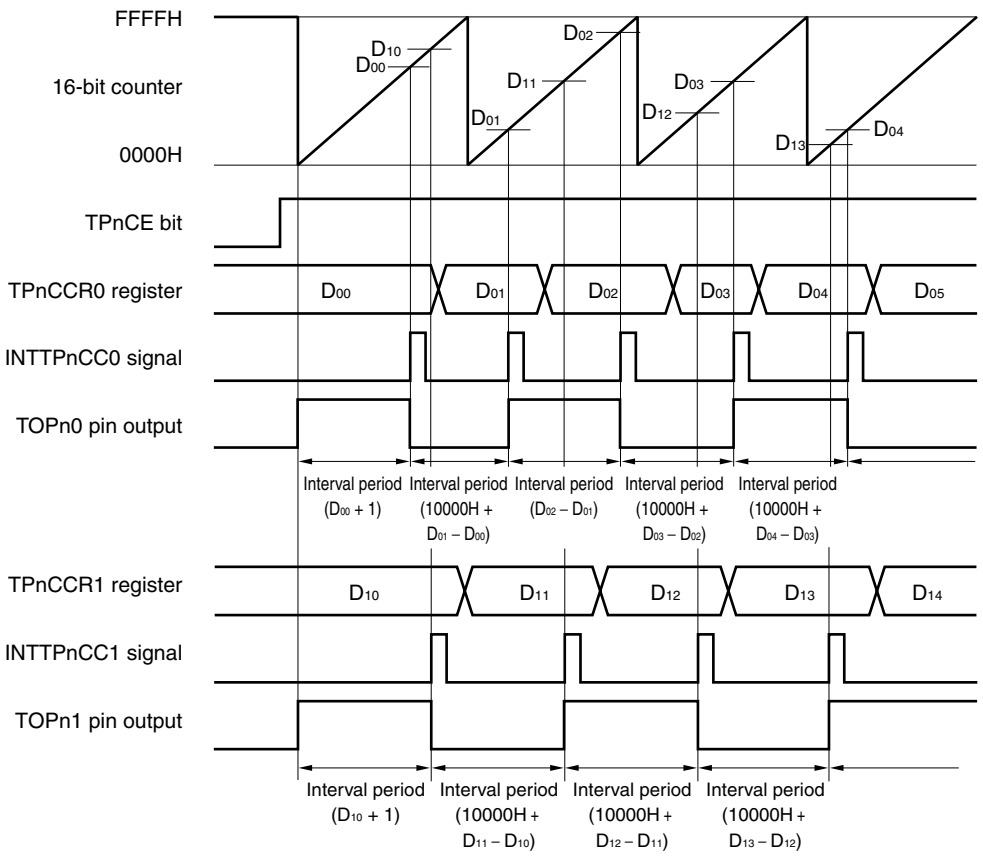
Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

(2) Operation timing in free-running timer mode

(a) Interval operation with compare register

When 16-bit timer/event counter P is used as an interval timer with the TPnCCRm register used as a compare register, software processing is necessary for setting a comparison value to generate the next interrupt request signal each time the INTTPnCCm signal has been detected.



When performing an interval operation in the free-running timer mode, two intervals can be set with one channel.

To perform the interval operation, the value of the corresponding TPnCCRm register must be re-set in the interrupt servicing that is executed when the INTTPnCCm signal is detected.

The set value for re-setting the TPnCCRm register can be calculated by the following expression, where "D_m" is the interval period.

Compare register default value: $D_m - 1$

Value set to compare register second and subsequent time: Previous set value + D_m

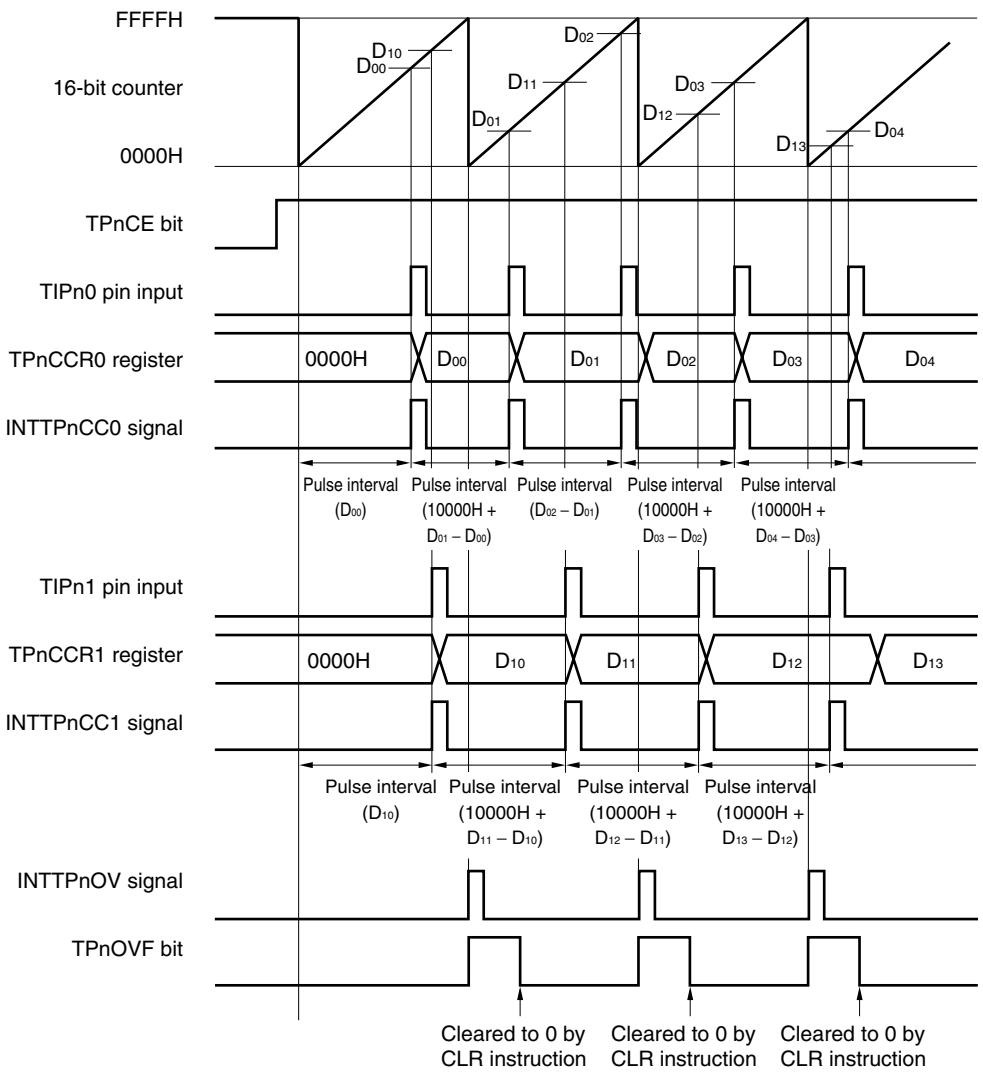
(If the calculation result is greater than FFFFH, subtract 10000H from the result and set this value to the register.)

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(b) Pulse width measurement with capture register

When pulse width measurement is performed with the TPnCCRm register used as a capture register, software processing is necessary for reading the capture register each time the INTTPnCCm signal has been detected and for calculating an interval.



When executing pulse width measurement in the free-running timer mode, two pulse widths can be measured with one channel.

To measure a pulse width, the pulse width can be calculated by reading the value of the TPnCCRm register in synchronization with the INTTPnCCm signal, and calculating the difference between the read value and the previously read value.

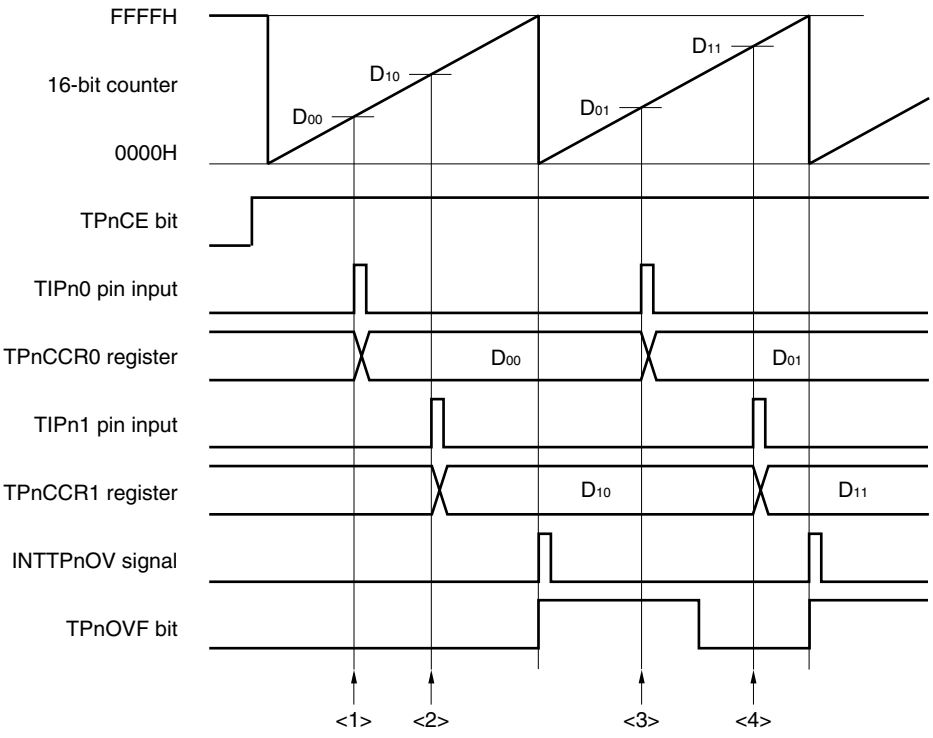
Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(c) Processing of overflow when two capture registers are used

Care must be exercised in processing the overflow flag when two capture registers are used. First, an example of incorrect processing is shown below.

Example of incorrect processing when two capture registers are used



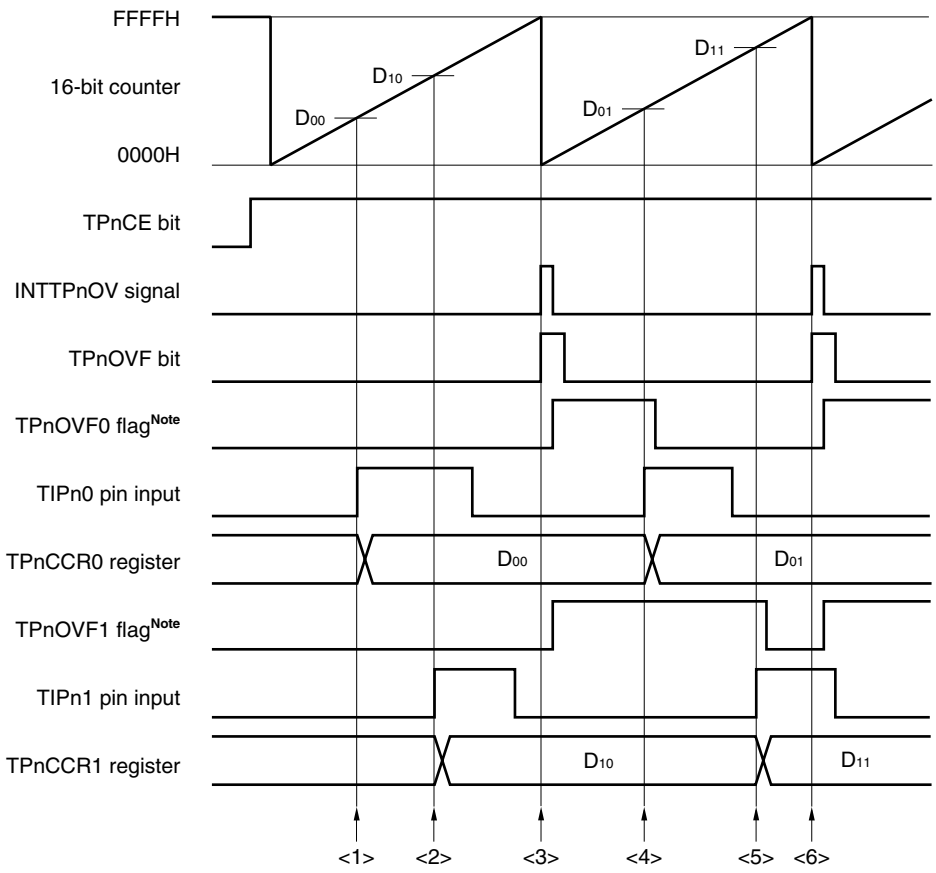
The following problem may occur when two pulse widths are measured in the free-running timer mode.

<1> Read the TPnCCR0 register (setting of the default value of the TIPn0 pin input).
<2> Read the TPnCCR1 register (setting of the default value of the TIPn1 pin input).
<3> Read the TPnCCR0 register.
Read the overflow flag. If the overflow flag is 1, clear it to 0.
Because the overflow flag is 1, the pulse width can be calculated by (10000H + D_{01} – D_{00}).
<4> Read the TPnCCR1 register.
Read the overflow flag. Because the flag is cleared in <3>, 0 is read.
Because the overflow flag is 0, the pulse width can be calculated by (D_{11} – D_{10}) (incorrect).

Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

When two capture registers are used, and if the overflow flag is cleared to 0 by one capture register, the other capture register may not obtain the correct pulse width.
Use software when using two capture registers. An example of how to use software is shown below.

Example when two capture registers are used (using overflow interrupt)



Note The TPnOVF0 and TPnOVF1 flags are set on the internal RAM by software.

<1> Read the TPnCCR0 register (setting of the default value of the TIPn0 pin input).

<2> Read the TPnCCR1 register (setting of the default value of the TIPn1 pin input).

<3> An overflow occurs. Set the TPnOVF0 and TPnOVF1 flags to 1 in the overflow interrupt servicing, and clear the overflow flag to 0.

<4> Read the TPnCCR0 register.
Read the TPnOVF0 flag. If the TPnOVF0 flag is 1, clear it to 0.
Because the TPnOVF0 flag is 1, the pulse width can be calculated by (10000H + D_{01} – D_{00}).

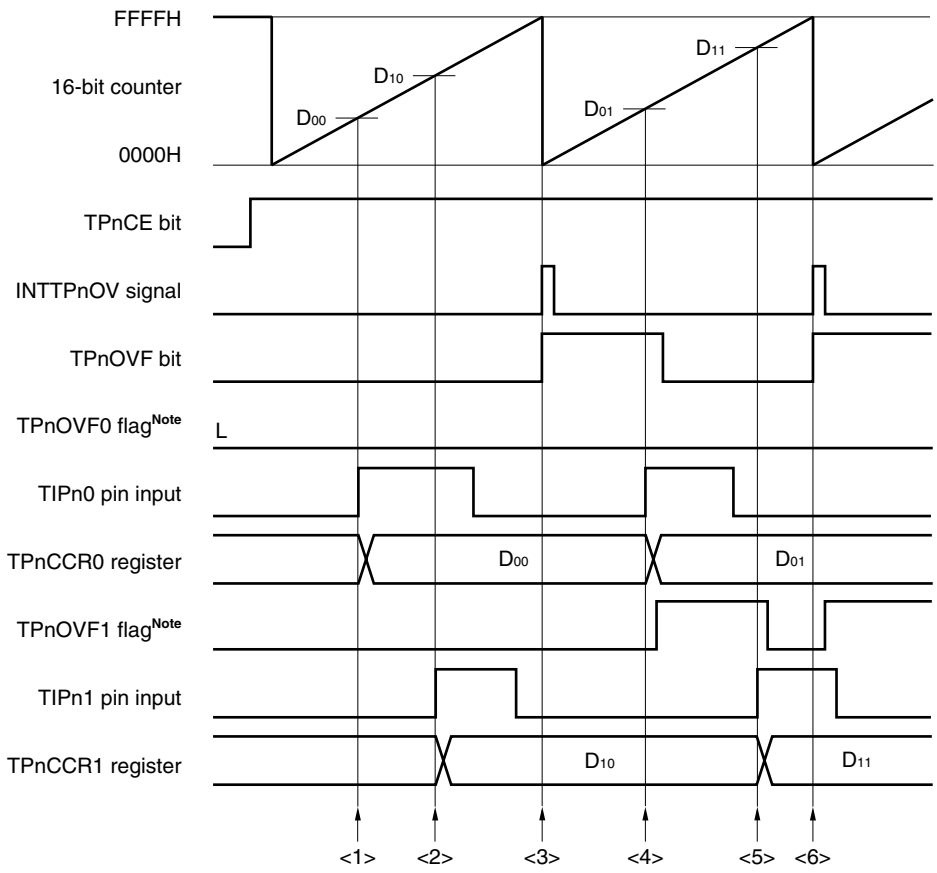
<5> Read the TPnCCR1 register.
Read the TPnOVF1 flag. If the TPnOVF1 flag is 1, clear it to 0 (the TPnOVF0 flag is cleared in <4>, and the TPnOVF1 flag remains 1).
Because the TPnOVF1 flag is 1, the pulse width can be calculated by (10000H + D_{11} – D_{10}) (correct).

<6> Same as <3>

Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

(2/2)

Example when two capture registers are used (without using overflow interrupt)



Note The TPnOVF0 and TPnOVF1 flags are set on the internal RAM by software.

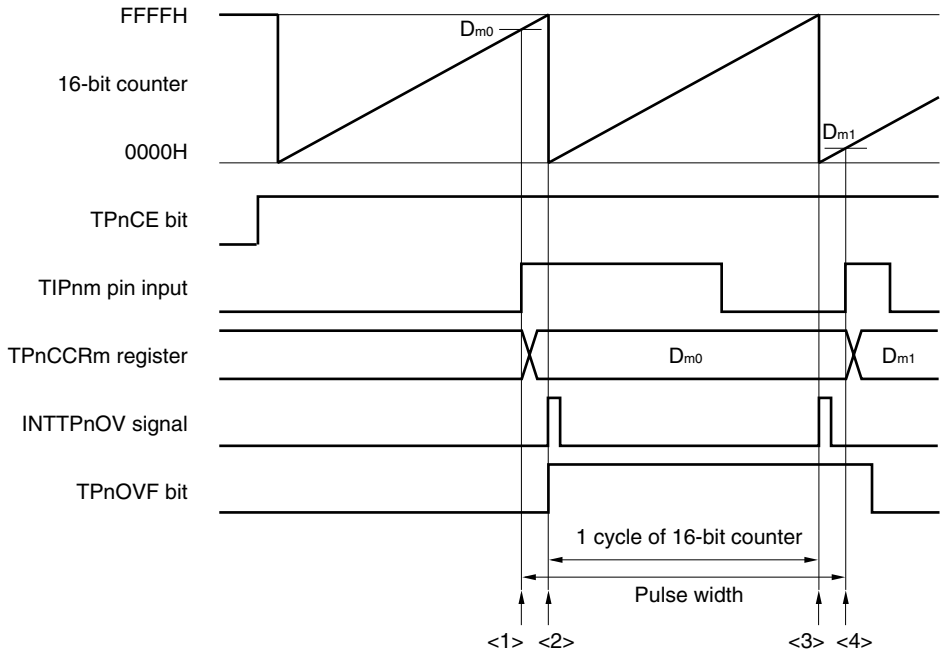
<1> Read the TPnCCR0 register (setting of the default value of the TIPn0 pin input).
<2> Read the TPnCCR1 register (setting of the default value of the TIPn1 pin input).
<3> An overflow occurs. Nothing is done by software.
<4> Read the TPnCCR0 register.
Read the overflow flag. If the overflow flag is 1, set only the TPnOVF1 flag to 1, and clear the overflow flag to 0.
Because the overflow flag is 1, the pulse width can be calculated by (10000H + D_{01} – D_{00}).
<5> Read the TPnCCR1 register.
Read the overflow flag. Because the overflow flag is cleared in <4>, 0 is read.
Read the TPnOVF1 flag. If the TPnOVF1 flag is 1, clear it to 0.
Because the TPnOVF1 flag is 1, the pulse width can be calculated by (10000H + D_{11} – D_{10}) (correct).
<6> Same as <3>

Remark μPD703249Y and 703252Y: n = 0 to 4
<R> μPD703253Y: n = 0 to 5

(d) Processing of overflow if capture trigger interval is long

If the pulse width is greater than one cycle of the 16-bit counter, care must be exercised because an overflow may occur more than once from the first capture trigger to the next. First, an example of incorrect processing is shown below.

Example of incorrect processing when capture trigger interval is long



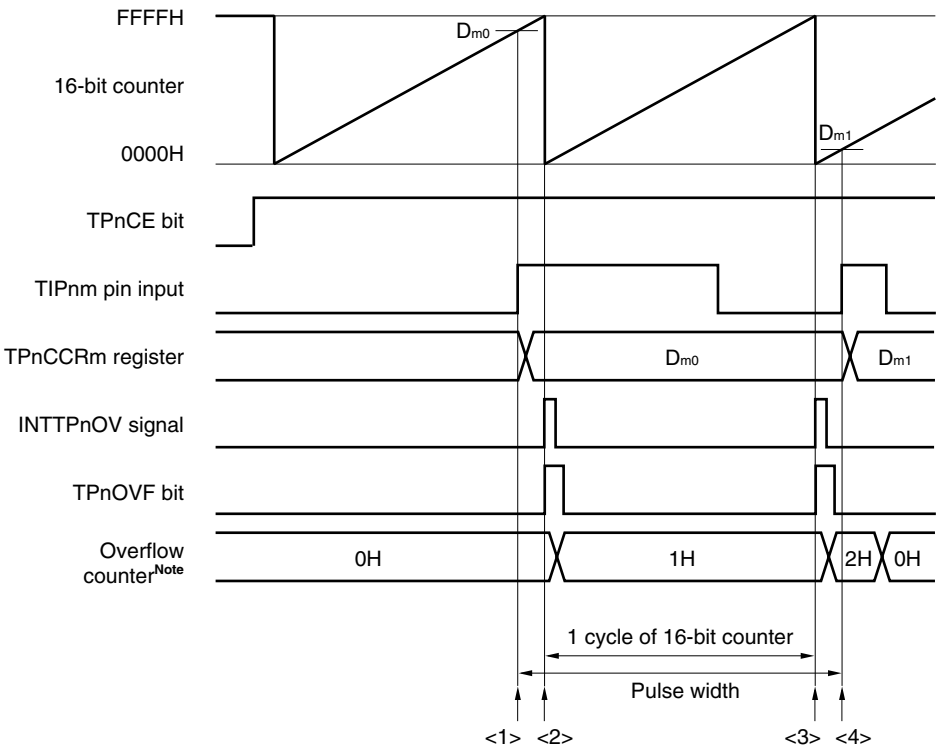
The following problem may occur when long pulse width is measured in the free-running timer mode.

<1> Read the TPnCCRm register (setting of the default value of the TIPnm pin input).
<2> An overflow occurs. Nothing is done by software.
<3> An overflow occurs a second time. Nothing is done by software.
<4> Read the TPnCCRm register.
Read the overflow flag. If the overflow flag is 1, clear it to 0.
Because the overflow flag is 1, the pulse width can be calculated by (10000H + D_{m1} – D_{m0}) (incorrect).
Actually, the pulse width must be (20000H + D_{m1} – D_{m0}) because an overflow occurs twice.

<R> **Remark** μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1, μPD703253Y: n = 0 to 5, m = 0 or 1

If an overflow occurs twice or more when the capture trigger interval is long, the correct pulse width may not be obtained.
If the capture trigger interval is long, slow the count clock to lengthen one cycle of the 16-bit counter, or use software. An example of how to use software is shown next.

Example when capture trigger interval is long



Note The overflow counter is set arbitrarily by software on the internal RAM.

<1> Read the TPnCCRm register (setting of the default value of the TIPnm pin input).

<2> An overflow occurs. Increment the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<3> An overflow occurs a second time. Increment (+1) the overflow counter and clear the overflow flag to 0 in the overflow interrupt servicing.

<4> Read the TPnCCRm register.

Read the overflow counter.

→ When the overflow counter is "N", the pulse width can be calculated by ($N \times 10000H + D_{m1} - D_{m0}$).

In this example, the pulse width is ($20000H + D_{m1} - D_{m0}$) because an overflow occurs twice.

Clear the overflow counter (0H).

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

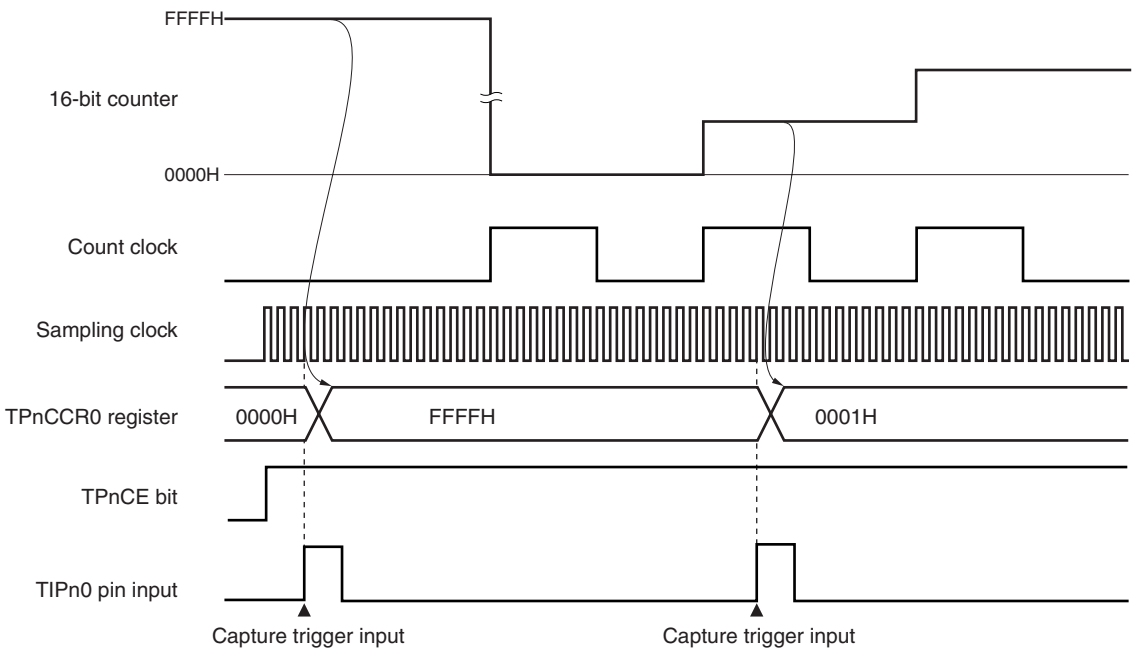
<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(e) Clearing overflow flag

The overflow flag can be cleared to 0 by clearing the TPnOVF bit to 0 with the CLR instruction after reading the TPnOVF bit when it is 1 and by writing 8-bit data (bit 0 is 0) to the TPnOPT0 register after reading the TPnOVF bit when it is 1.

<R> **(3) Note on capture operation**

If the capture operation is used and if a slow clock is selected as the count clock, FFFFH, not 0000H, may be captured to the TPnCCRm register if the capture trigger is input immediately after the TPnCTL0.TPnCE bit is set to 1.



Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1
μPD703253Y: n = 0 to 5, m = 0 or 1

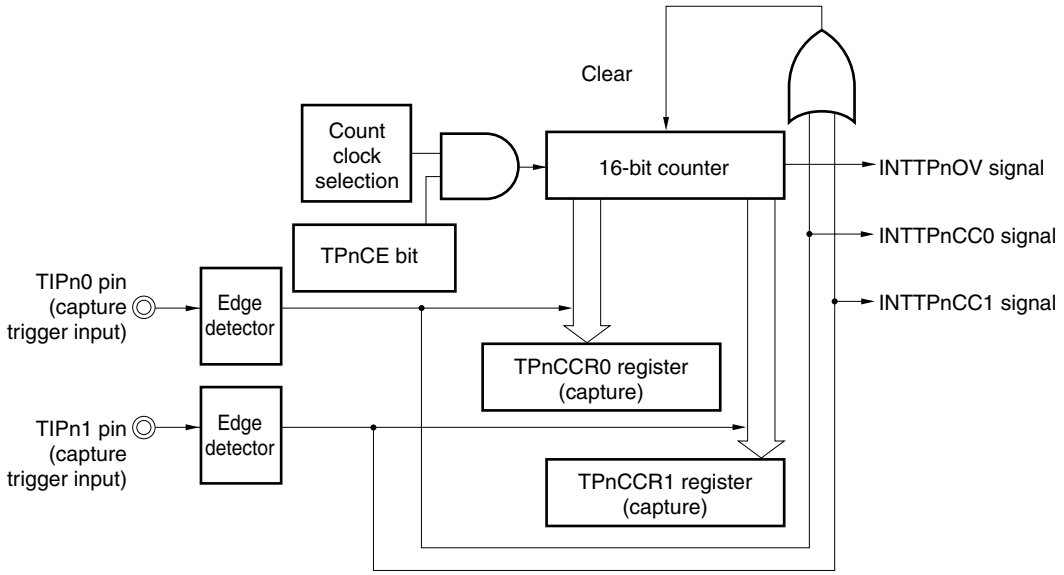
7.6.7 Pulse width measurement mode (TPnMD2 to TPnMD0 bits = 110)

In the pulse width measurement mode, 16-bit timer/event counter P starts counting when the TPnCTL0.TPnCE bit is set to 1. Each time the valid edge input to the TIPnm pin has been detected, the count value of the 16-bit counter is stored in the TPnCCRm register, and the 16-bit counter is cleared to 0000H.

The interval of the valid edge can be measured by reading the TPnCCRm register after a capture interrupt request signal (INTTPnCCm) occurs.

For example, in case of Figure 7-39, select either the TIPn0 or TIPn1 pin as the capture trigger input pin, and specify “No edge detected” by using the TPnIOC1 register for the unused pins.

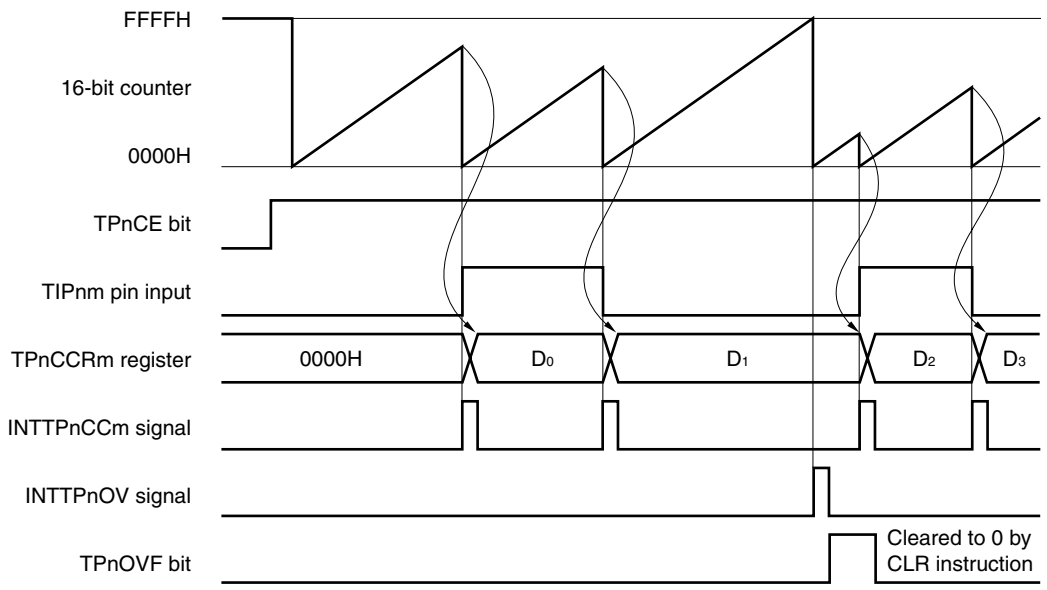
<R> **Figure 7-38. Configuration in Pulse Width Measurement Mode**



Caution When in pulse width measurement mode, select the internal clock (set TPnCTL1.TPnEEE bit = 0) as the count clock.

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1
μPD703253Y: n = 0 to 5, m = 0 or 1

Figure 7-39. Basic Timing in Pulse Width Measurement Mode



Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

When the TPnCE bit is set to 1, the 16-bit counter starts counting. When the valid edge input to the TIPnm pin is later detected, the count value of the 16-bit counter is stored in the TPnCCRm register, the 16-bit counter is cleared to 0000H, and a capture interrupt request signal (INTTPnCCm) is generated.

The pulse width is calculated as follows.

Pulse width = Captured value × Count clock cycle

If the valid edge is not input to the TIPnm pin even when the 16-bit counter counted up to FFFFH, an overflow interrupt request signal (INTTPnOV) is generated at the next count clock, and the counter is cleared to 0000H and continues counting. At this time, the overflow flag (TPnOPT0.TPnOVF bit) is also set to 1. Clear the overflow flag to 0 by executing the CLR instruction via software.

If the overflow flag is set to 1, the pulse width can be calculated as follows.

Pulse width = (10000H × TPnOVF bit set (1) count + Captured value) × Count clock cycle

Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

<R> μPD703253Y: n = 0 to 5, m = 0 or 1

Figure 7-40. Register Setting in Pulse Width Measurement Mode (1/2)

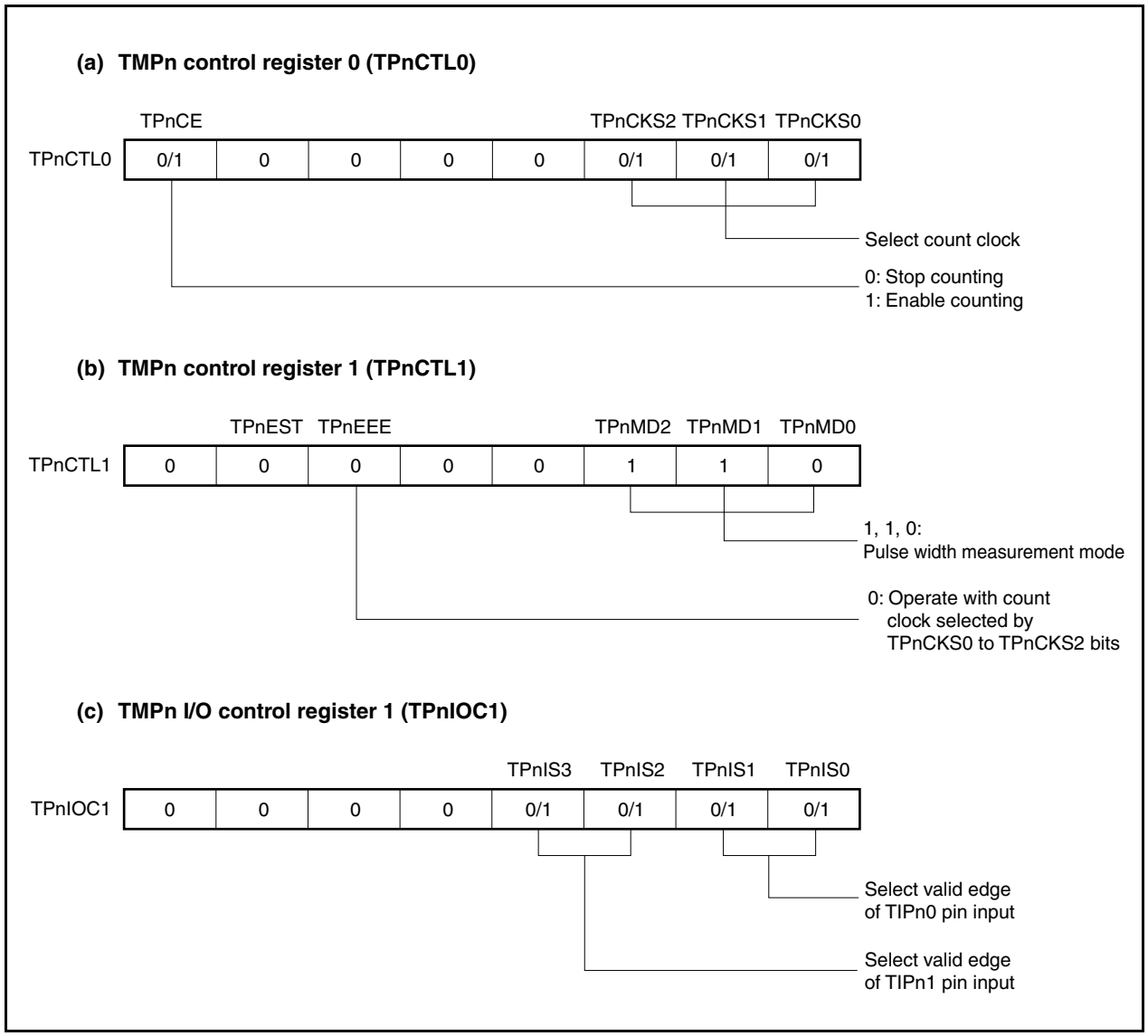


Figure 7-40. Register Setting in Pulse Width Measurement Mode (2/2)

(d) TMPn option register 0 (TPnOPT0)

			TPnCCS1	TPnCCS0				TPnOVF
TPnOPT0	0	0	0	0	0	0	0	0/1

Overflow flag

(e) TMPn counter read buffer register (TPnCNT)

The value of the 16-bit counter can be read by reading the TPnCNT register.

(f) TMPn capture/compare registers 0 and 1 (TPnCCR0 and TPnCCR1)

These registers store the count value of the 16-bit counter when the valid edge input to the TIPnm pin is detected.

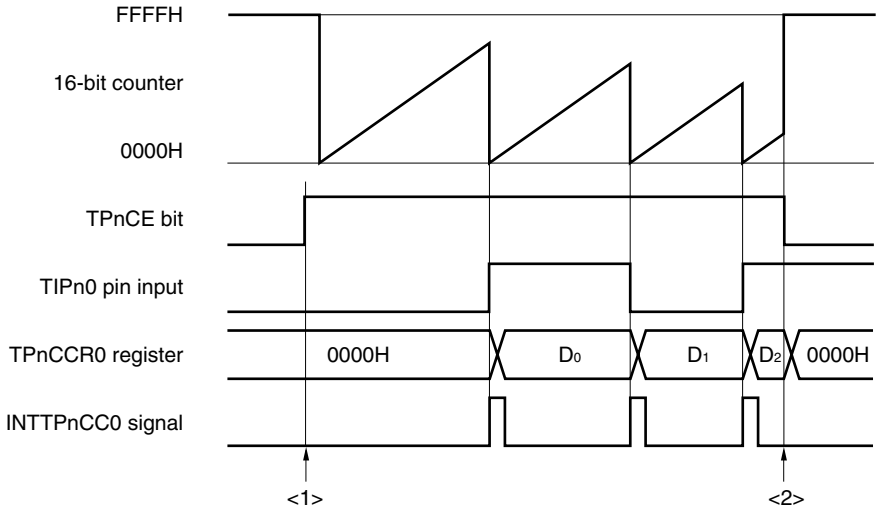
<R> **Remarks 1.** TMPn I/O control register 0 (TPnIOC0) and TMPn I/O control register 2 (TPnIOC2) are not used in the pulse width measurement mode.

2. μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1

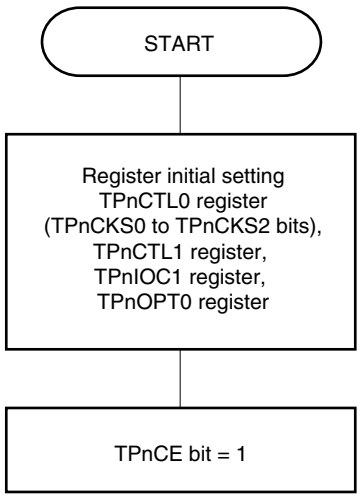
<R> μPD703253Y: n = 0 to 5, m = 0 or 1

(1) Operation flow in pulse width measurement mode

Figure 7-41. Software Processing Flow in Pulse Width Measurement Mode



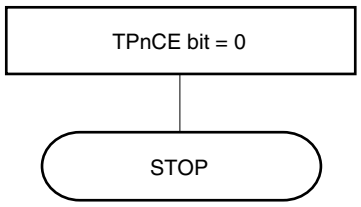
<R> <1> Count operation start flow



Initial setting of these registers is performed before the TPnCE bit is set to 1.

The TPnCKS0 to TPnCKS2 bits can be set when counting starts (TPnCE bit = 1).

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TPnCE bit to 0.

Remark μPD703249Y and 703252Y: n = 0 to 4

<R> μPD703253Y: n = 0 to 5

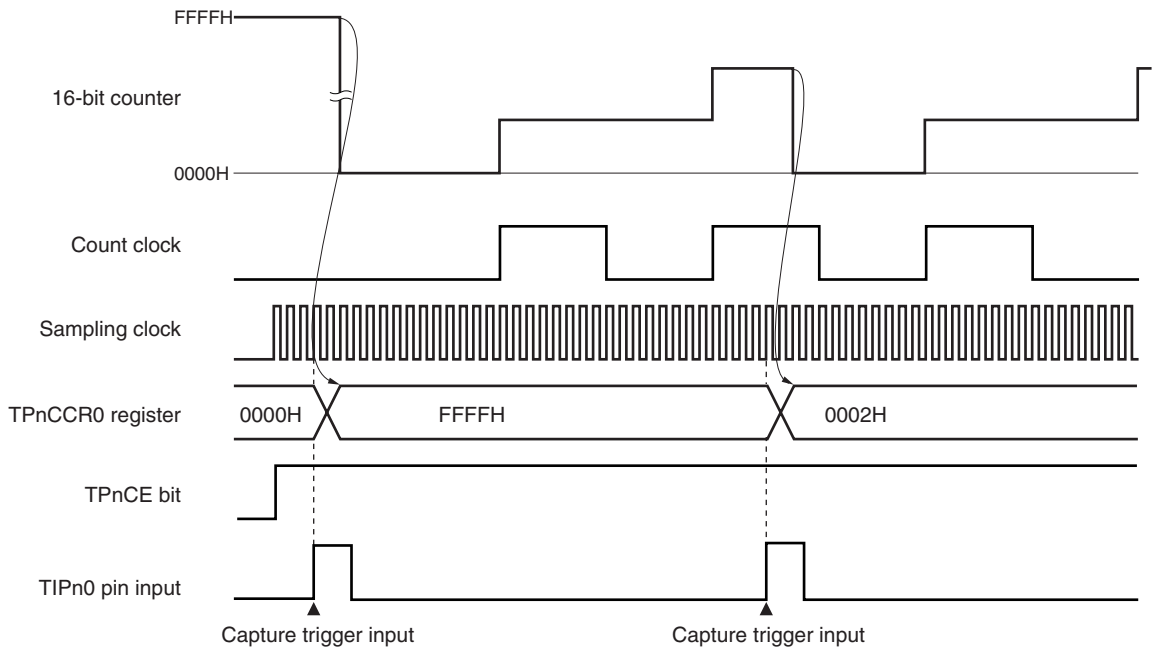
(2) Operation timing in pulse width measurement mode

(a) Clearing overflow flag

The overflow flag can be cleared to 0 by clearing the TPnOVF bit to 0 with the CLR instruction after reading the TPnOVF bit when it is 1 and by writing 8-bit data (bit 0 is 0) to the TPnOPT0 register after reading the TPnOVF bit when it is 1.

<R> **(3) Notes**

If a slow clock is selected as the count clock, FFFFH, not 0000H, may be captured to the TPnCCRm register if the capture trigger is input immediately after the TPnCTL0.TPnCE bit has been set to 1.



Remark μPD703249Y and 703252Y: n = 0 to 4, m = 0 or 1
μPD703253Y: n = 0 to 5, m = 0 or 1

CHAPTER 8 16-BIT INTERVAL TIMER M (TMM)

Timer M (TMM) is a 16-bit interval timer.
The V850ES/SG1 incorporates TMM0.

8.1 Overview

An outline of TMM0 is shown below.

- Interval function
- 8 clocks selectable
- 16-bit counter × 1
  (The 16-bit counter cannot be read during timer count operation.)
- Compare register × 1
  (The compare register cannot be written during timer counter operation.)
- Compare match interrupt × 1

Timer M supports only the clear & start mode. The free-running timer mode is not supported.

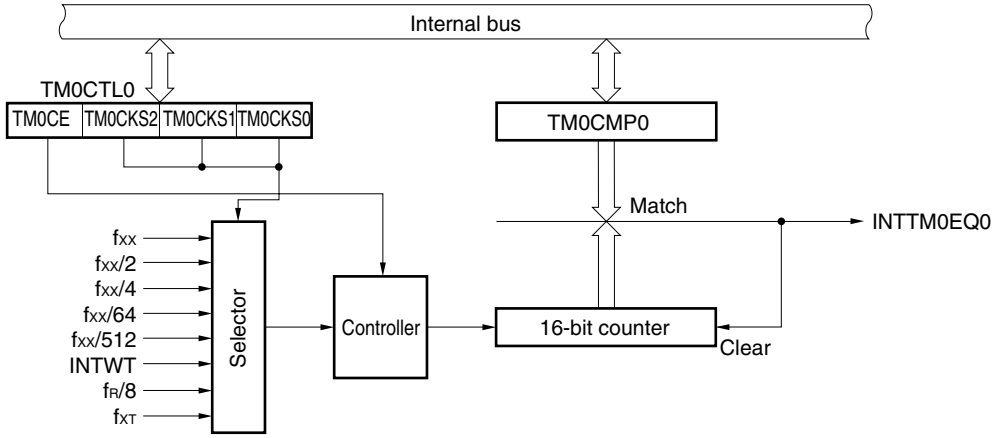
8.2 Configuration

TMM0 includes the following hardware.

Table 8-1. Configuration of TMM0

Item	Configuration
Timer register	16-bit counter
Register	TMM0 compare register 0 (TM0CMP0)
Control register	TMM0 control register 0 (TM0CTL0)

Figure 8-1. Block Diagram of TMM0



Remark f_{XX}: Main clock frequency
f_R: Internal oscillation clock frequency
f_{XT}: Subclock frequency
INTWT: Watch timer interrupt request signal

(1) 16-bit counter

This is a 16-bit counter that counts the internal clock.

The 16-bit counter cannot be read or written.

(2) TMM0 compare register 0 (TM0CMP0)

The TM0CMP0 register is a 16-bit compare register.

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

Reset sets this register to 0000H.

The same value can always be written to the TM0CMP0 register by software.

TM0CMP0 register rewrite is prohibited during the TMM0 operation (TM0CTL0.TM0CE bit = 1).

After reset: 0000H R/W Address: FFFFF694H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TM0CMP0																

Caution Do not set the TM0CMP0 register to FFFFH.

8.3 Register

(1) TMM0 control register (TM0CTL0)

The TM0CTL0 register is an 8-bit register that controls the TMM0 operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

The same value can always be written to the TM0CTL0 register by software.

After reset: 00H R/W Address: FFFFF690H

	<7>	6	5	4	3	2	1	0
TM0CTL0	TM0CE	0	0	0	0	TM0CKS2	TM0CKS1	TM0CKS0

TM0CE	Internal clock operation enable/disable specification
0	TMM0 operation disabled (16-bit counter reset asynchronously). Operation clock application stopped.
1	TMM0 operation enabled. Operation clock application started. TMM0 operation started.
The internal clock control and internal circuit reset for TMM0 are performed asynchronously with the TM0CE bit. When the TM0CE bit is cleared to 0, the internal clock of TMM0 is disabled (fixed to low level) and 16-bit counter is reset asynchronously.	

TM0CKS2	TM0CKS1	TM0CKS0	Count clock selection
0	0	0	f_{XX}
0	0	1	$f_{XX}/2$
0	1	0	$f_{XX}/4$
0	1	1	$f_{XX}/64$
1	0	0	$f_{XX}/512$
1	0	1	INTWT
1	1	0	$f_R/8$
1	1	1	f_{XT}

Cautions 1. Set the TM0CKS2 to TM0CKS0 bits when TM0CE bit = 0.
When changing the value of TM0CE from 0 to 1, it is not possible to set the value of the TM0CKS2 to TM0CKS0 bits simultaneously.
2. Be sure to clear bits 3 to 6 to "0".

Remark f_{XX}: Main clock frequency
f_R: Internal oscillation clock frequency
f_{XT}: Subclock frequency

8.4 Operation

Caution Do not set the TM0CMP0 register to FFFFH.

8.4.1 Interval timer mode

In the interval timer mode, an interrupt request signal (INTTM0EQ0) is generated at the interval set by the TM0CMP0 register if the TM0CTL0.TM0CE bit is set to 1.

Figure 8-2. Configuration of Interval Timer

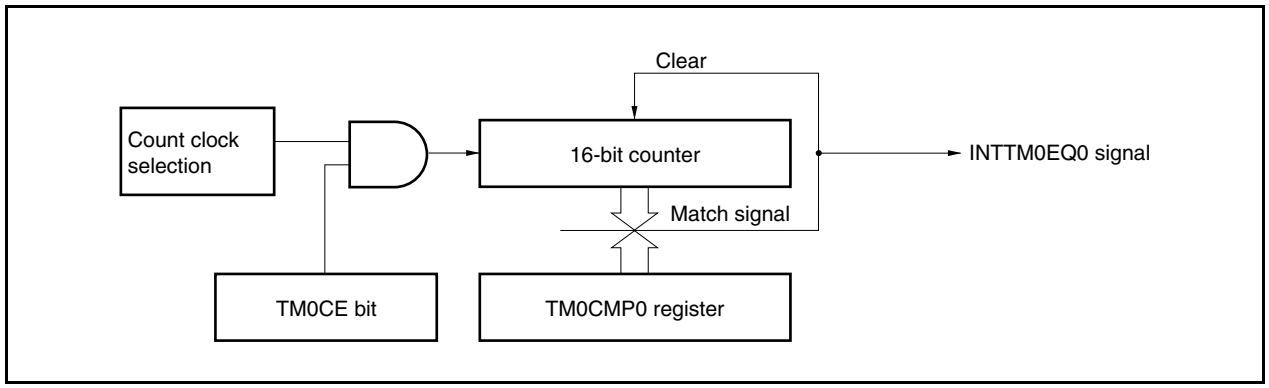
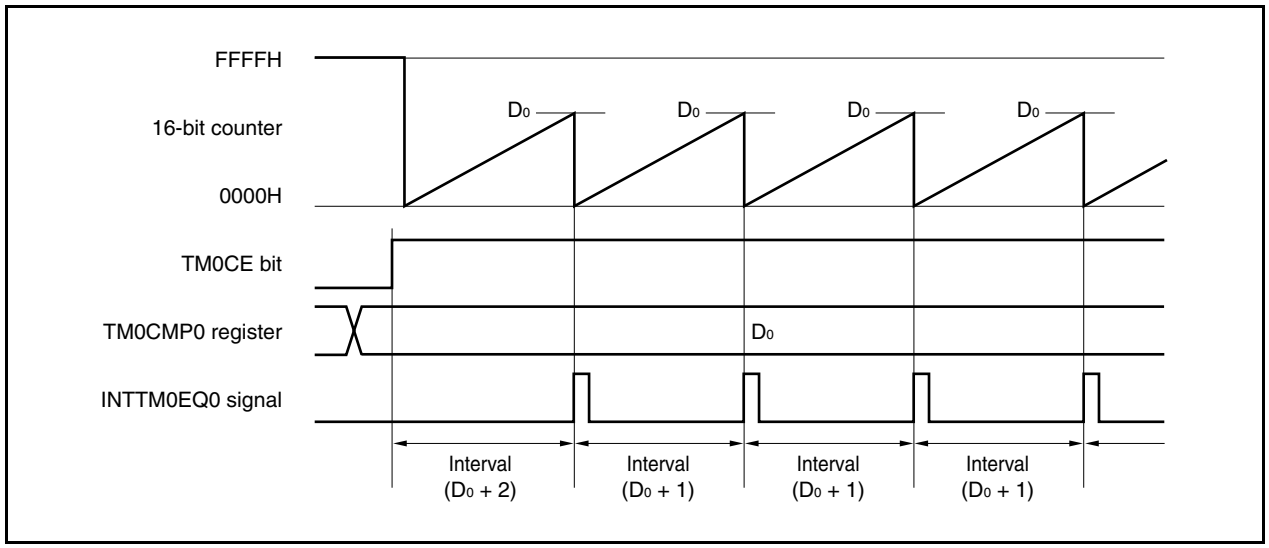


Figure 8-3. Basic Timing of Operation in Interval Timer Mode

<R>



When the TM0CE bit is set to 1, the value of the 16-bit counter is cleared from FFFFH to 0000H in synchronization with the count clock, and the counter starts counting.

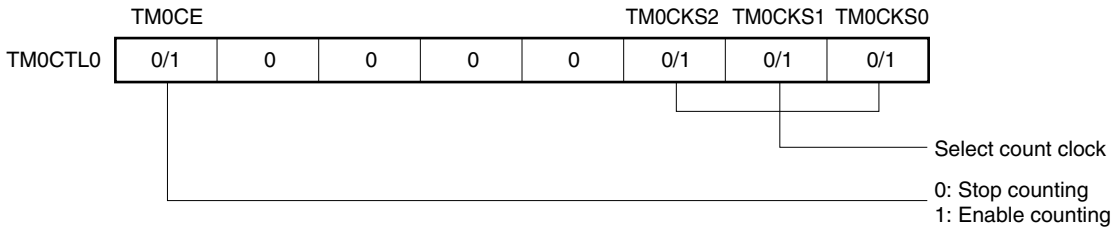
When the count value of the 16-bit counter matches the value of the TM0CMP0 register, the 16-bit counter is cleared to 0000H and a compare match interrupt request signal (INTTM0EQ0) is generated.

The interval can be calculated by the following expression.

Interval = (Set value of TM0CMP0 register + 1) × Count clock cycle

Figure 8-4. Register Setting for Interval Timer Mode Operation

(a) TMM0 control register 0 (TM0CTL0)



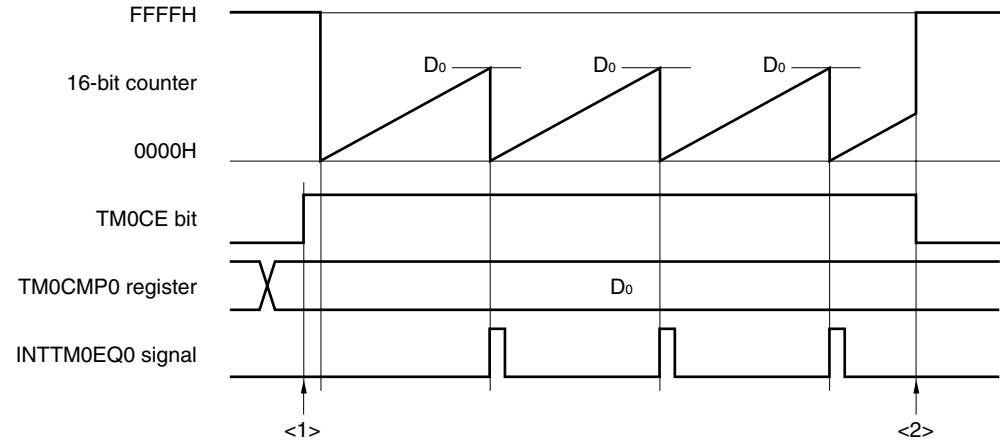
(b) TMM0 compare register 0 (TM0CMP0)

If the TM0CMP0 register is set to D_0, the interval is as follows.

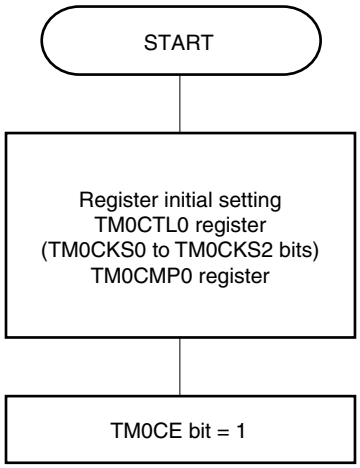
Interval = $(D_0 + 1) \times$ Count clock cycle

(1) Interval timer mode operation flow

Figure 8-5. Software Processing Flow in Interval Timer Mode



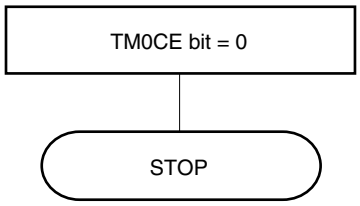
<1> Count operation start flow



Initial setting of these registers is performed before the TM0CE bit is set to 1.

The TM0CKS0 to TM0CKS2 bits cannot be set when counting starts (TM0CE bit = 1).

<2> Count operation stop flow



The counter is initialized and counting is stopped by clearing the TM0CE bit to 0.

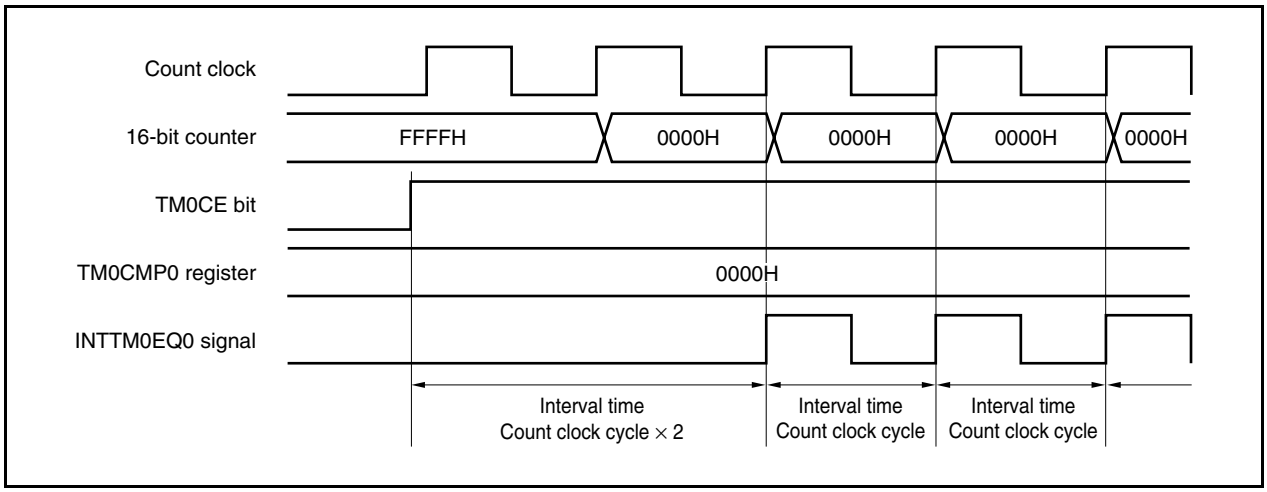
(2) Interval timer mode operation timing

Caution Do not set the TM0CMP0 register to FFFFH.

<R>

(a) Operation if TM0CMP0 register is set to 0000H

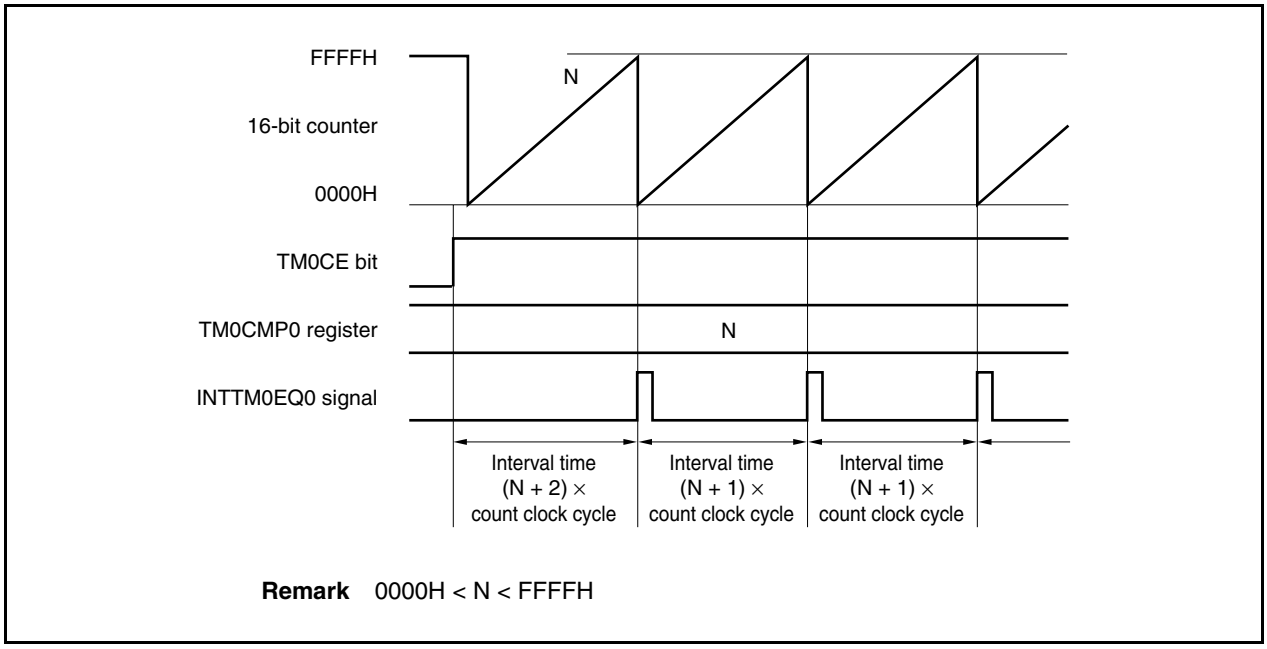
If the TM0CMP0 register is set to 0000H, the INTTM0EQ0 signal is generated at each count clock. The value of the 16-bit counter is always 0000H.



(b) Operation if TM0CMP0 register is set to N

If the TM0CMP0 register is set to N, the 16-bit counter counts up to N. The counter is cleared to 0000H in synchronization with the next count-up timing and the INTTM0EQ0 signal is generated.

<R>



Remark 0000H < N < FFFFH

8.4.2 Cautions

(1) It takes the 16-bit counter up to the following time to start counting after the TM0CTL0.TM0CE bit is set to 1, depending on the count clock selected.

Selected Count Clock	Maximum Time Before Counting Start
f_{XX}	$2/f_{XX}$
$f_{XX}/2$	$6/f_{XX}$
$f_{XX}/4$	$24/f_{XX}$
$f_{XX}/64$	$128/f_{XX}$
$f_{XX}/512$	$1024/f_{XX}$
INTWT	Second rising edge of INTWT signal
$f_R/8$	$16/f_R$
f_{XT}	$2/f_{XT}$

(2) Rewriting the TM0CMP0 and TM0CTL0 registers is prohibited while TMM0 is operating.
If these registers are rewritten while the TM0CE bit is 1, the operation cannot be guaranteed.
If they are rewritten by mistake, clear the TM0CTL0.TM0CE bit to 0, and re-set the registers.

<R> (3) Do not set the TM0CMP0 register to FFFFH.

CHAPTER 9 WATCH TIMER FUNCTIONS

9.1 Functions

The watch timer has the following functions.

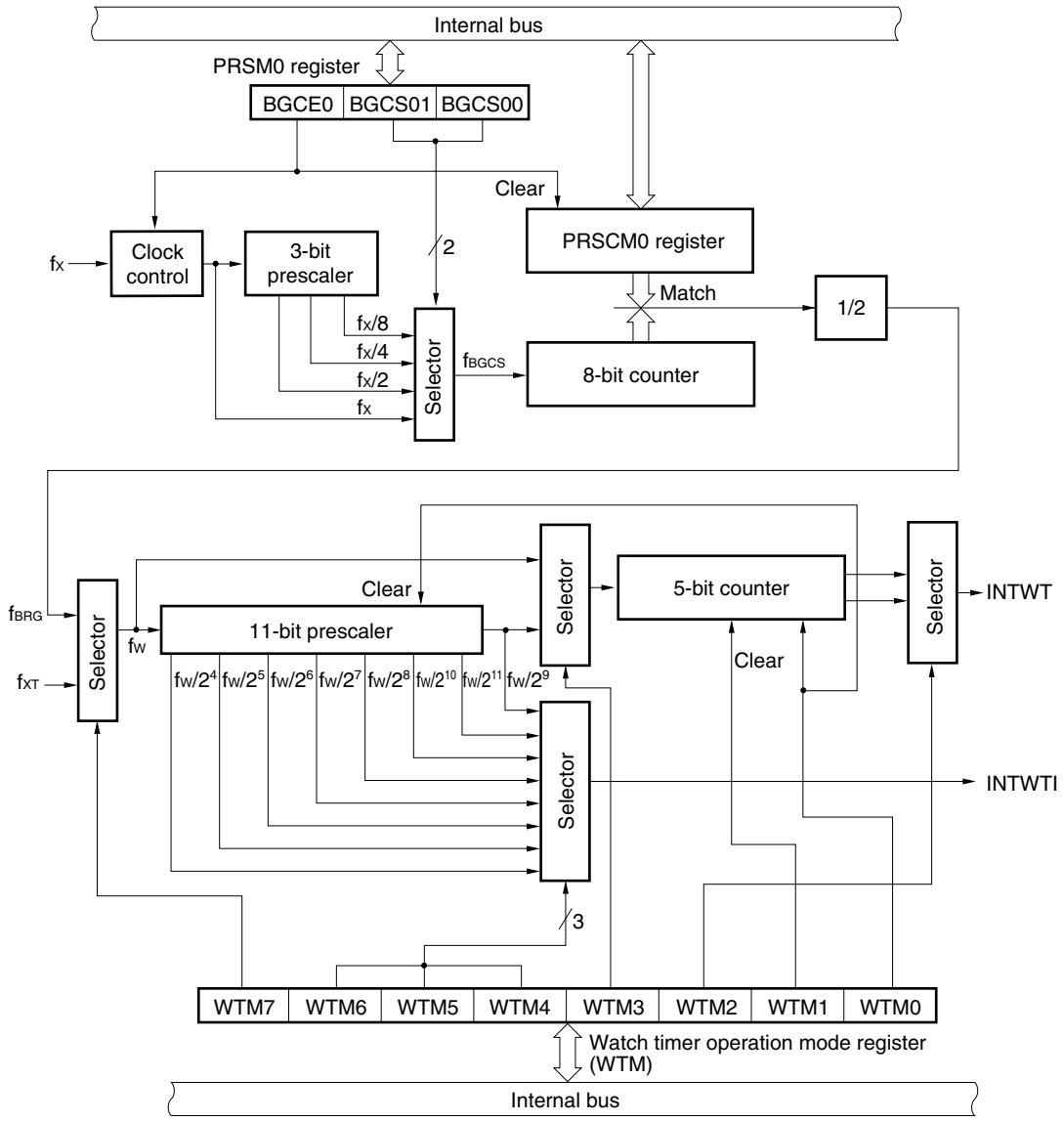
- Watch timer: An interrupt request signal (INTWT) is generated at intervals of 0.5 or 0.25 seconds by using the main clock or subclock.
- Interval timer: An interrupt request signal (INTWTI) is generated at set intervals.

The watch timer and interval timer functions can be used at the same time.

9.2 Configuration

The block diagram of the watch timer is shown below.

Figure 9-1. Block Diagram of Watch Timer



Remark fX: Main clock oscillation frequency
fBGCS: Watch timer source clock frequency
fBRG: Watch timer count clock frequency
fXT: Subclock frequency
fW: Watch timer clock frequency
INTWT: Watch timer interrupt request signal
INTWTI: Interval timer interrupt request signal

(1) Clock control

This block controls supplying and stopping the operating clock (f_X) when the watch timer operates on the main clock.

(2) 3-bit prescaler

This prescaler divides f_X to generate $f_X/2$, $f_X/4$, or $f_X/8$.

(3) 8-bit counter

This 8-bit counter counts the source clock (f_{BGCS}).

(4) 11-bit prescaler

This prescaler divides f_W to generate a clock of $f_W/2^4$ to $f_W/2^{11}$.

(5) 5-bit counter

This counter counts f_W or $f_W/2^9$, and generates a watch timer interrupt request signal at intervals of $2^4/f_W$, $2^5/f_W$, $2^{12}/f_W$, or $2^{14}/f_W$.

(6) Selector

The watch timer has the following five selectors.

- Selector that selects one of f_X, $f_X/2$, $f_X/4$, or $f_X/8$ as the source clock of the watch timer
- Selector that selects the main clock (f_X) or subclock (f_{XT}) as the clock of the watch timer
- Selector that selects f_W or $f_W/2^9$ as the count clock frequency of the 5-bit counter
- Selector that selects $2^4/f_W$, $2^{13}/f_W$, $2^5/f_W$, or $2^{14}/f_W$ as the INTWT signal generation time interval
- Selector that selects $2^4/f_W$ to $2^{11}/f_W$ as the interval timer interrupt request signal (INTWTI) generation time interval

(7) PRSCM register

This is an 8-bit compare register that sets the interval time.

(8) PRSM register

This register controls clock supply to the watch timer.

(9) WTM register

This is an 8-bit register that controls the operation of the watch timer/interval timer, and sets the interrupt request signal generation interval.

9.3 Control Registers

The following registers are provided for the watch timer.

- Prescaler mode register 0 (PRSM0)
- Prescaler compare register 0 (PRSCM0)
- Watch timer operation mode register (WTM)

(1) Prescaler mode register 0 (PRSM0)

The PRSM0 register controls the generation of the watch timer count clock.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF8B0H

	7	6	5	<4>	3	2	1	0
PRSM0	0	0	0	BGCE0	0	0	BGCS01	BGCS00

BGCE0	Main clock operation enable
0	Disabled
1	Enabled

BGCS01	BGCS00	Selection of watch timer source clock (f_{BGCS})		
			5 MHz	4 MHz
0	0	f_X	200 ns	250 ns
0	1	$f_X/2$	400 ns	500 ns
1	0	$f_X/4$	800 ns	1 μs
1	1	$f_X/8$	1.6 μs	2 μs

Cautions 1. Do not change the values of the BGCS00 and BGCS01 bits during watch timer operation.

2. Set the PRSM0 register before setting the BGCE0 bit to 1.

3. Set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain an f_{BRG} frequency of 32.768 kHz.

(2) Prescaler compare register 0 (PRSCM0)

The PRSCM0 register is an 8-bit compare register.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF8B1H

	7	6	5	4	3	2	1	0
PRSCM0	PRSCM07	PRSCM06	PRSCM05	PRSCM04	PRSCM03	PRSCM02	PRSCM01	PRSCM00

Cautions 1. Do not rewrite the PRSCM0 register during watch timer operation.

2. Set the PRSCM0 register before setting the PRSM0.BGCE0 bit to 1.

3. Set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain an f_{BRG} frequency of 32.768 kHz.

The calculation for f_{BRG} is shown below.

$$f_{BRG} = f_{BGCS}/2N$$

Remark f_{BGCS}: Watch timer source clock set by the PRSM0 register

N: Set value of the PRSCM0 register = 1 to 256

However, N = 256 when the PRSCM0 register is set to 00H.

(3) Watch timer operation mode register (WTM)

The WTM register enables or disables the count clock and operation of the watch timer, sets the interval time of the prescaler, controls the operation of the 5-bit counter, and sets the set time of the watch flag.

Set the PRSM0 register before setting the WTM register.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

(1/2)

After reset: 00H R/W Address: FFFFF680H

	7	6	5	4	3	2	<1>	<0>
WTM	WTM7	WTM6	WTM5	WTM4	WTM3	WTM2	WTM1	WTM0

WTM7	WTM6	WTM5	WTM4	Selection of interval time of prescaler
0	0	0	0	$2^4/f_W$ (488 μs: $f_W = f_{XT}$)
0	0	0	1	$2^5/f_W$ (977 μs: $f_W = f_{XT}$)
0	0	1	0	$2^6/f_W$ (1.95 ms: $f_W = f_{XT}$)
0	0	1	1	$2^7/f_W$ (3.91 ms: $f_W = f_{XT}$)
0	1	0	0	$2^8/f_W$ (7.81 ms: $f_W = f_{XT}$)
0	1	0	1	$2^9/f_W$ (15.6 ms: $f_W = f_{XT}$)
0	1	1	0	$2^{10}/f_W$ (31.3 ms: $f_W = f_{XT}$)
0	1	1	1	$2^{11}/f_W$ (62.5 ms: $f_W = f_{XT}$)
1	0	0	0	$2^4/f_W$ (488 μs: $f_W = f_{BRG}$)
1	0	0	1	$2^5/f_W$ (977 μs: $f_W = f_{BRG}$)
1	0	1	0	$2^6/f_W$ (1.95 ms: $f_W = f_{BRG}$)
1	0	1	1	$2^7/f_W$ (3.90 ms: $f_W = f_{BRG}$)
1	1	0	0	$2^8/f_W$ (7.81 ms: $f_W = f_{BRG}$)
1	1	0	1	$2^9/f_W$ (15.6 ms: $f_W = f_{BRG}$)
1	1	1	0	$2^{10}/f_W$ (31.2 ms: $f_W = f_{BRG}$)
1	1	1	1	$2^{11}/f_W$ (62.5 ms: $f_W = f_{BRG}$)

(2/2)

WTM7	WTM3	WTM2	Selection of set time of watch flag
0	0	0	$2^{14}/f_W$ (0.5 s: $f_W = f_{XT}$)
0	0	1	$2^{13}/f_W$ (0.25 s: $f_W = f_{XT}$)
0	1	0	$2^5/f_W$ (977 μs: $f_W = f_{XT}$)
0	1	1	$2^4/f_W$ (488 μs: $f_W = f_{XT}$)
1	0	0	$2^{14}/f_W$ (0.5 s: $f_W = f_{BRG}$)
1	0	1	$2^{13}/f_W$ (0.25 s: $f_W = f_{BRG}$)
1	1	0	$2^5/f_W$ (977 μs: $f_W = f_{BRG}$)
1	1	1	$2^4/f_W$ (488 μs: $f_W = f_{BRG}$)

WTM1	Control of 5-bit counter operation
0	Clears after operation stops
1	Starts

WTM0	Watch timer operation enable
0	Stops operation (clears both prescaler and 5-bit counter)
1	Enables operation

Caution Rewrite the WTM2 to WTM7 bits while both the WTM0 and WTM1 bits are 0.

Remarks 1. f_W: Watch timer clock frequency

2. Values in parentheses apply to operation with f_W = 32.768 kHz

9.4 Operation

9.4.1 Operation as watch timer

The watch timer generates an interrupt request signal (INTWT) at fixed time intervals. The watch timer operates using time intervals of 0.25 or 0.5 seconds with the subclock (32.768 kHz) or main clock.

The count operation starts when the WTM.WTM1 and WTM.WTM0 bits are set to 11. When the WTM0 bit is cleared to 0, the 11-bit prescaler and 5-bit counter are cleared and the count operation stops.

The time of the watch timer can be adjusted by clearing the WTM1 bit to 0 and then the 5-bit counter when operating at the same time as the interval timer. At this time, an error of up to 15.6 ms may occur for the watch timer, but the interval timer is not affected.

If the main clock is used as the count clock of the watch timer, set the count clock using the PRSM0.BGCS01 and BGCS00 bits, the 8-bit comparison value using the PRSCM0 register, and the count clock frequency (f_{BRG}) of the watch timer to 32.768 kHz.

When the PRSM0.BGCE0 bit is set (1), f_{BRG} is supplied to the watch timer.

f_{BRG} can be calculated by the following expression.

$$f_{BRG} = f_X/(2^{m+1} \times N)$$

To set f_{BRG} to 32.768 kHz, perform the following calculation and set the BGCS01 and BGCS00 bits and the PRSCM0 register.

<1> Set N = f_X/65,536. Set m = 0.
<2> When the value resulting from rounding up the first decimal place of N is even, set N before the roundup as N/2 and m as m + 1.
<3> Repeat <2> until N is odd or m = 3.
<4> Set the value resulting from rounding up the first decimal place of N to the PRSCM0 register and m to the BGCS01 and BGCS00 bits.

Example: When f_X = 4.00 MHz
<1> N = 4,000,000/65,536 = 61.03…, m = 0
<2>, <3> Because N (round up the first decimal place) is odd, N = 61, m = 0.
<4> Set value of PRSCM0 register: 3DH (61), set value of BGCS01 and BGCS00 bits: 00

At this time, the actual f_{BRG} frequency is as follows.

$f_{BRG} = f_X/(2^{m+1} \times N) = 4{,}000{,}000/(2 \times 61)$
$= 32.787$ kHz

Remark m: Division value (set value of BGCS01 and BGCS00 bits) = 0 to 3
N: Set value of PRSCM0 register = 1 to 256
However, N = 256 when PRSCM0 register is set to 00H.
f_X: Main clock oscillation frequency

9.4.2 Operation as interval timer

The watch timer can also be used as an interval timer that repeatedly generates an interrupt request signal (INTWTI) at intervals specified by a preset count value.

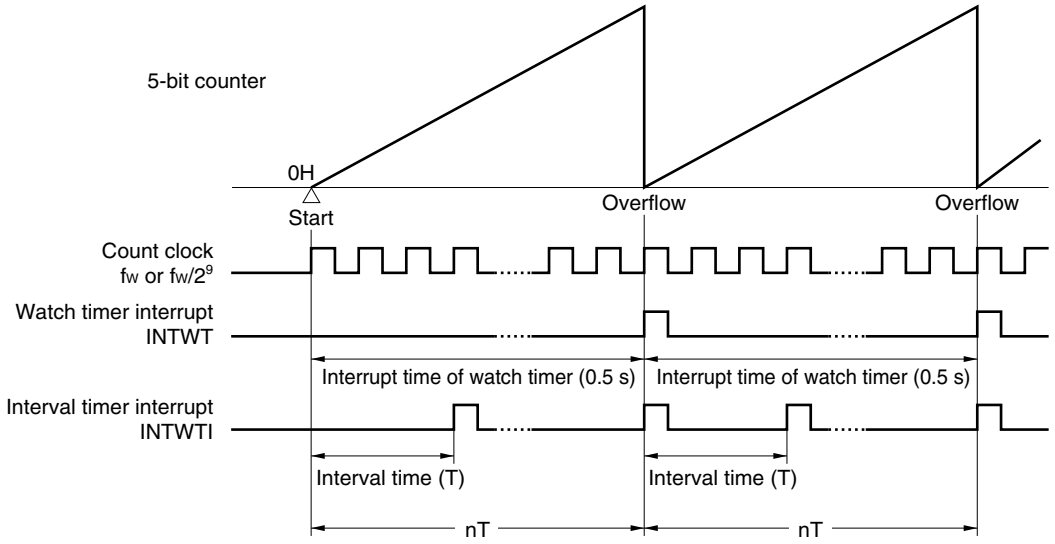
The interval time can be selected by the WTM.WTM4 to WTM.WTM7 bits.

Table 9-1. Interval Time of Interval Timer

WTM7	WTM6	WTM5	WTM4	Interval Time	
0	0	0	0	$2^4 \times 1/fw$	488 μs (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	0	1	$2^5 \times 1/fw$	977 μs (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	1	0	$2^6 \times 1/fw$	1.95 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	0	1	1	$2^7 \times 1/fw$	3.91 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	0	0	$2^8 \times 1/fw$	7.81 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	0	1	$2^9 \times 1/fw$	15.6 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	1	0	$2^{10} \times 1/fw$	31.3 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
0	1	1	1	$2^{11} \times 1/fw$	62.5 ms (operating at $f_W = f_{XT}$ = 32.768 kHz)
1	0	0	0	$2^4 \times 1/fw$	488 μs (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	0	1	$2^5 \times 1/fw$	977 μs (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	1	0	$2^6 \times 1/fw$	1.95 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	0	1	1	$2^7 \times 1/fw$	3.91 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	0	0	$2^8 \times 1/fw$	7.81 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	0	1	$2^9 \times 1/fw$	15.6 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	1	0	$2^{10} \times 1/fw$	31.3 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)
1	1	1	1	$2^{11} \times 1/fw$	62.5 ms (operating at $f_W = f_{BRG}$ = 32.768 kHz)

Remark f_W: Watch timer clock frequency

Figure 9-2. Operation Timing of Watch Timer/Interval Timer



Remarks 1. When 0.5 seconds of the watch timer interrupt time is set.

2. fw: Watch timer clock frequency

Values in parentheses apply to operation with fw = 32.768 kHz.

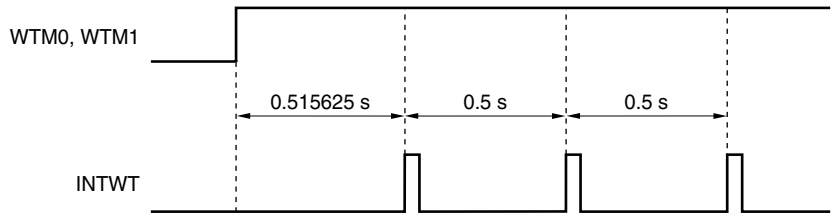
n: Number of interval timer operations

9.4.3 Cautions

Some time is required before the first watch timer interrupt request signal (INTWT) is generated after operation is enabled (WTM.WTM1 and WTM.WTM0 bits = 1).

Figure 9-3. Example of Generation of Watch Timer Interrupt Request Signal (INTWT) (When Interrupt Cycle = 0.5 s)

It takes 0.515625 seconds (max.) for the first INTWT signal to be generated ($2^9 \times 1/32768 = 0.015625$ seconds longer (max.)). The INTWT signal is then generated every 0.5 seconds.



CHAPTER 10 FUNCTIONS OF WATCHDOG TIMER 2

10.1 Functions

Watchdog timer 2 has the following functions.

- Default-start watchdog timer[Note 1]
  → Reset mode: Reset operation upon overflow of watchdog timer 2 (generation of WDT2RES signal)
  → Non-maskable interrupt request mode: NMI operation upon overflow of watchdog timer 2 (generation of INTWDT2 signal)[Note 2]
- Input selectable from main clock, internal oscillation clock, and subclock as the source clock

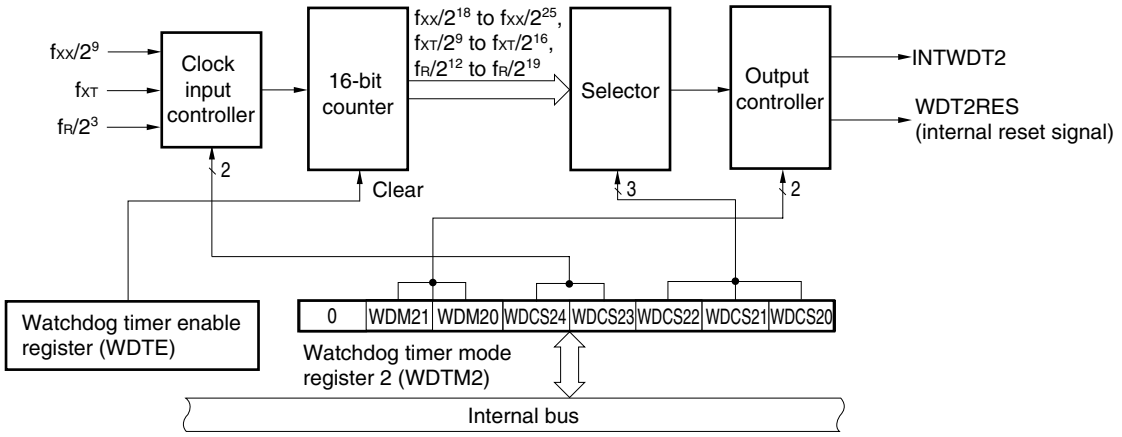
Notes 1. Watchdog timer 2 automatically starts in the reset mode following reset release.
When watchdog timer 2 is not used, either stop its operation before reset is executed via this function, or clear watchdog timer 2 once and stop it within the next interval time.
Also, write to the WDTM2 register for verification purposes only once, even if the default settings (reset mode, interval time: $f_R/2^{19}$) do not need to be changed.

2. For the non-maskable interrupt servicing due to a non-maskable interrupt request signal (INTWDT2), see **19.2.2 (2) INTWDT2 signal**.

10.2 Configuration

The following shows the block diagram of watchdog timer 2.

Figure 10-1. Block Diagram of Watchdog Timer 2



Remark fXX: Main clock frequency
fXT: Subclock frequency
fR: Internal oscillation clock frequency
INTWDT2: Non-maskable interrupt request signal from watchdog timer 2
WDTRES2: Watchdog timer 2 reset signal

Watchdog timer 2 includes the following hardware.

Table 10-1. Configuration of Watchdog Timer 2

Item	Configuration
Control registers	Watchdog timer mode register 2 (WDTM2) Watchdog timer enable register (WDTE)

10.3 Registers

(1) Watchdog timer mode register 2 (WDTM2)

The WDTM2 register sets the overflow time and operation clock of watchdog timer 2.

This register can be read or written in 8-bit units. This register can be read any number of times, but it can be written only once following reset release.

Reset input sets this register to 67H.

Caution Accessing the WDTM2 register is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset: 67H R/W Address: FFFFF6D0H

	7	6	5	4	3	2	1	0
WDTM2	0	WDM21	WDM20	WDCS24	WDCS23	WDCS22	WDCS21	WDCS20

WDM21	WDM20	Selection of operation mode of watchdog timer 2
0	0	Stops operation
0	1	Non-maskable interrupt request mode (generation of INTWDT2 signal)
1	–	Reset mode (generation of WDT2RES signal)

Cautions 1. For details on the WDCS20 to WDCS24 bits, see Table 10-2 Watchdog Timer 2 Clock Selection.

2. Although watchdog timer 2 can be stopped just by stopping operation of the internal oscillator, clear the WDTM2 register to 00H to securely stop the timer (to avoid selection of the main clock or subclock due to an erroneous write operation).

3. If the WDTM2 register is rewritten twice after reset, an overflow signal is forcibly generated and the counter is reset.

<R> **4. To intentionally generate an overflow signal, write to the WDTM2 register only twice or write a value other than ACH to the WDTE register once.**

However, when watchdog timer 2 is set to stop operation, an overflow signal is not generated even if data is written to the WDTM2 register only twice, or a value other than "ACH" is written to the WDTE register only once.

5. To stop the operation of watchdog timer 2, set the RCM.RSTOP bit to 1 (to stop the internal oscillator) and write 00H in the WDTM2 register. If the RCM.RSTOP bit cannot be set to 1, set the WDCS23 bit to 1 (2^n/fxx is selected and the clock can be stopped in the IDLE1, IDLW2, sub-IDLE, and subclock operation modes).

Table 10-2. Watchdog Timer 2 Clock Selection

WDCS24	WDCS23	WDCS22	WDCS21	WDCS20	Selected Clock	100 kHz (MIN.)	200 kHz (TYP.)	400 kHz (MAX.)
0	0	0	0	0	$2^{12}/f_R$	41.0 ms	20.5 ms	10.2 ms
0	0	0	0	1	$2^{13}/f_R$	81.9 ms	41.0 ms	20.5 ms
0	0	0	1	0	$2^{14}/f_R$	163.8 ms	81.9 ms	41.0 ms
0	0	0	1	1	$2^{15}/f_R$	327.7 ms	163.8 ms	81.9 ms
0	0	1	0	0	$2^{16}/f_R$	655.4 ms	327.7 ms	163.8 ms
0	0	1	0	1	$2^{17}/f_R$	1,310.7 ms	655.4 ms	327.7 ms
0	0	1	1	0	$2^{18}/f_R$	2,621.4 ms	1,310.7 ms	655.4 ms
0	0	1	1	1	$2^{19}/f_R$	5,242.9 ms	2,621.47 ms	1,310.7 ms
						f_{XX} = 20 MHz	f_{XX} = 16 MHz	f_{XX} = 10 MHz
0	1	0	0	0	$2^{18}/f_{XX}$	13.1 ms	16.4 ms	26.2 ms
0	1	0	0	1	$2^{19}/f_{XX}$	26.2 ms	32.8 ms	52.4 ms
0	1	0	1	0	$2^{20}/f_{XX}$	52.4 ms	65.5 ms	104.9 ms
0	1	0	1	1	$2^{21}/f_{XX}$	104.9 ms	131.1 ms	209.7 ms
0	1	1	0	0	$2^{22}/f_{XX}$	209.7 ms	262.1 ms	419.4 ms
0	1	1	0	1	$2^{23}/f_{XX}$	419.4 ms	524.3 ms	838.9 ms
0	1	1	1	0	$2^{24}/f_{XX}$	838.9 ms	1,048.6 ms	1,677.7 ms
0	1	1	1	1	$2^{25}/f_{XX}$	1,677.7 ms	2,097.2 ms	3,355.4 ms
						f_{XT} = 32.768 kHz		
1	×	0	0	0	$2^9/f_{XT}$	15.625 ms		
1	×	0	0	1	$2^{10}/f_{XT}$	31.25 ms		
1	×	0	1	0	$2^{11}/f_{XT}$	62.5 ms		
1	×	0	1	1	$2^{12}/f_{XT}$	125 ms		
1	×	1	0	0	$2^{13}/f_{XT}$	250 ms		
1	×	1	0	1	$2^{14}/f_{XT}$	500 ms		
1	×	1	1	0	$2^{15}/f_{XT}$	1,000 ms		
1	×	1	1	1	$2^{16}/f_{XT}$	2,000 ms		

(2) Watchdog timer enable register (WDTE)

The counter of watchdog timer 2 is cleared and counting restarted by writing “ACH” to the WDTE register.

The WDTE register can be read or written in 8-bit units.

Reset sets this register to 9AH.

After reset: 9AH R/W Address: FFFFF6D1H

	7	6	5	4	3	2	1	0
WDTE								

Cautions 1. When a value other than “ACH” is written to the WDTE register, an overflow signal is forcibly output.

2. When a 1-bit memory manipulation instruction is executed for the WDTE register, an overflow signal is forcibly output.

<R> **3. To intentionally generate an overflow signal, write to the WDTM2 register only twice or write a value other than ACH to the WDTE register once.**

However, when the watchdog timer 2 is set to stop operation, an overflow signal is not generated even if data is written to the WDTM2 register only twice, or a value other than “ACH” is written to the WDTE register only once

4. The read value of the WDTE register is “9AH” (which differs from written value “ACH”).

10.4 Operation

Watchdog timer 2 automatically starts in the reset mode following reset release.

The WDTM2 register can be written to only once following reset using byte access. To use watchdog timer 2, write the operation mode and the interval time to the WDTM2 register using an 8-bit memory manipulation instruction. After this, the operation of watchdog timer 2 cannot be stopped.

The WDCS24 to WDCS20 bits of the WDTM2 register are used to select the watchdog timer 2 loop detection time interval.

Writing ACH to the WDTE register clears the counter of watchdog timer 2 and starts the count operation again. After the count operation has started, write ACH to WDTE within the loop detection time interval.

If the time interval expires without ACH being written to the WDTE register, a reset signal (WDT2RES) or a non-maskable interrupt request signal (INTWDT2) is generated, depending on the set values of the WDM21 and WDTM2.WDM20 bits.

When the WDTM2.WDM21 bit is set to 1 (reset mode), if a WDT overflow occurs during oscillation stabilization after a reset or standby is released, no internal reset will occur and the CPU clock will switch to the internal oscillation clock.

To not use watchdog timer 2, write 00H to the WDTM2 register.

For the non-maskable interrupt servicing while the non-maskable interrupt request mode is set, see **19.2.2 (2) INTWDT2 signal**.

CHAPTER 11 A/D CONVERTER

11.1 Overview

The A/D converter converts analog input signals into digital values, has a resolution of 10 bits, and can handle 12 analog input signal channels (ANI0 to ANI11).

The A/D converter has the following features.

- 10-bit resolution
- 12 channels
- Successive approximation method
- Operating voltage: AV_{REF0} = 3.0 to 3.6 V
- Analog input voltage: 0 V to AV_{REF0}
- The following functions are provided as operation modes.
  - Continuous select mode
  - Continuous scan mode
  - One-shot select mode
  - One-shot scan mode
- The following functions are provided as trigger modes.
  - Software trigger mode
  - External trigger mode (external, 1)
  - Timer trigger mode
- Power-fail monitor function (conversion result compare function)

11.2 Functions

(1) 10-bit resolution A/D conversion

An analog input channel is selected from ANI0 to ANI11, and an A/D conversion operation is repeated at a resolution of 10 bits. Each time A/D conversion has been completed, an interrupt request signal (INTAD) is generated.

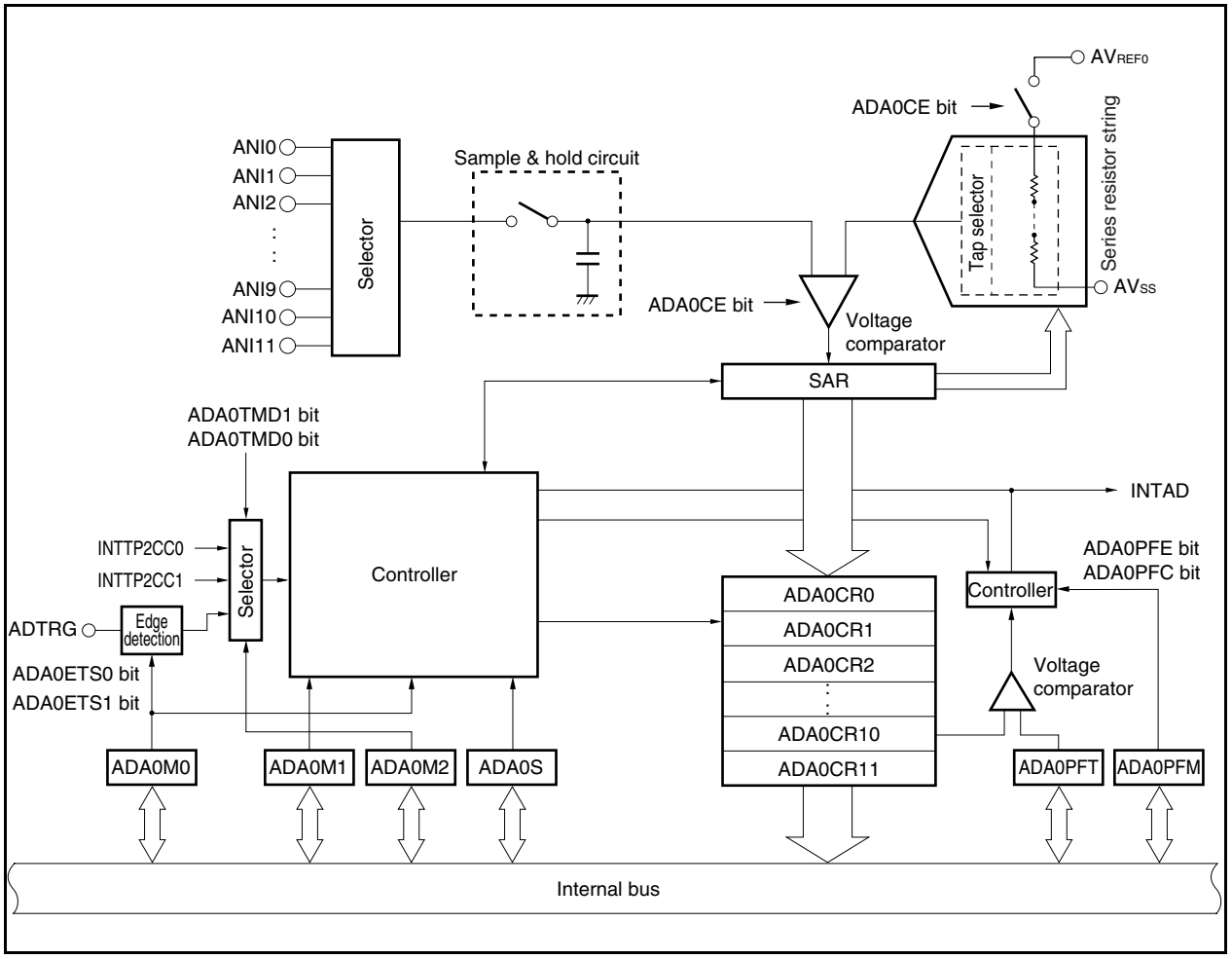
(2) Power-fail detection function

This function is used to detect a drop in the battery voltage. The result of A/D conversion (the value of the ADA0CRnH register) is compared with the value of the ADA0PFT register, and the INTAD signal is generated only when a specified comparison condition is satisfied (n = 0 to 11).

11.3 Configuration

The block diagram of the A/D converter is shown below.

Figure 11-1. Block Diagram of A/D Converter



The A/D converter includes the following hardware.

Table 11-1. Configuration of A/D Converter

Item	Configuration
Analog inputs	12 channels (ANI0 to ANI11 pins)
Registers	Successive approximation register (SAR) A/D conversion result registers 0 to 11 (ADA0CR0 to ADA0CR11) A/D conversion result registers 0H to 11H (ADCR0H to ADCR11H): Only higher 8 bits can be read
Control registers	A/D converter mode registers 0 to 2 (ADA0M0 to ADA0M2) A/D converter channel specification register 0 (ADA0S) Power fail compare mode register (ADA0PFM) Power fail compare threshold value register (ADA0PFT)

(1) Successive approximation register (SAR)

The SAR register compares the voltage value of the analog input signal with the voltage tap (compare voltage) value from the series resistor string, and holds the comparison result starting from the most significant bit (MSB).

When the comparison result has been held down to the least significant bit (LSB) (i.e., when A/D conversion is complete), the contents of the SAR register are transferred to the ADA0CRn register.

Remark n = 0 to 11

(2) A/D conversion result register n (ADA0CRn), A/D conversion result register nH (ADA0CRnH)

The ADA0CRn register is a 16-bit register that stores the A/D conversion result. ADA0ARn consist of 12 registers and the A/D conversion result is stored in the 10 higher bits of the AD0CRn register corresponding to analog input. (The lower 6 bits are fixed to 0.)

(3) A/D converter mode register 0 (ADA0M0)

This register specifies the operation mode and controls the conversion operation by the A/D converter.

(4) A/D converter mode register 1 (ADA0M1)

This register sets the conversion time of the analog input signal to be converted.

(5) A/D converter mode register 2 (ADA0M2)

This register sets the hardware trigger mode.

(6) A/D converter channel specification register (ADA0S)

This register sets the input port that inputs the analog voltage to be converted.

(7) Power-fail compare mode register (ADA0PFM)

This register sets the power-fail monitor mode.

(8) Power-fail compare threshold value register (ADA0PFT)

The ADA0PFT register sets a threshold value that is compared with the value of A/D conversion result register nH (ADA0CRnH). The 8-bit data set to the ADA0PFT register is compared with the higher 8 bits of the A/D conversion result register (ADA0CRnH).

(9) Controller

The controller compares the result of the A/D conversion (the value of the ADA0CRnH register) with the value of the ADA0PFT register when A/D conversion is completed or when the power-fail detection function is used, and generates the INTAD signal only when a specified comparison condition is satisfied.

(10) Sample & hold circuit

The sample & hold circuit samples each of the analog input signals selected by the input circuit and sends the sampled data to the voltage comparator. This circuit also holds the sampled analog input signal voltage during A/D conversion.

(11) Voltage comparator

The voltage comparator compares a voltage value that has been sampled and held with the voltage value of the series resistor string.

(12) Series resistor string

This series resistor string is connected between AV_{REF0} and AV_{SS} and generates a voltage for comparison with the analog input signal.

(13) ANI0 to ANI11 pins

These are analog input pins for the 12 A/D converter channels and are used to input analog signals to be converted into digital signals. Pins other than the one selected as the analog input by the ADA0S register can be used as input port pins.

Caution Make sure that the voltages input to the ANI0 to ANI11 pins do not exceed the rated values. In particular if a voltage of AV_{REF0} or higher is input to a channel, the conversion value of that channel becomes undefined, and the conversion values of the other channels may also be affected.

(14) AV_{REF0} pin

This is the pin used to input the reference voltage of the A/D converter. Always make the potential at this pin the same as that at the V_{DD} pin even when the A/D converter is not used. The signals input to the ANI0 to ANI11 pins are converted to digital signals based on the voltage applied between the AV_{REF0} and AV_{SS} pins.

(15) AV_{SS} pin

This is the ground pin of the A/D converter. Always make the potential at this pin the same as that at the V_{SS} pin even when the A/D converter is not used.

11.4 Registers

The A/D converter is controlled by the following registers.

- A/D converter mode registers 0, 1, 2 (ADA0M0, ADA0M1, ADA0M2)
- A/D converter channel specification register 0 (ADA0S)
- Power-fail compare mode register (ADA0PFM)

The following registers are also used.

- A/D conversion result register n (ADA0CRn)
- A/D conversion result register nH (ADA0CRnH)
- Power-fail compare threshold value register (ADA0PFT)

(1) A/D converter mode register 0 (ADA0M0)

The ADA0M0 register is an 8-bit register that specifies the operation mode and controls conversion operations.
This register can be read or written in 8-bit or 1-bit units. However, ADA0EF bit is read-only.
Reset sets this register to 00H.

Caution Accessing the ADA0M0 register is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.
• When the CPU operates with the subclock and the main clock oscillation is stopped
• When the CPU operates with the internal oscillation clock

After reset: 00H R/W Address: FFFFF200H

	<7>	6	5	4	3	2	1	<0>
ADA0M0	ADA0CE	0	ADA0MD1	ADA0MD0	ADA0ETS1	ADA0ETS0	ADA0TMD	ADA0EF

ADA0CE	A/D conversion control
0	Stops A/D conversion
1	Enables A/D conversion

ADA0MD1	ADA0MD0	Specification of A/D converter operation mode
0	0	Continuous select mode
0	1	Continuous scan mode
1	0	One-shot select mode
1	1	One-shot scan mode

ADA0ETS1	ADA0ETS0	Specification of external trigger (ADTRG pin) input valid edge
0	0	No edge detection
0	1	Falling edge detection
1	0	Rising edge detection
1	1	Detection of both rising and falling edges

(2/2)

ADA0TMD	Trigger mode specification
0	Software trigger mode
1	External trigger mode/timer trigger mode

ADA0EF	A/D converter status display
0	A/D conversion stopped
1	A/D conversion in progress

Cautions 1. A write operation to this bit is ignored.

<R> **2. Changing the ADA0M1 register bits is prohibited while A/D conversion is enabled (ADA0CE bit = 1).**

<R> **3. Stop the A/D conversion operation (ADA0CE = 0) before writing to the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1).**

- **Normal conversion mode**
- **One-shot select mode/one-shot scan mode in high-speed conversion mode**

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, and ADA0PFT registers are written during A/D conversion (ADA0EF bit = 1) in mode other than above, the following will be performed according to the mode.

- **In software trigger mode**
  A/D conversion is stopped and started again from the beginning.
- **In hardware trigger mode**

A/D conversion is stopped, and the trigger standby state is set.

<R> **4. When the external trigger or timer trigger mode is selected (ADA0TMD bit = 1), set the high-speed conversion mode (ADA0M1.ADA0HS1 bit = 1). Do not input a trigger during the stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0CE bit = 1).**

5. When not using the A/D converter, stop the operation by setting the ADA0CE bit to 0 to reduce the power consumption.

(2) A/D converter mode register 1 (ADA0M1)

The ADA0M1 register is an 8-bit register that specifies the conversion time.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this bit to 00H.

After reset: 00H R/W Address: FFFFF201H

	7	6	5	4	3	2	1	0
ADA0M1	ADA0HS1	0	0	0	0	ADA0FR2	ADA0FR1	ADA0FR0

ADA0HS1	Specification of normal conversion mode/high-speed mode (A/D conversion time)
0	Normal conversion mode
1	High-speed conversion mode

Cautions 1, Changing the ADA0M1 register is prohibited while A/D conversion is enabled (ADA0M0.ADA0CE bit = 1).

<R> **2. When the external trigger or timer trigger mode is selected (ADA0M0.ADA0TMD bit = 1), set high-speed conversion mode (ADA0HS1 bit = 1), and do not input a trigger during the stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0CE = 1).**

3. Be sure to clear bits 6 to 3 to "0".

Remark For A/D conversion time setting examples, see **Tables 11-2** and **11-3**.

<R> **Table 11-2. Conversion Time Selection in Normal Conversion Mode (ADA0HS1 Bit = 0)**

ADA0FR2 to ADA0FR0 Bits	A/D Conversion Time				
	Stabilization Time + Conversion Time + Wait Time	f_{XX} = 20 MHz	f_{XX} = 16 MHz	f_{XX} = 4 MHz	Trigger Response Time
000	66/f_{XX} (13/f_{XX} + 26/f_{XX} + 27/f_{XX})	Setting prohibited	Setting prohibited	16.50 μs	3/f_{XX}
001	131/f_{XX} (26/f_{XX} + 52/f_{XX} + 53/f_{XX})	6.55 μs	8.19 μs	Setting prohibited	3/f_{XX}
010	196/f_{XX} (39/f_{XX} + 78/f_{XX} + 79/f_{XX})	9.80 μs	12.25 μs	Setting prohibited	3/f_{XX}
011	259/f_{XX} (50/f_{XX} + 104/f_{XX} + 105/f_{XX})	12.95 μs	16.19 μs	Setting prohibited	3/f_{XX}
100	311/f_{XX} (50/f_{XX} + 130/f_{XX} + 131/f_{XX})	15.55 μs	19.44 μs	Setting prohibited	3/f_{XX}
101	363/f_{XX} (50/f_{XX} + 156/f_{XX} + 157/f_{XX})	18.15 μs	22.69 μs	Setting prohibited	3/f_{XX}
110	415/f_{XX} (50/f_{XX} + 182/f_{XX} + 183/f_{XX})	20.75 μs	Setting prohibited	Setting prohibited	3/f_{XX}
111	467/f_{XX} (50/f_{XX} + 208/f_{XX} + 209/f_{XX})	23.35 μs	Setting prohibited	Setting prohibited	3/f_{XX}

Remark Stabilization time: A/D converter setup time (1 μs or longer)
Conversion time: Actual A/D conversion time (2.6 to 10.4 μs)
Wait time: Wait time inserted before the next conversion
Trigger response time: If a software trigger, external trigger, or timer trigger is generated after the stabilization time, it is inserted before the conversion time.

In the normal conversion mode, the conversion is started after the stabilization time elapsed from the ADA0M0.ADA0CE bit is set to 1, and A/D conversion is performed only during the conversion time (2.6 to 10.4 μs). Operation is stopped after the conversion ends and the A/D conversion end interrupt request signal (INTAD) is generated after the wait time is elapsed.
Because the conversion operation is stopped during the wait time, operation current can be reduced.

<R> **Caution Set as 2.6 μs ≤ conversion time ≤ 10.4 μs.**

<R> **Table 11-3. Conversion Time Selection in High-Speed Conversion Mode (ADA0HS1 Bit = 1)**

ADA0FR2 to ADA0FR0 Bits	A/D Conversion Time				
	Conversion Time (+ Stabilization Time)	f_{XX} = 20 MHz	f_{XX} = 16 MHz	f_{XX} = 4 MHz	Trigger Response Time
000	26/f_{XX} (+ 13/f_{XX})	Setting prohibited	Setting prohibited	6.5 μs (+ 3.25 μs)	3/f_{XX}
001	52/f_{XX} (+ 26/f_{XX})	2.6 μs (+ 1.3 μs)	3.25 μs (+ 1.625 μs)	Setting prohibited	3/f_{XX}
010	78/f_{XX} (+ 39/f_{XX})	3.9 μs (+ 1.95 μs)	4.875 μs (+ 2.4375 μs)	Setting prohibited	3/f_{XX}
011	104/f_{XX} (+ 50/f_{XX})	5.2 μs (+ 2.5 μs)	6.5 μs (+ 3.125 μs)	Setting prohibited	3/f_{XX}
100	130/f_{XX} (+ 50/f_{XX})	6.5 μs (+ 2.5 μs)	8.125 μs (+ 3.125 μs)	Setting prohibited	3/f_{XX}
101	156/f_{XX} (+ 50/f_{XX})	7.8 μs (+ 2.5 μs)	9.75 μs (+ 3.125 μs)	Setting prohibited	3/f_{XX}
110	182/f_{XX} (+ 50/f_{XX})	9.1 μs (+ 2.5 μs)	Setting prohibited	Setting prohibited	3/f_{XX}
111	208/f_{XX} (+ 50/f_{XX})	10.4 μs (+ 2.5 μs)	Setting prohibited	Setting prohibited	3/f_{XX}

Remark Conversion time: Actual A/D conversion time (2.6 to 10.4 μs)
Stabilization time: A/D converter setup time (1 μs or longer)
Trigger response time: If a software trigger, external trigger, or timer trigger is generated after the stabilization time, it is inserted before the conversion time.

In the high-speed conversion mode, the conversion is started after the stabilization time elapsed from the ADA0M0.ADA0CE bit is set to 1, and A/D conversion is performed only during the conversion time (2.6 to 10.4 μs). The A/D conversion end interrupt request signal (INTAD) is generated immediately after the conversion ends.
In continuous conversion mode, the stabilization time is inserted only before the first conversion, and not inserted after the second conversion (the A/D converter remains running).

<R> **Caution Set as 2.6 μs ≤ conversion time ≤ 10.4 μs.**

(3) A/D converter mode register 2 (ADA0M2)

The ADA0M2 register specifies the hardware trigger mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF203H

	7	6	5	4	3	2	1	0
ADA0M2	0	0	0	0	0	0	ADA0TMD1	ADA0TMD0

ADA0TMD1	ADA0TMD0	Specification of hardware trigger mode
0	0	External trigger mode (when ADTRG pin valid edge detected)
0	1	Timer trigger mode 0 (when INTTP2CC0 interrupt request generated)
1	0	Timer trigger mode 1 (when INTTP2CC1 interrupt request generated)
1	1	Setting prohibited

<R> **Cautions 1. Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0M2 register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1).**

- **Normal conversion mode**
- **One-shot select mode/one-shot scan mode in high-speed conversion mode**

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

(4) A/D converter channel specification register (ADA0S)

The ADA0S register specifies the pin that inputs the analog voltage to be converted into a digital signal.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF202H

	7	6	5	4	3	2	1	0
ADA0S	0	0	0	0	ADA0S3	ADA0S2	ADA0S1	ADA0S0

ADA0S3	ADA0S2	ADA0S1	ADA0S0	Select mode	Scan mode
0	0	0	0	ANI0	ANI0
0	0	0	1	ANI1	ANI0, ANI1
0	0	1	0	ANI2	ANI0 to ANI2
0	0	1	1	ANI3	ANI0 to ANI3
0	1	0	0	ANI4	ANI0 to ANI4
0	1	0	1	ANI5	ANI0 to ANI5
0	1	1	0	ANI6	ANI0 to ANI6
0	1	1	1	ANI7	ANI0 to ANI7
1	0	0	0	ANI8	ANI0 to ANI8
1	0	0	1	ANI9	ANI0 to ANI9
1	0	1	0	ANI10	ANI0 to ANI10
1	0	1	1	ANI11	ANI0 to ANI11
1	1	0	0	Setting prohibited	Setting prohibited
1	1	0	1	Setting prohibited	Setting prohibited
1	1	1	0	Setting prohibited	Setting prohibited
1	1	1	1	Setting prohibited	Setting prohibited

<R> **Cautions 1. Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0S register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1).**

- **Normal conversion mode**
- **One-shot select mode/one-shot scan mode in high-speed conversion mode**

2. Be sure to clear bits 7 to 4 to "0".

(5) A/D conversion result registers n, nH (ADA0CRn, ADA0CRnH)

The ADA0CRn and ADA0CRnH registers store the A/D conversion results.

These registers are read-only, in 16-bit or 8-bit units. However, specify the ADA0CRn register for 16-bit access and the ADA0CRnH register for 8-bit access. The 10 bits of the conversion result are read from the higher 10 bits of the ADA0CRn register, and 0 is read from the lower 6 bits. The higher 8 bits of the conversion result are read from the ADA0CRnH register.

Caution Accessing the ADA0CRn and ADA0CRnH registers is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

After reset: Undefined R Address: ADA0CR0 FFFFF210H, ADA0CR1 FFFFF212H, ADA0CR2 FFFFF214H, ADA0CR3 FFFFF216H, ADA0CR4 FFFFF218H, ADA0CR5 FFFFF21AH, ADA0CR6 FFFFF21CH, ADA0CR7 FFFFF21EH, ADA0CR8 FFFFF220H, ADA0CR9 FFFFF222H, ADA0CR10 FFFFF224H, ADA0CR11 FFFFF226H

ADA0CRn (n = 0 to 11)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	0	0	0	0	0	0

After reset: Undefined R Address: ADA0CR0H FFFFF211H, ADA0CR1H FFFFF213H, ADA0CR2H FFFFF215H, ADA0CR3H FFFFF217H, ADA0CR4H FFFFF219H, ADA0CR5H FFFFF21BH, ADA0CR6H FFFFF21DH, ADA0CR7H FFFFF21FH, ADA0CR8H FFFFF221H, ADA0CR9H FFFFF223H, ADA0CR10H FFFFF225H, ADA0CR11H FFFFF227H

ADA0CRnH (n = 0 to 11)

7	6	5	4	3	2	1	0
AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2

Caution A write operation to the ADA0M0 and ADA0S registers may cause the contents of the ADA0CRn register to become undefined. After the conversion, read the conversion result before writing to the ADA0M0 and ADA0S registers. Correct conversion results may not be read if a sequence other than the above is used.

The relationship between the analog voltage input to the analog input pins (ANI0 to ANI11) and the A/D conversion result (ADA0CRn register) is as follows.

$$SAR = INT\left(\frac{V_{IN}}{AV_{REF0}} \times 1{,}024 + 0.5\right)$$

$$ADA0CR^{Note} = SAR \times 64$$

Or,

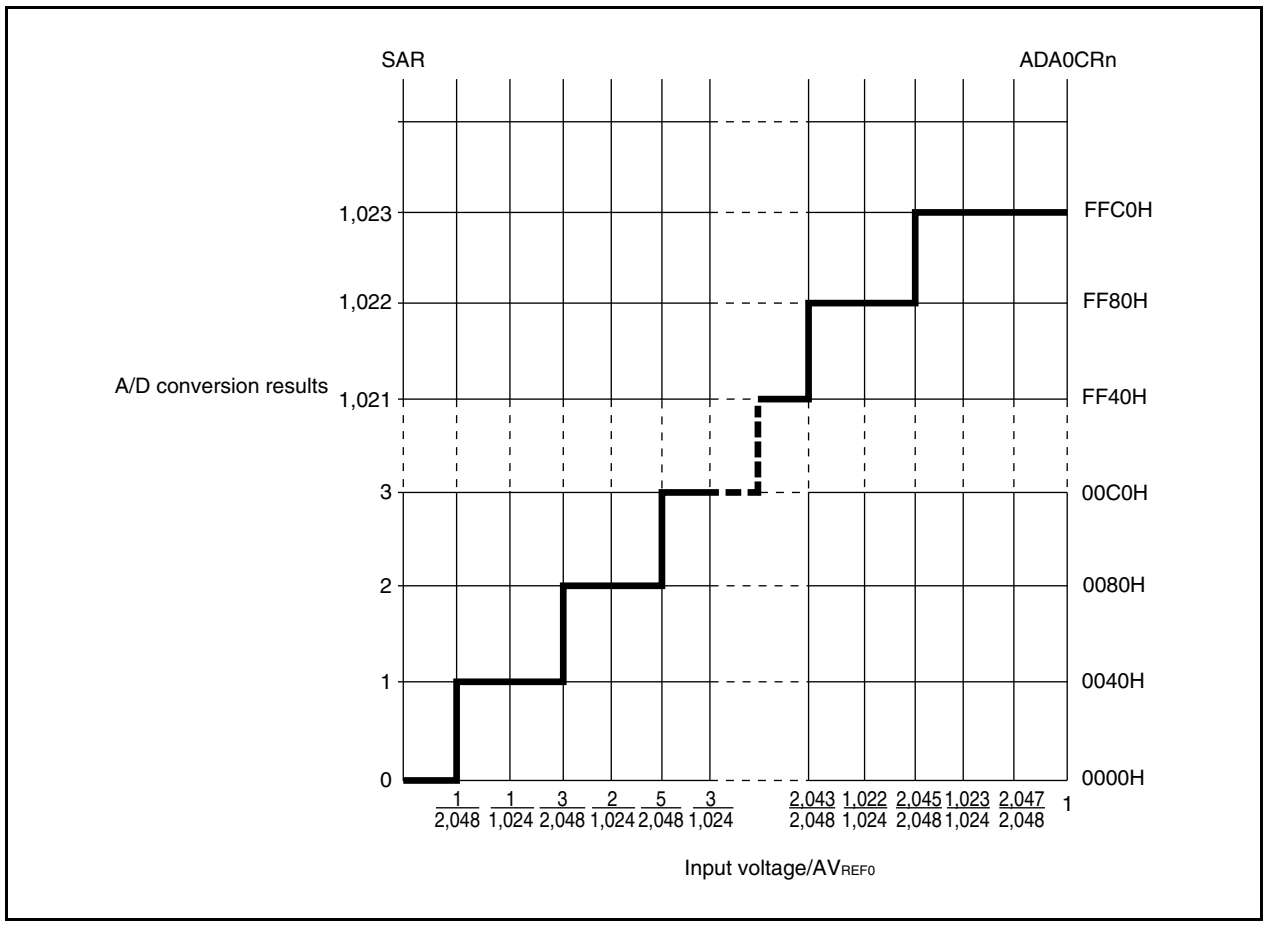
$$(SAR - 0.5) \times \frac{AV_{REF0}}{1{,}024} \leq V_{IN} < (SAR + 0.5) \times \frac{AV_{REF0}}{1{,}024}$$

INT( ): Function that returns the integer of the value in ( )
V_{IN}: Analog input voltage
AV_{REF0}: AV_{REF0} pin voltage
ADA0CR: Value of ADA0CRn register

Note The lower 6 bits of the ADA0CRn register are fixed to 0.

The following shows the relationship between the analog input voltage and the A/D conversion results.

Figure 11-2. Relationship Between Analog Input Voltage and A/D Conversion Results



(6) Power-fail compare mode register (ADA0PFM)

The ADA0PFM register is an 8-bit register that sets the power-fail compare mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF204H

	<7>	6	5	4	3	2	1	0
ADA0PFM	ADA0PFE	ADA0PFC	0	0	0	0	0	0

ADA0PFE	Selection of power-fail compare enable/disable
0	Power-fail compare disabled
1	Power-fail compare enabled

ADA0PFC	Selection of power-fail compare mode
0	Generates an interrupt request signal (INTAD) when ADA0CRnH ≥ ADA0PFT
1	Generates an interrupt request signal (INTAD) when ADA0CRnH < ADA0PFT

Cautions 1. In the select mode, the 8-bit data set to the ADA0PFT register is compared with the value of the ADA0CRnH register specified by the ADA0S register. If the result matches the condition specified by the ADA0PFC bit, the conversion result is stored in the ADA0CRn register and the INTAD signal is generated. If it does not match, however, the interrupt signal is not generated.

2. In the scan mode, the 8-bit data set to the ADA0PFT register is compared with the contents of the ADA0CR0H register. If the result matches the condition specified by the ADA0PFC bit, the conversion result is stored in the ADA0CR0 register and the INTAD signal is generated. If it does not match, however, the INTAD signal is not generated. Regardless of the comparison result, the scan operation is continued and the conversion result is stored in the ADA0CRn register until the scan operation is completed. However, the INTAD signal is not generated after the scan operation has been completed.

<R> **3. Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0PFM register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1).**

- **Normal conversion mode**
- **One-shot select mode/one-shot scan mode in high-speed conversion mode**

(7) Power-fail compare threshold value register (ADA0PFT)

The ADA0PFT register sets the compare value in the power-fail compare mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF205H

	7	6	5	4	3	2	1	0
ADA0PFT								

<R> **Caution Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0PFT register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1).**

- **Normal conversion mode**
- **One-shot select mode/one-shot scan mode in high-speed conversion mode**

11.5 Operation

11.5.1 Basic operation

<1> Set the operation mode, trigger mode, and conversion time for executing A/D conversion by using the ADA0M0, ADA0M1, ADA0M2, and ADA0S registers. When the ADA0M0.ADA0CE bit are set, conversion is started in the software trigger mode and the A/D converter waits for a trigger in the external or timer trigger mode.

<2> When A/D conversion is started, the voltage input to the selected analog input channel is sampled by the sample & hold circuit.

<3> When the sample & hold circuit samples the input channel for a specific time, it enters the hold status, and holds the input analog voltage until A/D conversion is complete.

<4> Set bit 9 of the successive approximation register (SAR). The tap selector selects (1/2) AV_{REF0} as the voltage tap of the series resistor string.

<5> The voltage difference between the voltage of the series resistor string and the analog input voltage is compared by the voltage comparator. If the analog input voltage is higher than (1/2) AV_{REF0}, the MSB of the SAR register remains set. If it is lower than (1/2) AV_{REF0}, the MSB is reset.

<6> Next, bit 8 of the SAR register is automatically set and the next comparison is started. Depending on the value of bit 9, to which a result has been already set, the voltage tap of the series resistor string is selected as follows.

- Bit 9 = 1: (3/4) AV_{REF0}
- Bit 9 = 0: (1/4) AV_{REF0}

This voltage tap and the analog input voltage are compared and, depending on the result, bit 8 is manipulated as follows.

Analog input voltage ≥ Voltage tap: Bit 8 = 1

Analog input voltage ≤ Voltage tap: Bit 8 = 0

<7> This comparison is continued to bit 0 of the SAR register.

<8> When comparison of the 10 bits is complete, the valid digital result is stored in the SAR register, which is then transferred to and stored in the ADA0CRn register. After that, an A/D conversion end interrupt request signal (INTAD) is generated.

<9> In one-shot select mode, conversion is stopped[Note]. In one-shot scan mode, conversion is stopped after scanning once[Note]. In continuous select mode, repeat steps <2> to <8> until the ADA0M0.ADA0CE bit is cleared to 0. In continuous scan mode, repeat steps <2> to <8> for each channel.

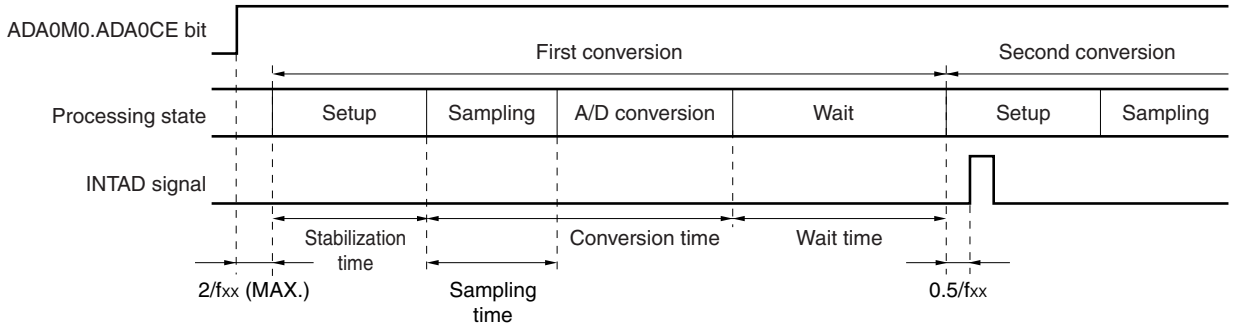
Note In the external trigger mode, timer trigger mode 0, or timer trigger mode 1, the trigger standby status is entered.

<R> **Remark** The trigger standby status means the status after the stabilization time has elapsed.

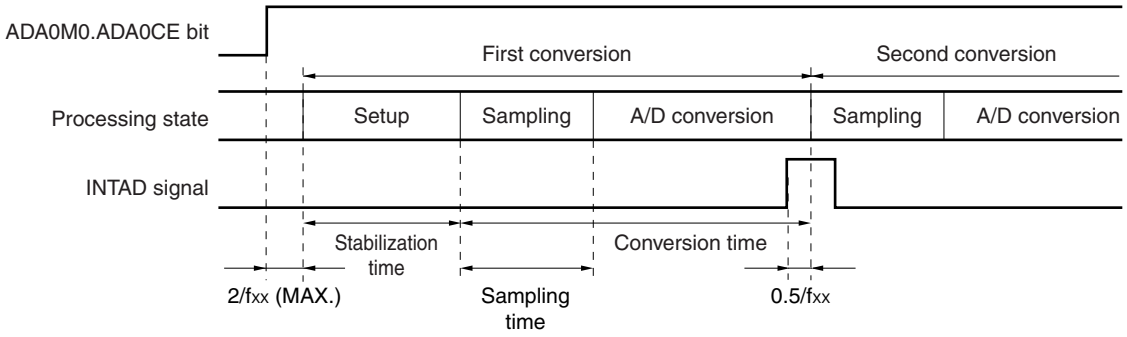
11.5.2 Conversion operation timing

<R> **Figure 11-3. Conversion Operation Timing (Continuous Conversion)**

(1) Operation in normal conversion mode (ADA0HS1 bit = 0)



(2) Operation in high-speed conversion mode (ADA0HS1 bit = 1)



ADA0FR2 to ADA0FR0 Bits	Stabilization Time	Conversion Time (Sampling Time)	Wait Time	Trigger Response Time
000	13/fxx	26/fxx (4/fxx)	27fxx	3/fxx
001	26/fxx	52/fxx (8/fxx)	53/fxx	3/fxx
010	39/fxx	78/fxx (12/fxx)	79/fxx	3/fxx
011	50/fxx	104/fxx (16/fxx)	105/fxx	3/fxx
100	50/fxx	130/fxx (20/fxx)	131/fxx	3/fxx
101	50/fxx	156/fxx (24/fxx)	157/fxx	3/fxx
110	50/fxx	182/fxx (28/fxx)	183/fxx	3/fxx
111	50/fxx	208/fxx (32/fxx)	209/fxx	3/fxx

Remark The above timings are when a trigger generates within the stabilization time. If the trigger generates after the stabilization time, a trigger response time is inserted.

11.5.3 Trigger mode

The timing of starting the conversion operation is specified by setting a trigger mode. The trigger mode includes a software trigger mode and hardware trigger modes. The hardware trigger modes include timer trigger modes 0 and 1, and external trigger mode. The ADA0M0.ADA0TMD bit is used to set the trigger mode. The hardware trigger modes are set by the ADA0M2.ADA0TMD1 and ADA0M2.ADA0TMD0 bits.

(1) Software trigger mode

When the ADA0M0.ADA0CE bit is set to 1, the signal of the analog input pin (ANI0 to ANI11 pin) specified by the ADA0S register is converted. When conversion is complete, the result is stored in the ADA0CRn register. At the same time, the A/D conversion end interrupt request signal (INTAD) is generated.

If the operation mode specified by the ADA0M0.ADA0MD1 and ADA0M0.ADA0MD0 bits is the continuous select/scan mode, the next conversion is started, unless the ADA0CE bit is cleared to 0 after completion of the first conversion. Conversion is performed once and ends if the operation mode is the one-shot select/scan mode.

When conversion is started, the ADA0M0.ADA0EF bit is set to 1 (indicating that conversion is in progress).

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during conversion, the conversion is aborted and started again from the beginning. However, writing to these registers is prohibited in
<R> the normal conversion mode and one-shot select mode/one-shot scan mode of the high-speed conversion mode.

(2) External trigger mode

In this mode, converting the signal of the analog input pin (ANI0 to ANI11) specified by the ADA0S register is started when an external trigger is input (to the ADTRG pin). Which edge of the external trigger is to be detected (i.e., the rising edge, falling edge, or both rising and falling edges) can be specified by using the ADA0M0.ADA0ETS1 and ADA0M0.ATA0ETS0 bits. When the ADA0CE bit is set to 1, the A/D converter waits for the trigger, and starts conversion after the external trigger has been input.

When conversion is completed, the result of conversion is stored in the ADA0CRn register, regardless of whether the continuous select, continuous scan, one-shot select, or one-shot scan mode is set as the operation mode by the ADA0MD1 and ADA0MD0 bits. At the same time, the INTAD signal is generated, and the A/D converter waits for the trigger again.

When conversion is started, the ADA0EF bit is set to 1 (indicating that conversion is in progress). While the A/D converter is waiting for the trigger, however, the ADA0EF bit is cleared to 0 (indicating that conversion is stopped). If the valid trigger is input during the conversion operation, the conversion is aborted and started again from the beginning.

If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during the conversion operation,
<R> the conversion is not aborted, and the A/D converter waits for the trigger again. However, writing to these registers is prohibited in the one-shot select mode/one-shot scan mode.

<R> **Caution When the external trigger mode is selected, set the high-speed conversion mode. Do not input a trigger during stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0M0.ADA0CE bit = 1).**

<R> **Remark** The trigger standby status means the status after the stabilization time has elapsed.

(3) Timer trigger mode

In this mode, converting the signal of the analog input pin (ANI0 to ANI11) specified by the ADA0S register is started by the compare match interrupt request signal (INTTP2CC0 or INTTP2CC1) of the capture/compare register connected to the timer. The INTTP2CC0 or INTTP2CC1 signal is selected by the ADA0TMD1 and ADA0TMD0 bits, and conversion is started at the rising edge of the specified compare match interrupt request signal. When the ADA0CE bit is set to 1, the A/D converter waits for a trigger, and starts conversion when the compare match interrupt request signal of the timer is input.

When conversion is completed, regardless of whether the continuous select, continuous scan, one-shot select, or one-shot scan mode is set as the operation mode by the ADA0MD1 and ADA0MD0 bits, the result of the conversion is stored in the ADA0CRn register. At the same time, the INTAD signal is generated, and the A/D converter waits for the trigger again.

When conversion is started, the ADA0EF bit is set to 1 (indicating that conversion is in progress). While the A/D converter is waiting for the trigger, however, the ADA0EF bit is cleared to 0 (indicating that conversion is stopped). If the valid trigger is input during the conversion operation, the conversion is aborted and started again from the beginning.

<R> If the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written during conversion, the conversion is stopped and the A/D converter waits for the trigger again. However, writing to these registers is prohibited in the one-shot select mode/one-shot scan mode.

<R> **Caution When the external trigger mode is selected, set the high-speed conversion mode. Do not input a trigger during stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0M0.ADA0CE bit = 1).**

<R> **Remark** The trigger standby status means the status after the stabilization time has elapsed.

11.5.4 Operation mode

Four operation modes are available as the modes in which to set the ANI0 to ANI11 pins: continuous select mode, continuous scan mode, one-shot select mode, and one-shot scan mode.

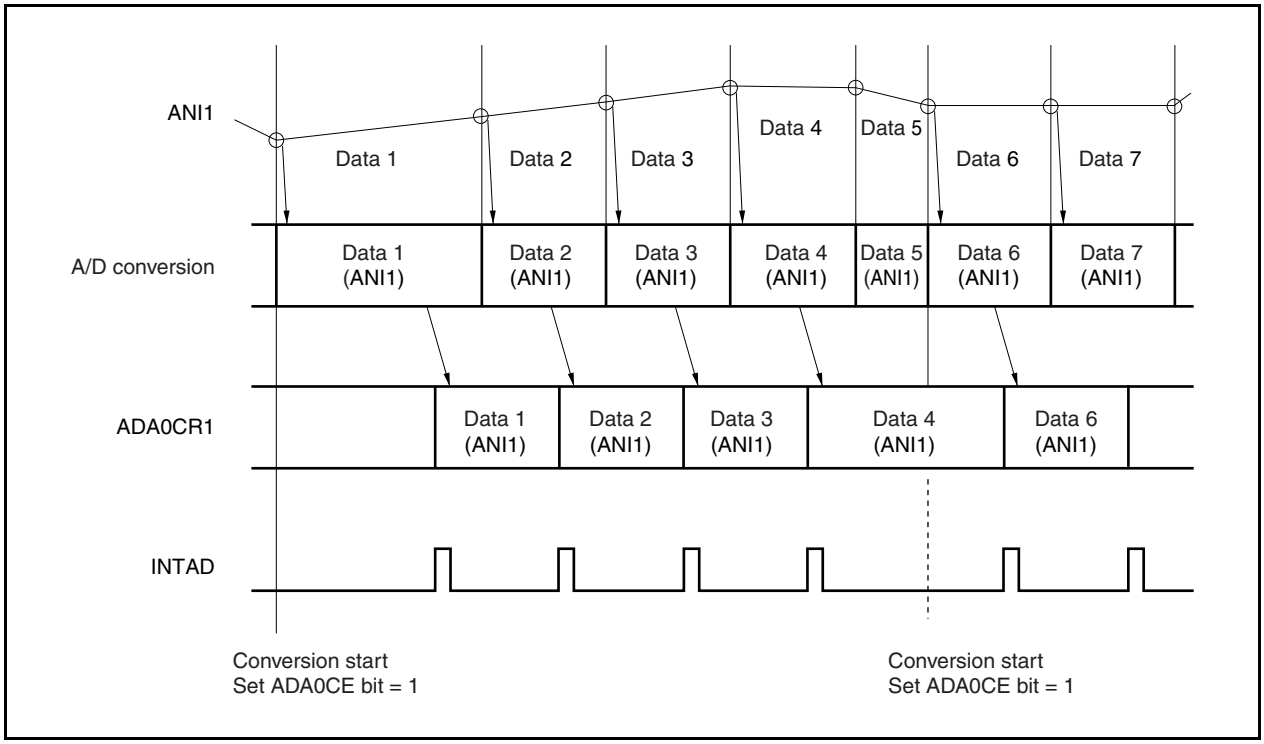
The operation mode is selected by the ADA0M0.ADA0MD1 and ADA0M0.ADA0MD0 bits.

(1) Continuous select mode

In this mode, the voltage of one analog input pin selected by the ADA0S register is continuously converted into a digital value.

The conversion result is stored in the ADA0CRn register corresponding to the analog input pin. In this mode, an analog input pin corresponds to an ADA0CRn register on a one-to-one basis. Each time A/D conversion is completed, the A/D conversion end interrupt request signal (INTAD) is generated. After completion of conversion, the next conversion is started, unless the ADA0M0.ADA0CE bit is cleared to 0 (n = 0 to 11).

Figure 11-4. Timing Example of Continuous Select Mode Operation (ADA0S Register = 01H)



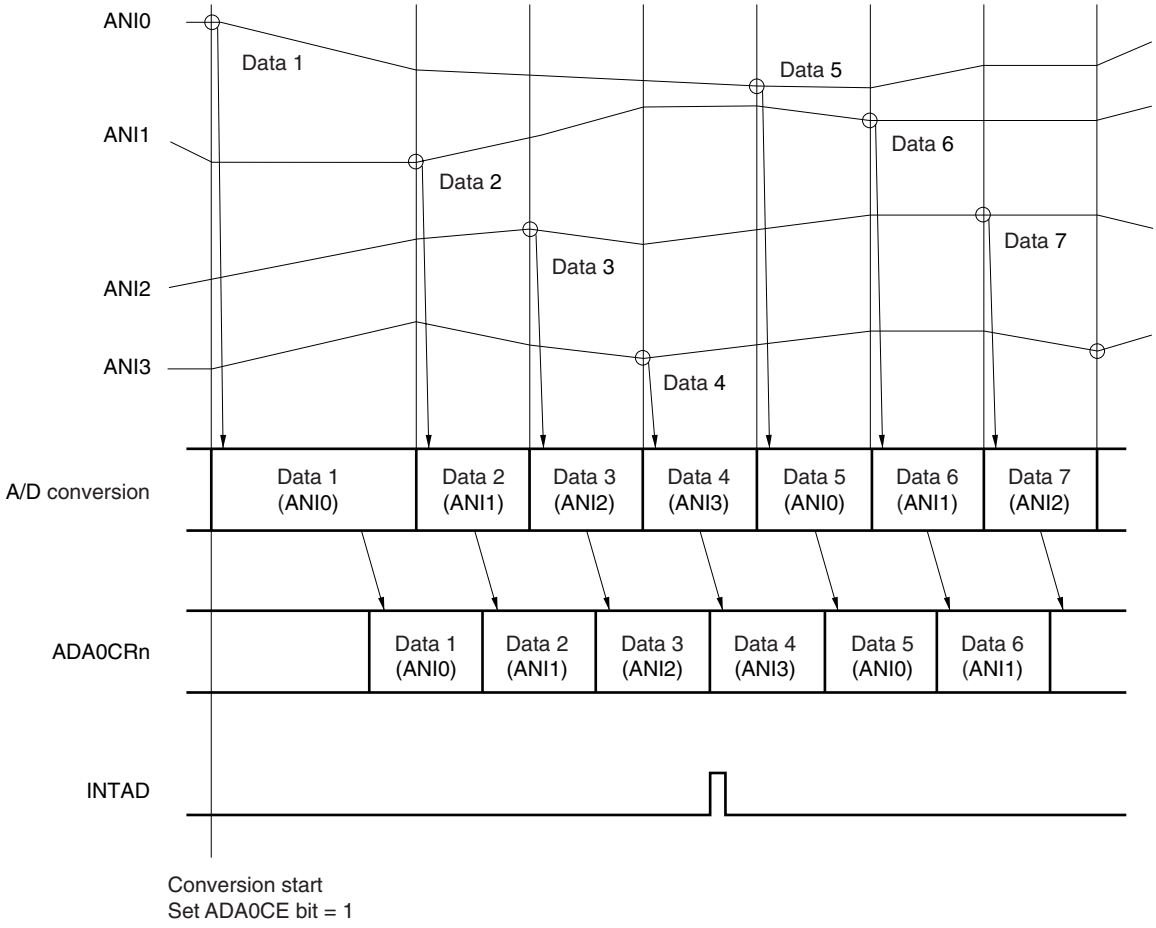
(2) Continuous scan mode

In this mode, analog input pins are sequentially selected, from the ANI0 pin to the pin specified by the ADA0S register, and their values are converted into digital values.

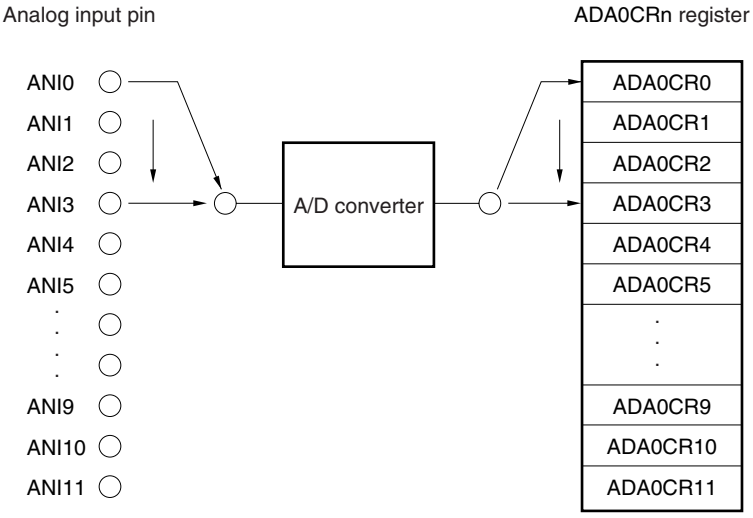
The result of each conversion is stored in the ADA0CRn register corresponding to the analog input pin. When conversion of the analog input pin specified by the ADA0S register is complete, the INTAD signal is generated, and A/D conversion is started again from the ANI0 pin, unless the ADA0CE bit is cleared to 0 (n = 0 to 11).

Figure 11-5. Timing Example of Continuous Scan Mode Operation (ADA0S Register = 03H)

(a) Timing example



(b) Block diagram

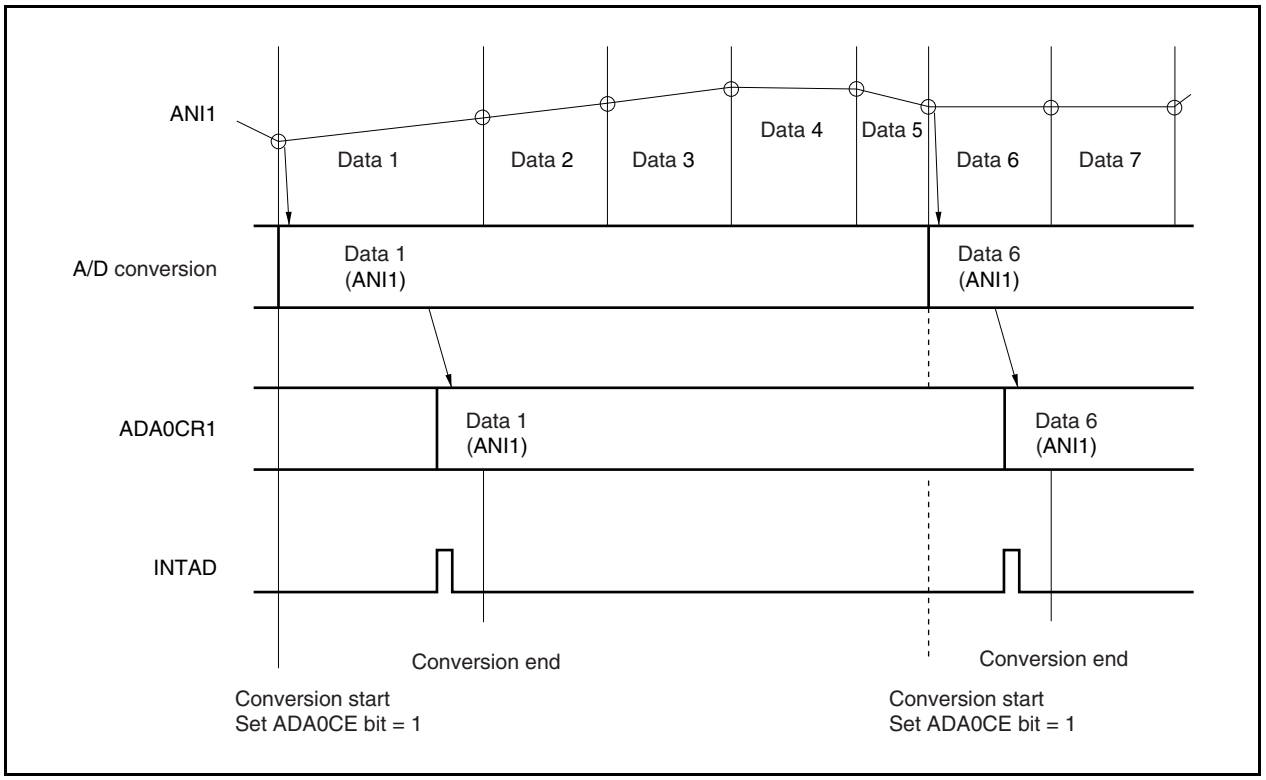


(3) One-shot select mode

In this mode, the voltage on the analog input pin specified by the ADA0S register is converted into a digital value only once.

The conversion result is stored in the ADA0CRn register corresponding to the analog input pin. In this mode, an analog input pin and an ADA0CRn register correspond on a one-to-one basis. When A/D conversion has been completed once, the INTAD signal is generated. The A/D conversion operation is stopped after it has been completed (n = 0 to 11).

Figure 11-6. Timing Example of One-Shot Select Mode Operation (ADA0S Register = 01H)



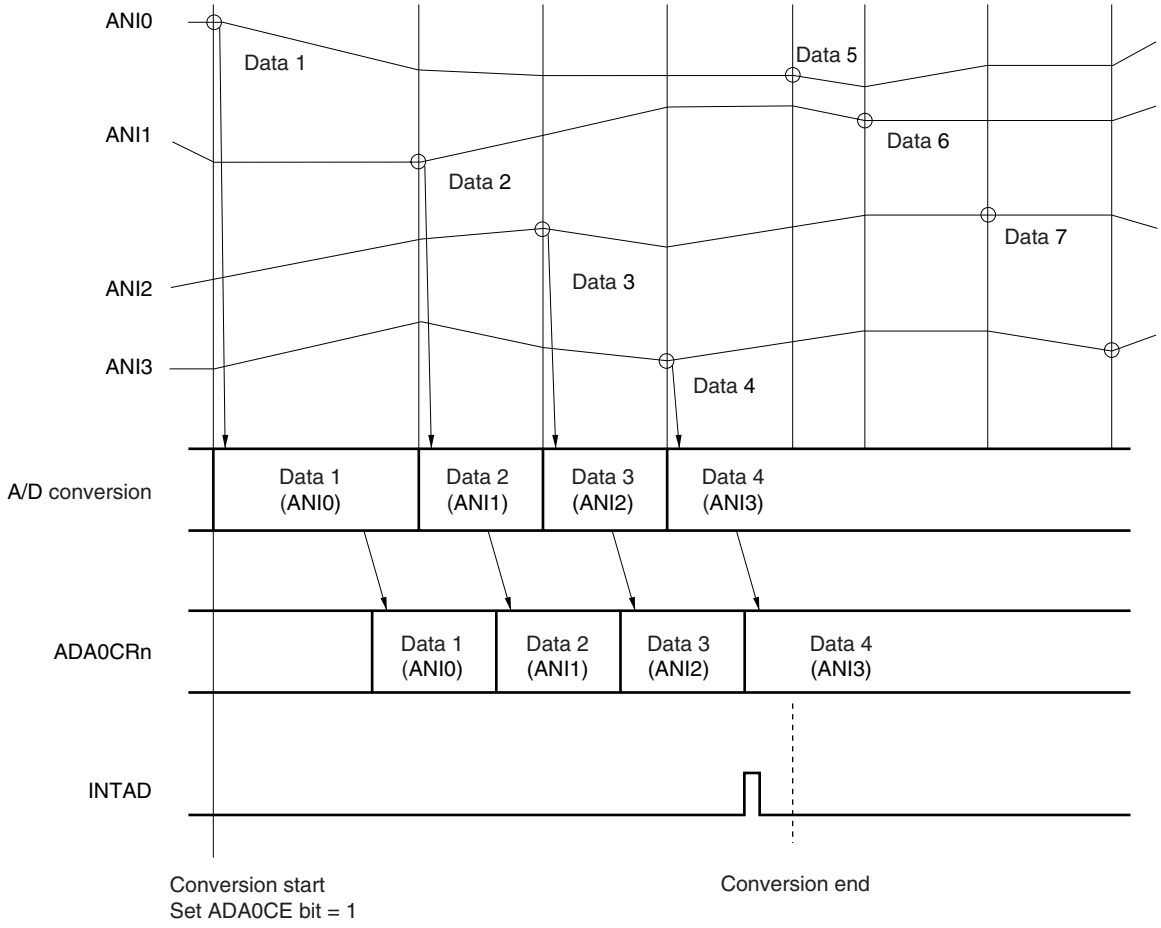
(4) One-shot scan mode

In this mode, analog input pins are sequentially selected, from the ANI0 pin to the pin specified by the ADA0S register, and their values are converted into digital values .

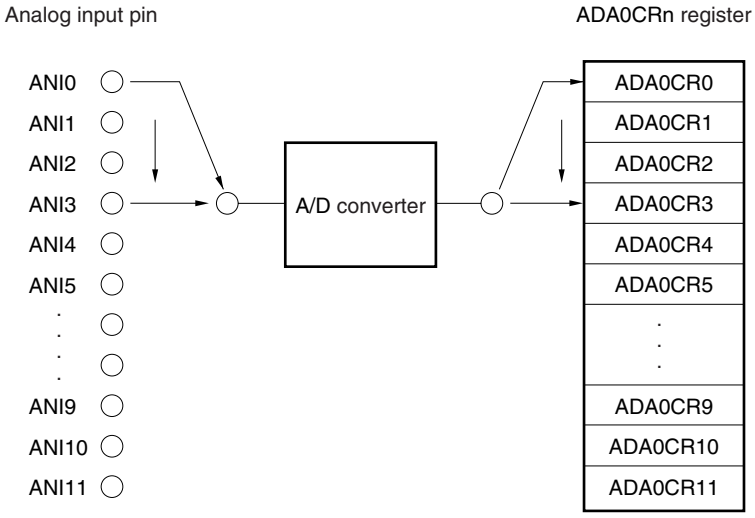
Each conversion result is stored in the ADA0CRn register corresponding to the analog input pin. When conversion of the analog input pin specified by the ADA0S register is complete, the INTAD signal is generated. A/D conversion is stopped after it has been completed (n = 0 to 11).

Figure 11-7. Timing Example of One-Shot Scan Mode Operation (ADA0S Register = 03H)

(a) Timing example



(b) Block diagram



11.5.5 Power-fail compare mode

The A/D conversion end interrupt request signal (INTAD) can be controlled as follows by the ADA0PFM and ADA0PFT registers.

- When the ADA0PFM.ADA0PFE bit = 0, the INTAD signal is generated each time conversion is completed (normal use of the A/D converter).
- When the ADA0PFE bit = 1 and when the ADA0PFM.ADA0PFC bit = 0, the value of the ADA0CRnH register is compared with the value of the ADA0PFT register when conversion is completed, and the INTAD signal is generated only if ADA0CRnH ≥ ADA0PFT.
- When the ADA0PFE bit = 1 and when the ADA0PFC bit = 1, the value of the ADA0CRnH register is compared with the value of the ADA0PFT register when conversion is completed, and the INTAD signal is generated only if ADA0CRnH < ADA0PFT.

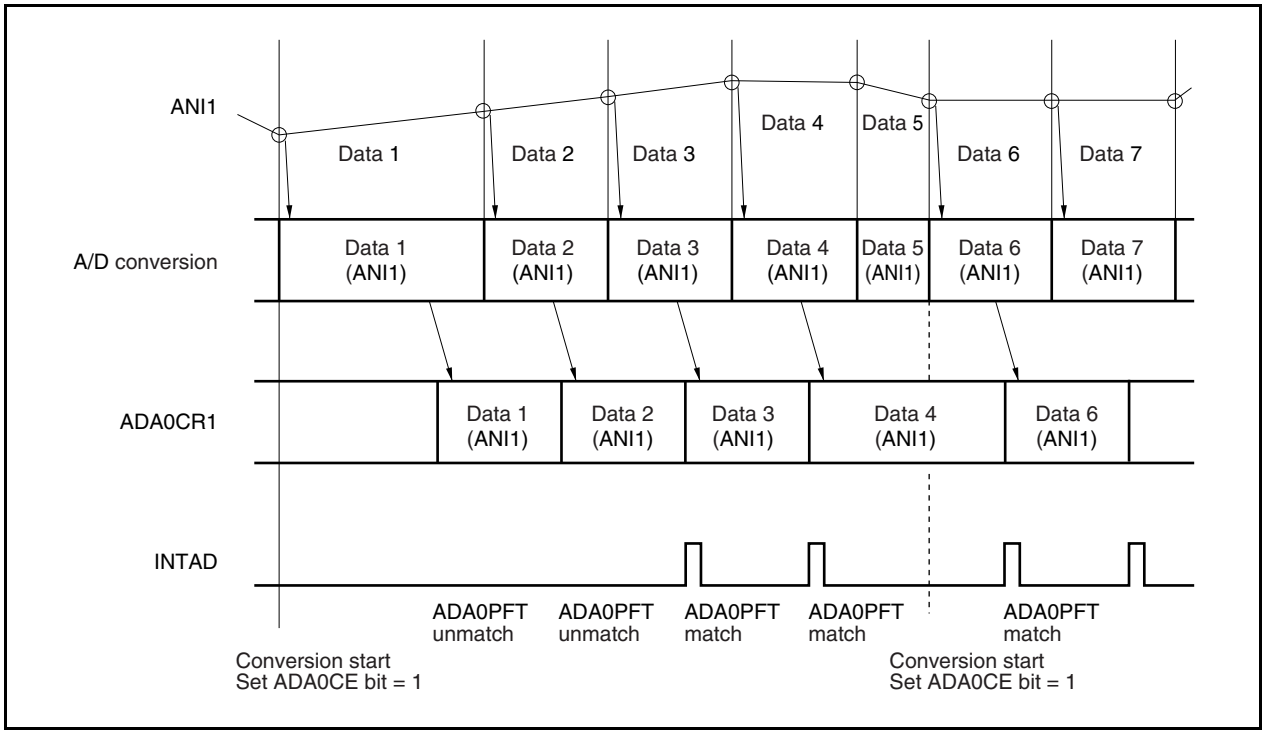
Remark n = 0 to 11

In the power-fail compare mode, four modes are available as modes in which to set the ANI0 to ANI11 pins: continuous select mode, continuous scan mode, one-shot select mode, and one-shot scan mode.

(1) Continuous select mode

In this mode, the result of converting the voltage of the analog input pin specified by the ADA0S register is compared with the set value of the ADA0PFT register. If the result of power-fail comparison matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CRn register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CRn register, and the INTAD signal is not generated. After completion of the first conversion, the next conversion is started, unless the ADA0M0.ADA0CE bit is cleared to 0 (n = 0 to 11).

Figure 11-8. Timing Example of Continuous Select Mode Operation (When Power-Fail Comparison Is Made: ADA0S Register = 01H)



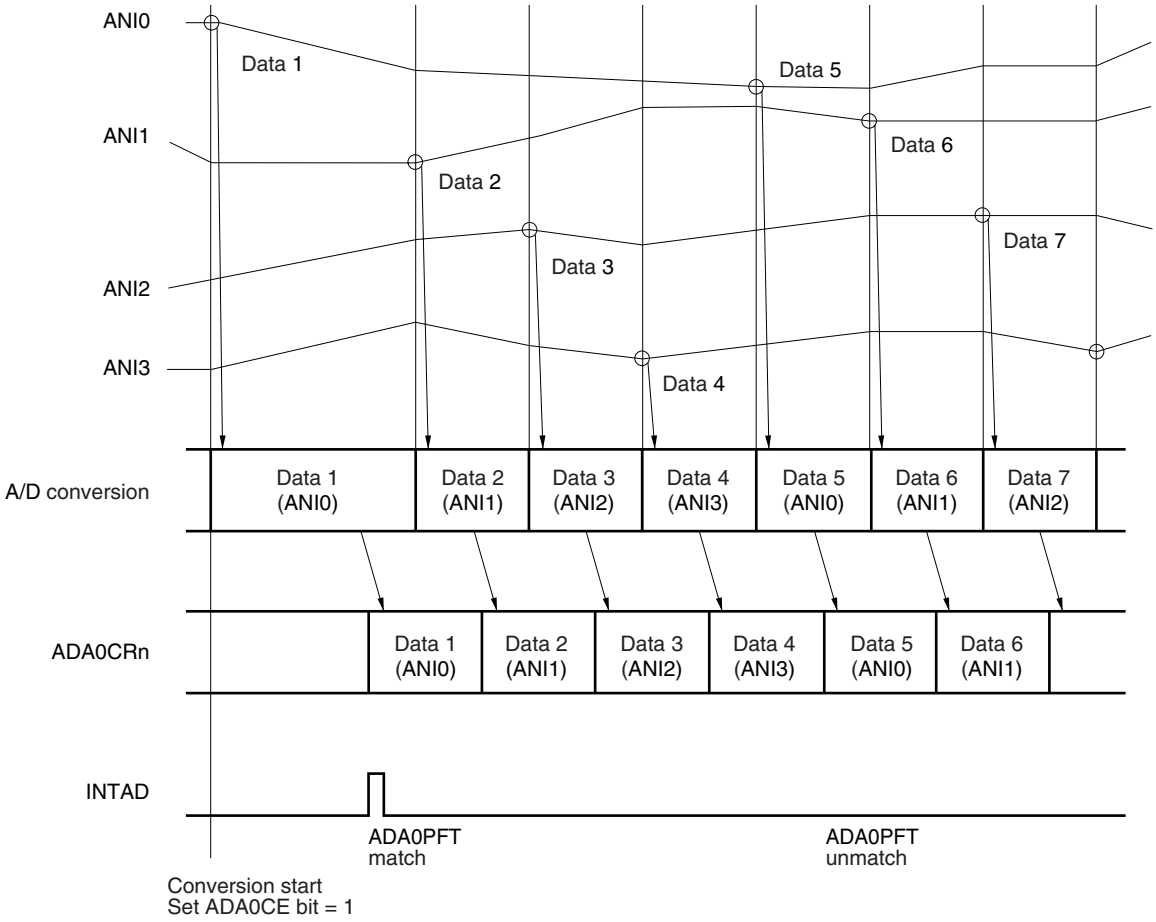
(2) Continuous scan mode

In this mode, the results of converting the voltages of the analog input pins sequentially selected from the ANI0 pin to the pin specified by the ADA0S register are stored, and the set value of the ADA0CR0H register of channel 0 is compared with the value of the ADA0PFT register. If the result of power-fail comparison matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CR0 register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CR0 register, and the INTAD signal is not generated.

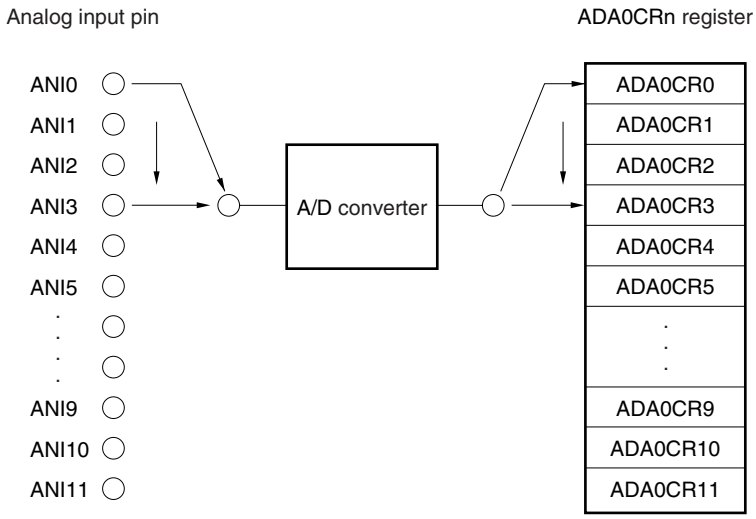
After the result of the first conversion has been stored in the ADA0CR0 register, the results of sequentially converting the voltages on the analog input pins up to the pin specified by the ADA0S register are continuously stored. After completion of conversion, the next conversion is started from the ANI0 pin again, unless the ADA0CE bit is cleared to 0.

Figure 11-9. Timing Example of Continuous Scan Mode Operation (When Power-Fail Comparison Is Made: ADA0S Register = 03H)

(b) Timing example



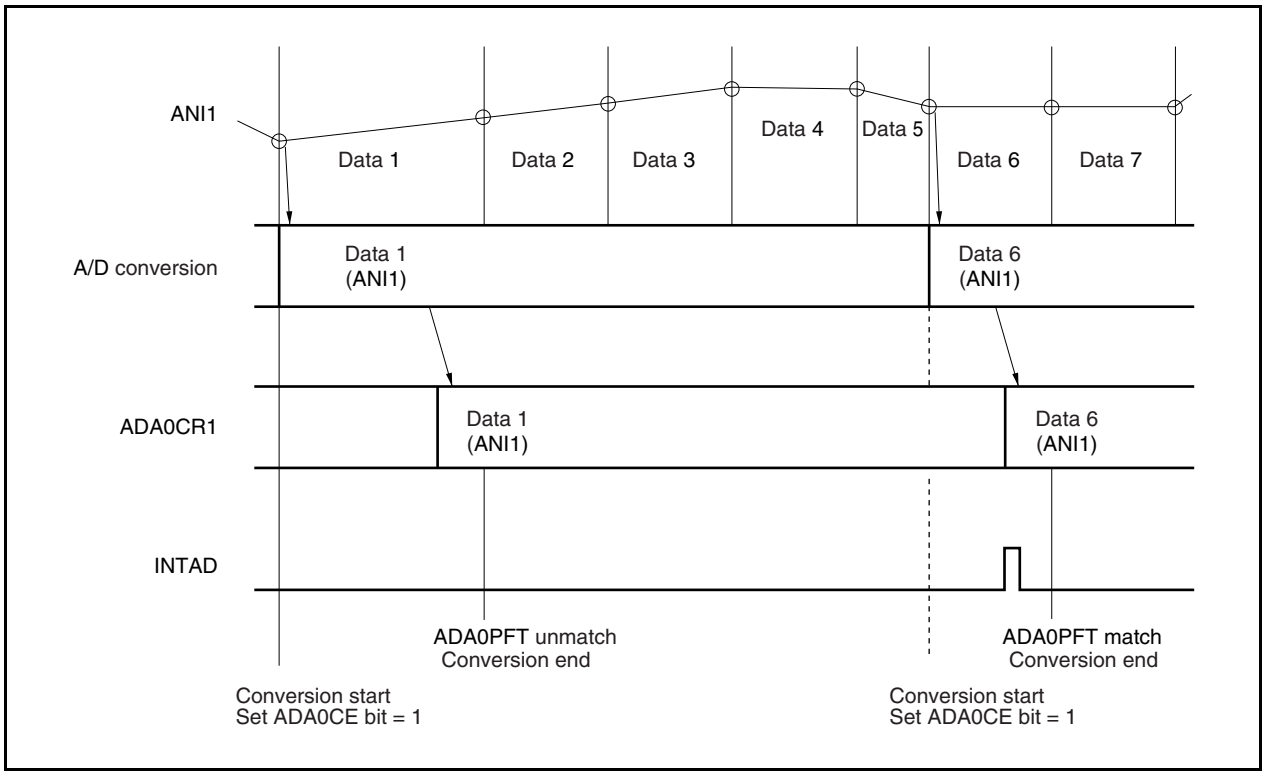
(b) Block diagram



(3) One-shot select mode

In this mode, the result of converting the voltage of the analog input pin specified by the ADA0S register is compared with the set value of the ADA0PFT register. If the result of power-fail comparison matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CRn register, and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CRn register, and the INTAD signal is not generated. Conversion is stopped after it has been completed.

Figure 11-10. Timing Example of One-Shot Select Mode Operation (When Power-Fail Comparison Is Made: ADA0S Register = 01H)

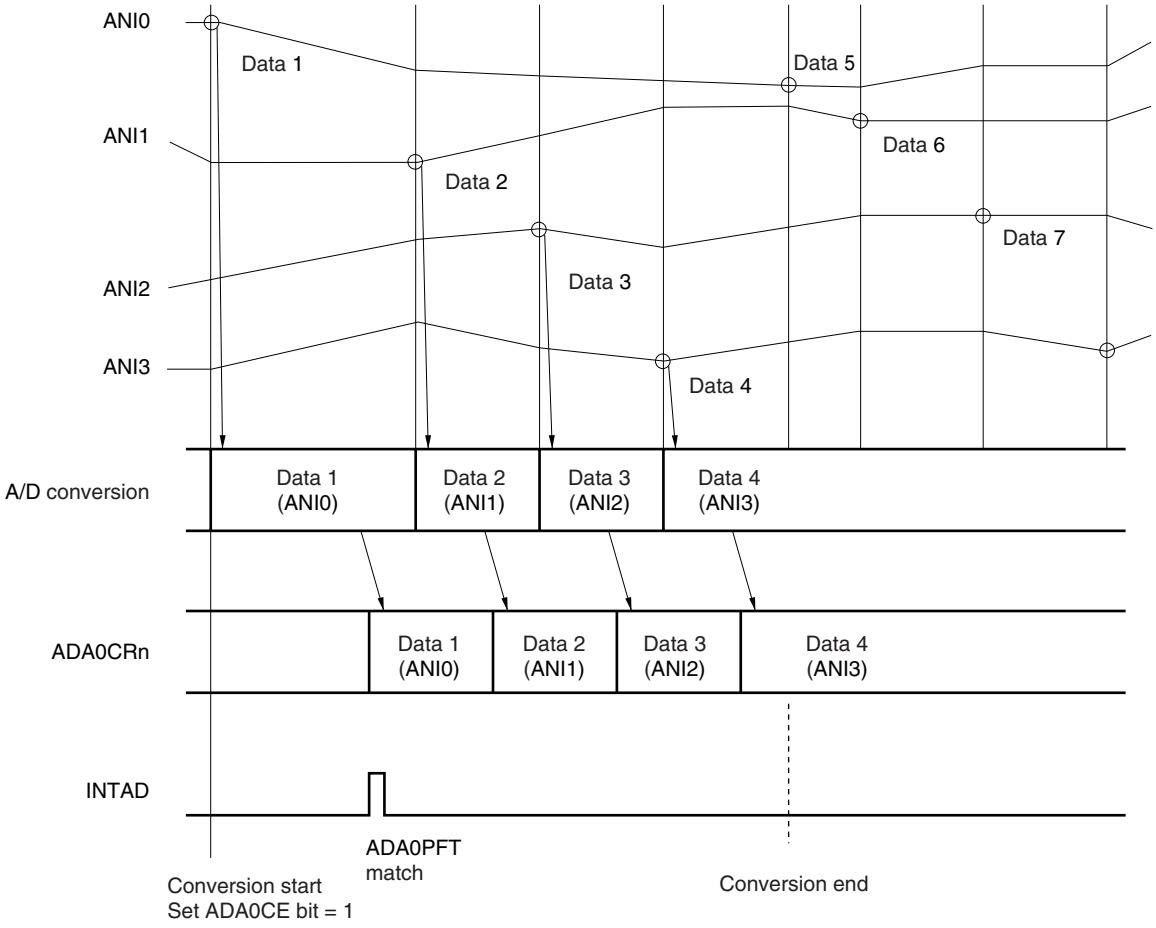


(4) One-shot scan mode

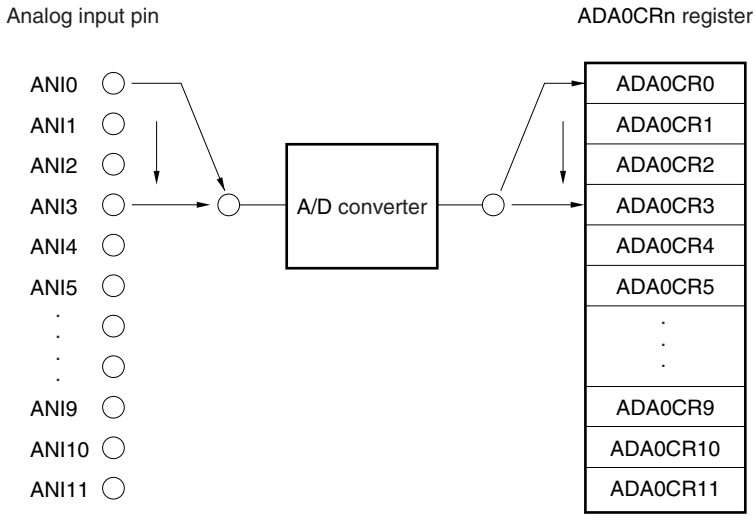
In this mode, the results of converting the voltages of the analog input pins sequentially selected from the ANI0 pin to the pin specified by the ADA0S register are stored, and the set value of the ADA0CR0H register of channel 0 is compared with the set value of the ADA0PFT register. If the result of power-fail comparison matches the condition set by the ADA0PFC bit, the conversion result is stored in the ADA0CR0 register and the INTAD signal is generated. If it does not match, the conversion result is stored in the ADA0CR0 register, and the INTAD0 signal is not generated. After the result of the first conversion has been stored in the ADA0CR0 register, the results of converting the signals on the analog input pins specified by the ADA0S register are sequentially stored. The conversion is stopped after it has been completed.

Figure 11-11. Timing Example of One-Shot Scan Mode Operation (When Power-Fail Comparison Is Made: ADA0S Register = 03H)

(a) Timing example



(b) Block diagram



11.6 Cautions

(1) When A/D converter is not used

When the A/D converter is not used, the power consumption can be reduced by clearing the ADA0M0.ADA0CE bit to 0.

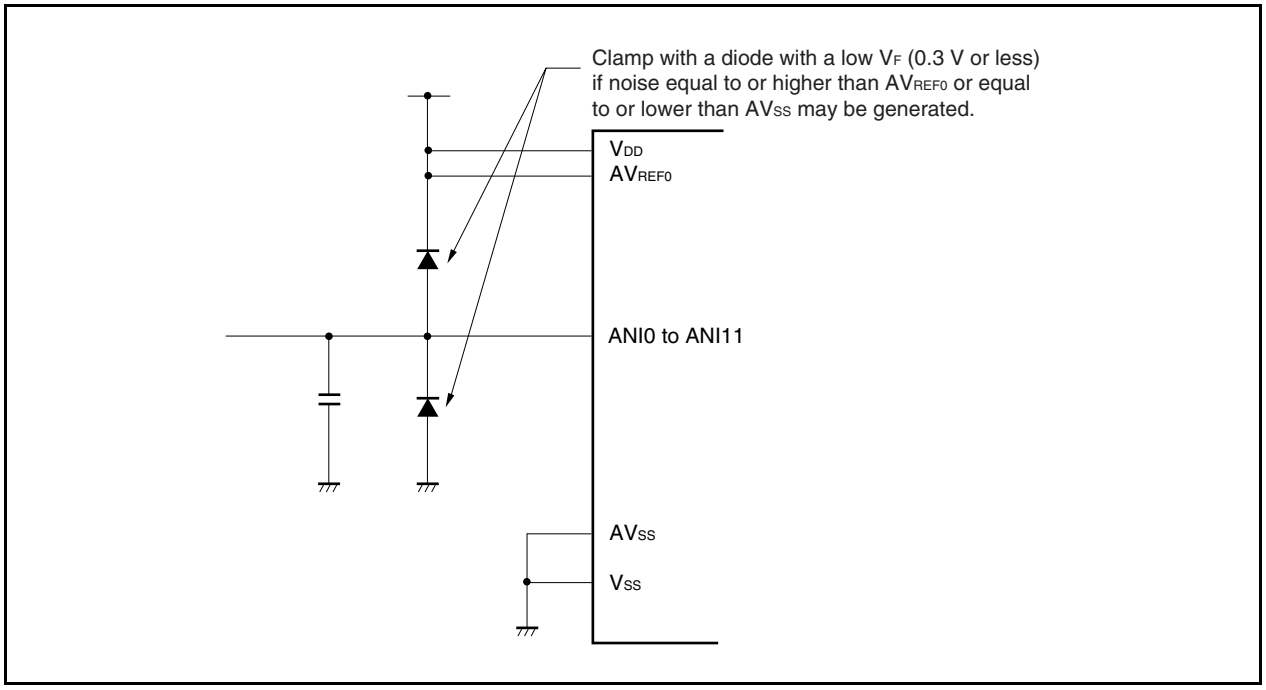
(2) Input range of ANI0 to ANI11 pins

Input the voltage within the specified range to the ANI0 to ANI11 pins. If a voltage equal to or higher than AV_{REF0} or equal to or lower than AV_{SS} (even within the range of the absolute maximum ratings) is input to any of these pins, the conversion value of that channel is undefined, and the conversion value of the other channels may also be affected.

(3) Countermeasures against noise

To maintain the 10-bit resolution, the ANI0 to ANI11 pins must be effectively protected from noise. The influence of noise increases as the output impedance of the analog input source becomes higher. To lower the noise, connecting an external capacitor as shown in Figure 11-12 is recommended.

Figure 11-12. Processing of Analog Input Pin



<R> **(4) Alternate I/O**

The analog input (ANI0 to ANI11) pins are multiplexed with port pins. The AV_{REF0} power pin is multiplexed with the reference power supply to the A/D converter and the I/O buffer power supply of port 7. If any of the following processings is performed during A/D conversion, therefore, the expected A/D conversion value may not be obtained.

(a) If a digital pulse is applied to a pin adjacent to a pin whose input analog signal is converted into a digital signal (for example, P72 and P74 pins during ANI3 conversion) (cause: influence of coupling noise)

(b) If AV_{REF0} power supply fluctuates as a result of executing an instruction to read the P7H or P7L register to the input port during A/D conversion or an instruction to write data to the output port (cause: influence on the AV_{REF0} power supply)

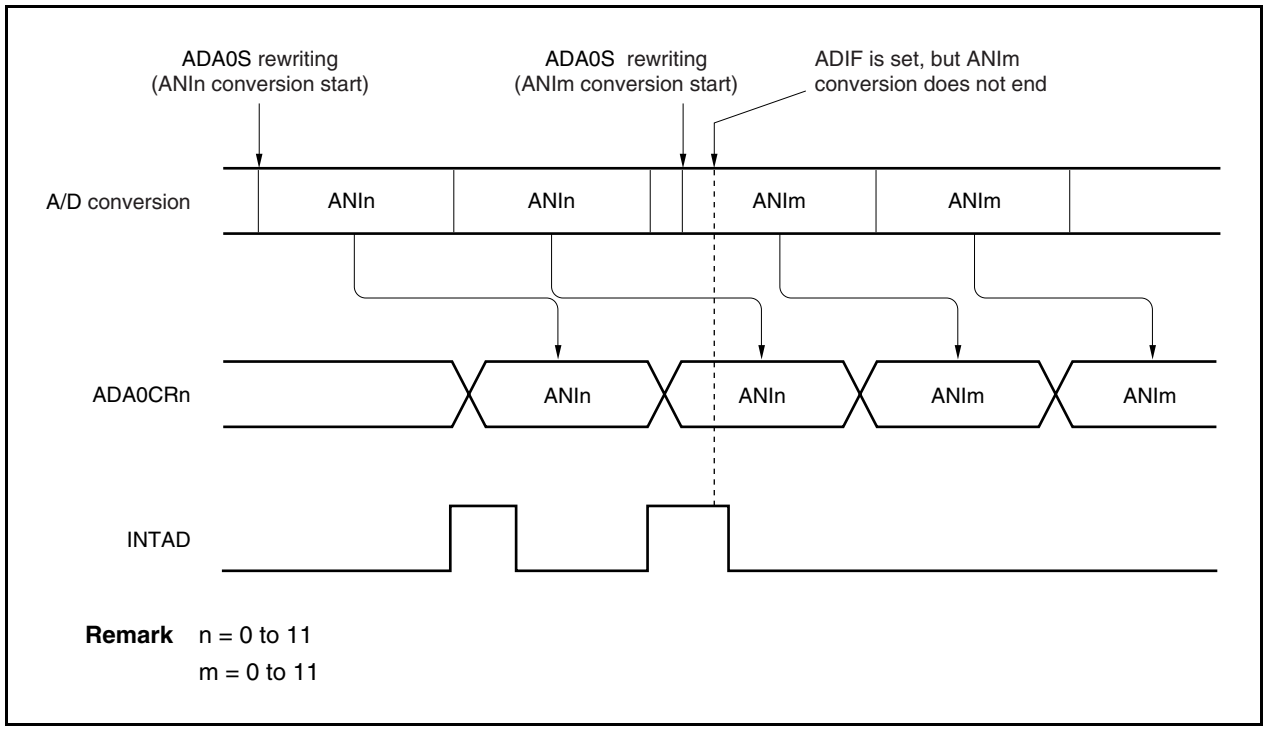
(c) If a current flows through a pin of port 7 (P70 to P711) that is set in the output mode because of the influence of the external circuit connected to the port pin and, as a result, the AV_{REF0} power supply fluctuates (cause: influence on the AV_{REF0} power supply)

If there is a possibility that any of the above processings may be executed during A/D conversion, be sure to execute A/D conversion more than once, check the A/D conversion value, and eliminate any abnormal value by program.

(5) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if the contents of the ADA0S register are changed. If the analog input pin is changed during A/D conversion, therefore, the result of converting the previously selected analog input signal may be stored and the conversion end interrupt request flag may be set immediately before the ADA0S register is rewritten. If the ADIF flag is read immediately after the ADA0S register is rewritten, the ADIF flag may be set even though the A/D conversion of the newly selected analog input pin has not been completed. When A/D conversion is stopped, clear the ADIF flag before resuming conversion.

Figure 11-13. Generation Timing of A/D Conversion End Interrupt Request

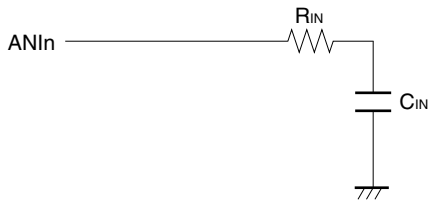


Remark n = 0 to 11
m = 0 to 11

(6) Internal equivalent circuit

The following shows the equivalent circuit of the analog input block.

Figure 11-14. Internal Equivalent Circuit of ANIn Pin



<R>

R_{IN}	C_{IN}
2.9 kΩ	4.0 pF

Remarks 1. The above values are reference values.

2. n = 0 to 11

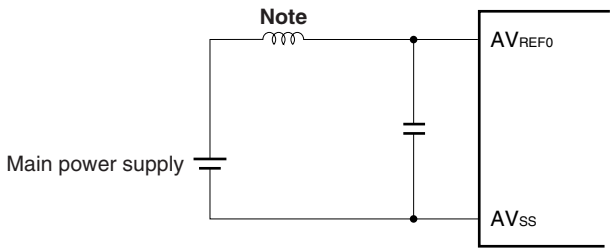
(7) AV_{REF0} pin

(a) The AV_{REF0} pin is used as the power supply pin of the A/D converter and also supplies power to the alternate-function ports. In an application where a backup power supply is used, be sure to supply the same voltage as V_{DD} to the AV_{REF0} pin as shown in Figure 11-15.

(b) The AV_{REF0} pin is also used as the reference voltage pin of the A/D converter. If the source supplying power to the AV_{REF0} pin has a high impedance or if the power supply has a low current supply capability, the reference voltage may fluctuate due to the current that flows during conversion (especially, immediately after the conversion operation enable bit ADA0CE has been set to 1). As a result, the conversion accuracy may drop. To avoid this, it is recommended to connect a capacitor across the AV_{REF0} and AV_{SS} pins to suppress the reference voltage fluctuation as shown in Figure 11-15.

(c) If the source supplying power to the AV_{REF0} pin has a high DC resistance (for example, because of insertion of a diode), the voltage when conversion is enabled may be lower than the voltage when conversion is stopped, because of a voltage drop caused by the A/D conversion current.

Figure 11-15. AV_{REF0} Pin Processing Example



Note Parasitic inductance

<R> **(8) Reading ADA0CRn register**

When the ADA0M0 to ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written, the contents of the ADA0CRn register may be undefined. Read the conversion result after completion of conversion and before writing to the ADA0M0 to ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register. Also, when an external/timer trigger is acknowledged, the contents of the ADA0CRn register may be undefined. Read the conversion result after completion of conversion and before the next external/timer trigger is acknowledged. The correct conversion result may not be read at a timing different from the above.

(9) Standby mode

Because the A/D converter stops operating in the STOP mode, conversion results are invalid, so power consumption can be reduced. Operations are resumed after the STOP mode is released, but the A/D conversion results after the STOP mode is released are invalid. When using the A/D converter after the STOP mode is released, before setting the STOP mode or releasing the STOP mode, clear the ADA0M0.ADA0CE bit to 0 then set the ADA0CE bit to 1 after releasing the STOP mode.

In the IDLE1, IDLE2, or subclock operation mode, operation continues. To lower the power consumption, therefore, clear the ADA0M0.ADA0CE bit to 0. In the IDLE1 and IDLE2 modes, since the analog input voltage value cannot be retained, the A/D conversion results after the IDLE1 and IDLE2 modes are released are invalid. The results of conversions before the IDLE1 and IDLE2 modes were set are valid.

<R> **(10) Restriction for each mode**

(a) When the external trigger mode or timer trigger mode is selected, set the high-speed conversion mode. Do not input a trigger during stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0M0.ADA0CE bit = 1).

(b) Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0PFT register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1).
- Normal conversion mode
- One-shot select mode/one-shot scan mode in high-speed conversion mode

Remark A/D control registers: ADA0M0, ADA0M2, ADA0S, ADA0PFM, and ADA0PFT registers

(11) Variation of A/D conversion results

The results of the A/D conversion may vary depending on the fluctuation of the supply voltage, or may be affected by noise. To reduce the variation, take counteractive measures with the program such as averaging the A/D conversion results.

(12) A/D conversion result hysteresis characteristics

The successive comparison type A/D converter holds the analog input voltage in the internal sample & hold capacitor and then performs A/D conversion. After the A/D conversion has finished, the analog input voltage remains in the internal sample & hold capacitor. As a result, the following phenomena may occur.

- When the same channel is used for A/D conversions, if the voltage is higher or lower than the previous A/D conversion, then hysteresis characteristics may appear where the conversion result is affected by the previous value. Thus, even if the conversion is performed at the same potential, the result may vary.
- When switching the analog input channel, hysteresis characteristics may appear where the conversion result is affected by the previous channel value. This is because one A/D converter is used for the A/D conversions. Thus, even if the conversion is performed at the same potential, the result may vary.

11.7 How to Read A/D Converter Characteristics Table

This section describes the terms related to the A/D converter.

(1) Resolution

The minimum analog input voltage that can be recognized, i.e., the ratio of an analog input voltage to 1 bit of digital output is called 1 LSB (least significant bit). The ratio of 1 LSB to the full scale is expressed as %FSR (full-scale range). %FSR is the ratio of a range of convertible analog input voltages expressed as a percentage, and can be expressed as follows, independently of the resolution.

1%FSR = (Maximum value of convertible analog input voltage – Minimum value of convertible analog input voltage)/100
= (AV_{REF0} – 0)/100
= AV_{REF0}/100

When the resolution is 10 bits, 1 LSB is as follows:

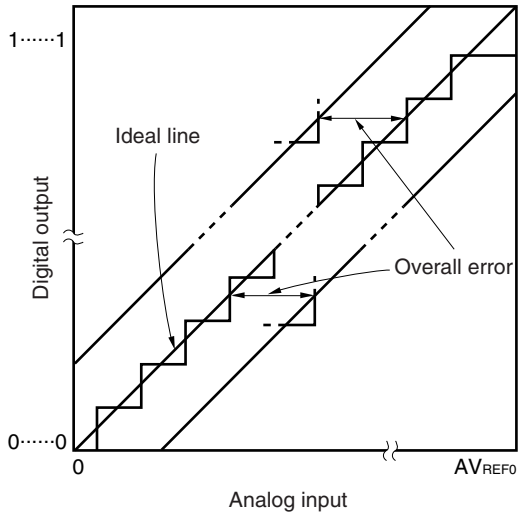
1 LSB = $1/2^{10}$ = 1/1,024
= 0.098%FSR

The accuracy is determined by the overall error, independently of the resolution.

(2) Overall error

This is the maximum value of the difference between an actually measured value and a theoretical value.
It is a total of zero-scale error, full-scale error, linearity error, and a combination of these errors.
The overall error in the characteristics table does not include the quantization error.

Figure 11-16. Overall Error

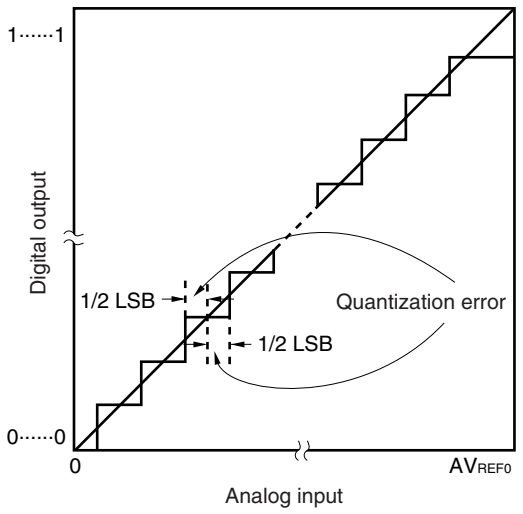


(3) Quantization error

This is an error of ±1/2 LSB that inevitably occurs when an analog value is converted into a digital value. Because the A/D converter converts analog input voltages in a range of ±1/2 LSB into the same digital codes, a quantization error is unavoidable.

This error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, or differential linearity error in the characteristics table.

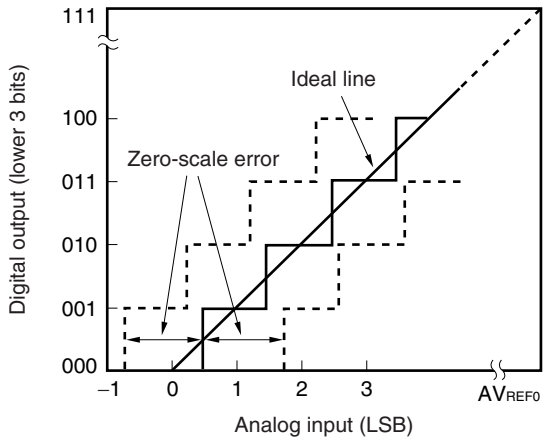
Figure 11-17. Quantization Error



(4) Zero-scale error

This is the difference between the actually measured analog input voltage and its theoretical value when the digital output changes from 0…000 to 0…001 (1/2 LSB).

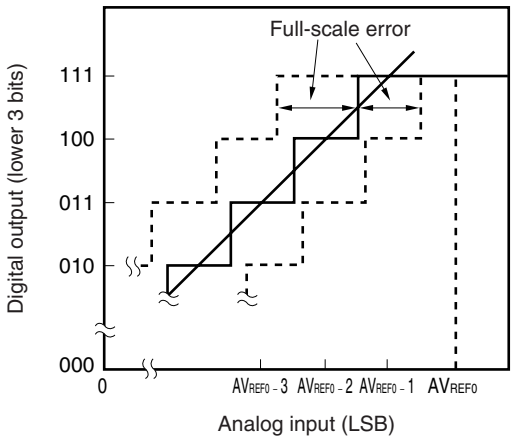
Figure 11-18. Zero-Scale Error



(5) Full-scale error

This is the difference between the actually measured analog input voltage and its theoretical value when the digital output changes from 1…110 to 1…111 (full scale – 3/2 LSB).

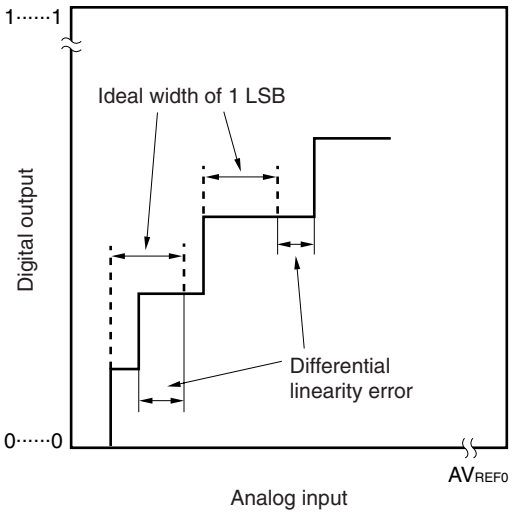
Figure 11-19. Full-Scale Error



(6) Differential linearity error

Ideally, the width to output a specific code is 1 LSB. This error indicates the difference between the actually measured value and its theoretical value when a specific code is output. This indicates the basic characteristics of the A/D conversion when the voltage applied to the analog input pins of the same channel is consistently increased bit by bit from AV_{SS} to AV_{REF0}. When the input voltage is increased or decreased, or when two or more channels are used, refer to **11.7 (2) Overall error**.

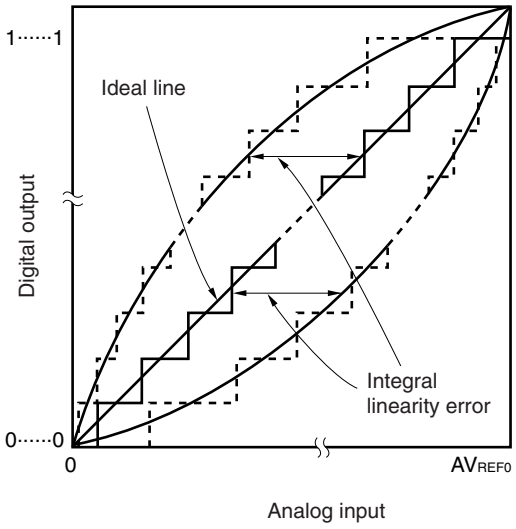
Figure 11-20. Differential Linearity Error



(7) Integral linearity error

This error indicates the extent to which the conversion characteristics differ from the ideal linear relationship. It indicates the maximum value of the difference between the actually measured value and its theoretical value where the zero-scale error and full-scale error are 0.

Figure 11-21. Integral Linearity Error



(8) Conversion time

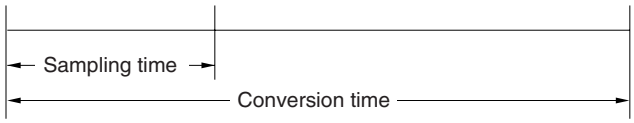
This is the time required to obtain a digital output after each trigger has been generated.

The conversion time in the characteristics table includes the sampling time.

(9) Sampling time

This is the time for which the analog switch is ON to load an analog voltage to the sample & hold circuit.

Figure 11-22. Sampling Time



<R>

CHAPTER 12 D/A CONVERTER

A D/A converter is provided only in the μPD703253Y.

12.1 Functions

The D/A converter has the following functions.

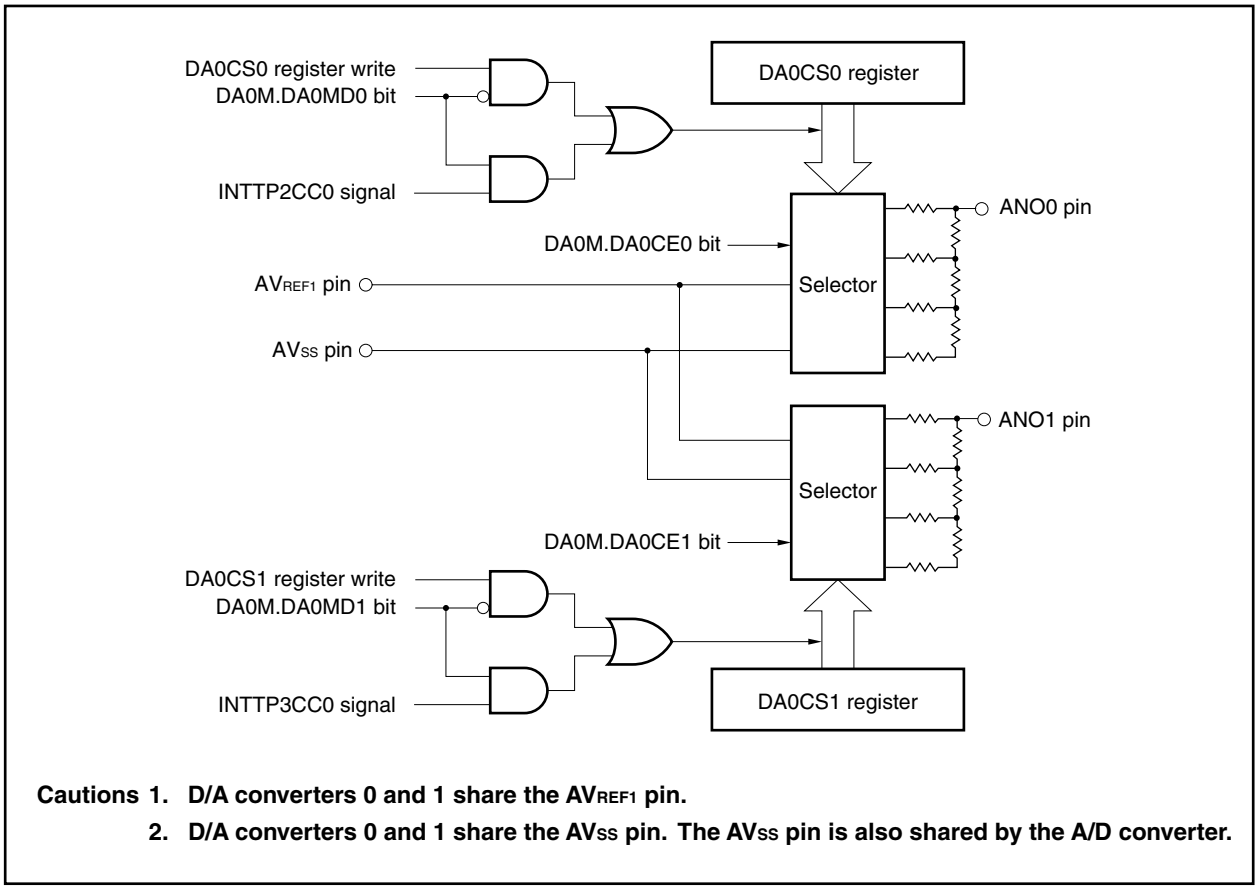
- 8-bit resolution × 2 channels (DA0CS0, DA0CS1)
- R-2R ladder method
- Settling time: 3 μs max. (when AV_{REF1} is 3.0 to 3.6 V and external load is 20 pF)
- Analog output voltage: AV_{REF1} × m/256 (m = 0 to 255; value set to DA0CSn register)
- Operation modes: Normal mode, real-time output mode

Remark n = 0, 1

12.2 Configuration

The D/A converter configuration is shown below.

Figure 12-1. Block Diagram of D/A Converter



Cautions 1. D/A converters 0 and 1 share the AV_{REF1} pin.

2. D/A converters 0 and 1 share the AV_{SS} pin. The AV_{SS} pin is also shared by the A/D converter.

The D/A converter includes the following hardware.

Table 12-1. Configuration of D/A Converter

Item	Configuration
Control registers	D/A converter mode register (DA0M) D/A converter conversion value setting registers 0, 1 (DA0CS0, DA0CS1)

12.3 Registers

The registers that control the D/A converter are as follows.

- D/A converter mode register (DA0M)
- D/A converter conversion value setting registers 0, 1 (DA0CS0, DA0CS1)

(1) D/A converter mode register (DA0M)

The DA0M register controls the operation of the D/A converter.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF282H

	7	6	<5>	<4>	3	2	1	0
DA0M	0	0	DA0CE1	DA0CE0	0	0	DA0MD1	DA0MD0

DA0CEn	Control of D/A converter operation enable/disable (n = 0, 1)
0	Disables operation
1	Enables operation

DA0MDn	Selection of D/A converter operation mode (n = 0, 1)
0	Normal mode
1	Real-time output mode[Note]

Note The output trigger in the real-time output mode (DA0MDn bit = 1) is as follows.

- When n = 0: INTTP2CC0 signal (see **CHAPTER 7 16-BIT TIMER/EVENT COUNTER P (TMP)**)
- When n = 1: INTTP3CC0 signal (see **CHAPTER 7 16-BIT TIMER/EVENT COUNTER P (TMP)**)

(2) D/A converter conversion value setting registers 0, 1 (DA0CS0, DA0CS1)

The DA0CS0 and DA0CS1 registers set the analog voltage value output to the ANO0 and ANO1 pins.

These registers can be read or written in 8-bit units.

Reset sets these registers to 00H.

After reset: 00H R/W Address: DA0CS0 FFFFF280H, DA0CS1 FFFFF281H

	7	6	5	4	3	2	1	0
DA0CSn	DA0CSn7	DA0CSn6	DA0CSn5	DA0CSn4	DA0CSn3	DA0CSn2	DA0CSn1	DA0CSn0

Caution In the real-time output mode (DA0M.DA0MDn bit = 1), set the DA0CSn register before the INTTP2CC0/INTTP3CC0 signals are generated. D/A conversion starts when the INTTP2CC0/INTTP3CC0 signals are generated.

Remark n = 0, 1

12.4 Operation

12.4.1 Operation in normal mode

D/A conversion is performed using a write operation to the DA0CSn register as the trigger.

The setting method is described below.

<1> Set the DA0M.DA0MDn bit to 0 (normal mode).

<2> Set the analog voltage value to be output to the ANOn pin to the DA0CSn register.
Steps <1> and <2> above constitute the initial settings.

<3> Set the DA0M.DA0CEn bit to 1 (D/A conversion enable).
D/A conversion starts when this setting is performed.

<4> To perform subsequent D/A conversions, write to the DA0CSn register.
The previous D/A conversion result is held until the next D/A conversion is performed.

Remarks 1. For the alternate-function pin settings, see **Table 4-15 Using Port Pin as Alternate-Function Pin**.

2. n = 0, 1

12.4.2 Operation in real-time output mode

D/A conversion is performed using the interrupt request signals (INTTP2CC0 and INTTP3CC0) of TMP2 and TMP3 as triggers.

The setting method is described below.

<1> Set the DA0M.DA0MDn bit to 1 (real-time output mode).

<2> Set the analog voltage value to be output to the ANOn pin to the DA0CSn register.

<3> Set the DA0M.DA0CEn bit to 1 (D/A conversion enable).
Steps <1> to <3> above constitute the initial settings.

<4> Operate TMP2 and TMP3.

<5> D/A conversion starts when the INTTP2CC0 and INTTP3CC0 signals are generated.

<6> After that, the value set in DA0CSn register is output every time the INTTP2CC0 and INTTP3CC0 signals are generated.

Remarks 1. The output values of the ANO0 and ANO1 pins up to <5> above are undefined.

2. For the output values of the ANO0 and ANO1 pins in the HALT, IDLE1, IDLE2, and STOP modes, see **CHAPTER 21 STANDBY FUNCTION**.

3. For the alternate-function pin settings, see **Table 4-15 Using Port Pin as Alternate-Function Pin**.

12.4.3 Cautions

Observe the following cautions when using the D/A converter of the μPD703253Y.

(1) Do not change the set value of the DA0CSn register while the trigger signal is being issued in the real-time output mode.

(2) Before changing the operation mode, be sure to clear the DA0M.DA0CEn bit to 0.

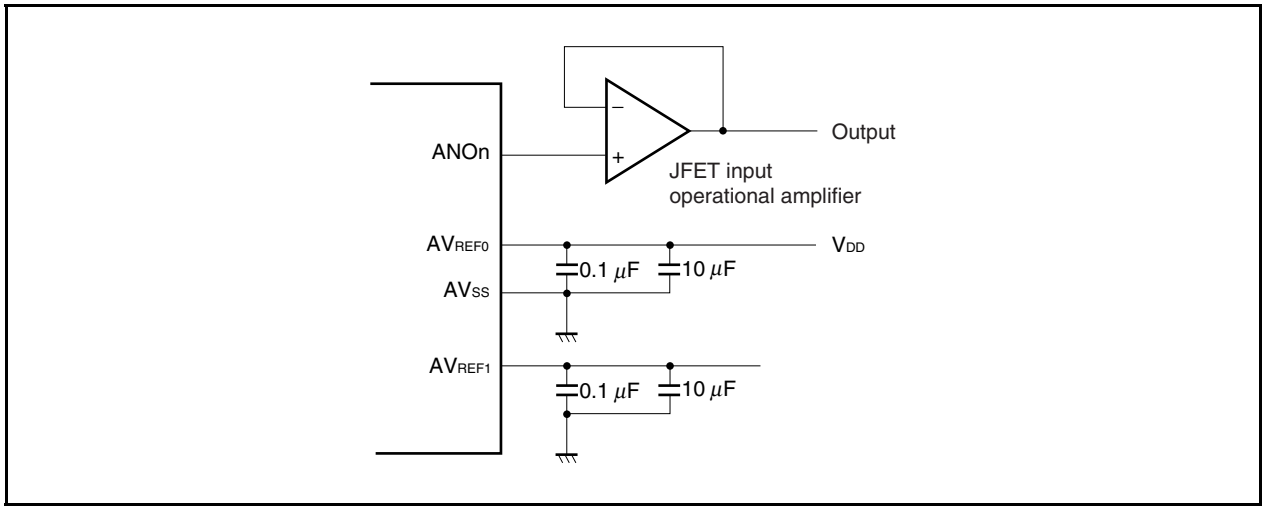
(3) When using one of the P10/AN00 and P11/AN01 pins as an I/O port and the other as a D/A output pin, do so in an application where the port I/O level does not change during D/A output.

(4) Make sure that $AV_{REF0} = V_{DD} = AV_{REF1}$ = 3.0 to 3.6 V. If this range is exceeded, the operation is not guaranteed.

(5) Apply power to AV_{REF1} at the same timing as AV_{REF0}.

(6) No current can be output from the ANOn pin (n = 0, 1) because the output impedance of the D/A converter is high. When connecting a resistor of 2 MΩ or less, insert a JFET input operational amplifier between the resistor and the ANOn pin.

Figure 12-2. External Pin Connection Example



(7) Because the D/A converter stops operation in the STOP mode, the ANO0 and ANO1 pins go into a high-impedance state, and the power consumption can be reduced.
In the IDLE1, IDLE2, or subclock operation mode, however, the operation continues. To lower the power consumption, therefore, clear the DA0M.DA0CEn bit to 0.

CHAPTER 13 ASYNCHRONOUS SERIAL INTERFACE A (UARTA)

13.1 Features

- Transfer rate: 300 bps to 312.5 kbps (using internal system clock of 20 MHz and dedicated baud rate generator)
- Full-duplex communication: Internal UARTAn receive data register (UAnRX)
  Internal UARTAn transmit data register (UAnTX)
- 2-pin configuration: TXDAn: Transmit data output pin
  RXDAn: Receive data input pin
- Reception error output function
  - Parity error
  - Framing error
  - Overrun error
- Interrupt sources: 2
  - Reception complete interrupt (INTUAnR): This interrupt occurs upon transfer of receive data from the receive shift register to UAnRX register after serial transfer completion, in the reception enabled status.
  - Transmission enable interrupt (INTUAnT): This interrupt occurs upon transfer of transmit data from the UAnTX register to the transmit shift register in the transmission enabled status.
- Character length: 7, 8 bits
- Parity function: Odd, even, 0, none
- Transmission stop bit: 1, 2 bits
- On-chip dedicated baud rate generator
- MSB-/LSB-first transfer selectable
- Transmit/receive data inverted input/output possible

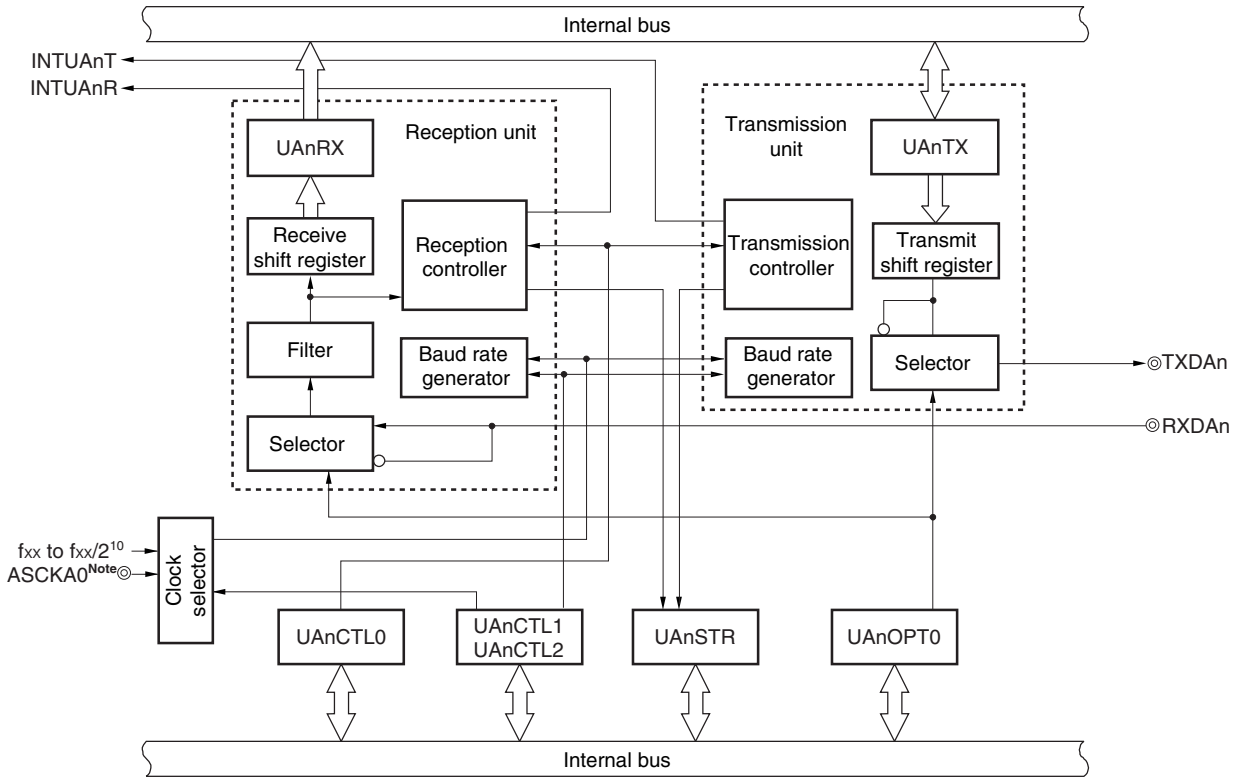
Caution The LIN (Local Interconnect Network) function is not supported in UARTA of the V850ES/SG1.

Remark n = 0, 1

13.2 Configuration

The block diagram of the UARTAn is shown below.

<R> **Figure 13-1. Block Diagram of Asynchronous Serial Interface An**



Note UARTA0 only

Remarks 1. n = 0, 1

2. For the configuration of the baud rate generator, see **Figure 13-9**.

UARTAn includes the following hardware.

Table 13-1. Configuration of UARTAn

Item	Configuration
Registers	UARTAn control register 0 (UAnCTL0) UARTAn control register 1 (UAnCTL1) UARTAn control register 2 (UAnCTL2) UARTAn option control register 0 (UAnOPT0) UARTAn status register (UAnSTR) UARTAn receive shift register UARTAn receive data register (UAnRX) UARTAn transmit shift register UARTAn transmit data register (UAnTX)

(1) UARTAn control register 0 (UAnCTL0)

The UAnCTL0 register is an 8-bit register used to specify the UARTAn operation.

(2) UARTAn control register 1 (UAnCTL1)

The UAnCTL1 register is an 8-bit register used to select the input clock for the UARTAn.

(3) UARTAn control register 2 (UAnCTL2)

The UAnCTL2 register is an 8-bit register used to control the baud rate for the UARTAn.

(4) UARTAn option control register 0 (UAnOPT0)

The UAnOPT0 register is an 8-bit register used to control serial transfer for the UARTAn.

(5) UARTAn status register (UAnSTR)

The UAnSTRn register consists of flags indicating the error contents when a reception error occurs. Each one of the reception error flags is set (to 1) upon occurrence of a reception error.

(6) UARTAn receive shift register

This is a shift register used to convert the serial data input to the RXDAn pin into parallel data. Upon reception of 1 byte of data and detection of the stop bit, the receive data is transferred to the UAnRX register.

This register cannot be manipulated directly.

(7) UARTAn receive data register (UAnRX)

The UAnRX register is an 8-bit register that holds receive data. When 7 characters are received, 0 is stored in the highest bit (when data is received LSB first).

In the reception enabled status, receive data is transferred from the UARTAn receive shift register to the UAnRX register in synchronization with the completion of shift-in processing of 1 frame.

Transfer to the UAnRX register also causes the reception complete interrupt request signal (INTUAnR) to be output.

(8) UARTAn transmit shift register

The transmit shift register is a shift register used to convert the parallel data transferred from the UAnTX register into serial data.

When 1 byte of data is transferred from the UAnTX register, the shift register data is output from the TXDAn pin.

This register cannot be manipulated directly.

(9) UARTAn transmit data register (UAnTX)

The UAnTX register is an 8-bit transmit data buffer. Transmission starts when transmit data is written to the UAnTX register. When data can be written to the UAnTX register (when data of one frame is transferred from the UAnTX register to the UARTAn transmit shift register), the transmission enable interrupt request signal (INTUAnT) is generated.

13.3 Registers

(1) UARTAn control register 0 (UAnCTL0)

The UAnCTL0 register is an 8-bit register that controls the UARTAn serial transfer operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 10H.

(1/2)

After reset: 10H R/W Address: UA0CTL0 FFFFFA00H, UA1CTL0 FFFFFA10H

UAnCTL0 (n = 0, 1)

<7>	<6>	<5>	<4>	3	2	1	0
UAnPWR	UAnTXE	UAnRXE	UAnDIR	UAnPS1	UAnPS0	UAnCL	UAnSL

UAnPWR	UARTAn operation control
0	Disable UARTAn operation (UARTAn reset asynchronously)
1	Enable UARTAn operation

The UARTAn operation is controlled by the UAnPWR bit. The TXDAn pin output is fixed to high level by clearing the UAnPWR bit to 0 (fixed to low level if UAnOPT0.UAnTDL bit = 1).

UAnTXE	Transmission operation enable
0	Disable transmission operation
1	Enable transmission operation

<R> • To start transmission, set the UAnPWR bit to 1 and then set the UAnTXE bit to 1.

• To initialize the transmission unit, clear the UAnTXE bit to 0, wait for two cycles of the base clock (f_{UCLK}), and then set the UAnTXE bit to 1 again. Otherwise, initialization may not be executed (for the base clock, see **13.6 (1) (a) Base clock**).

<R> • When the operation is enabled (UAnPWR bit = 1), the transmission operation is enabled after two or more cycles of the base clock (f_{UCLK}) have elapsed since UAnTXE = 1.

<R> • When the UAnPWR bit is cleared to 0, the status of the internal circuit becomes the same status as UAnTXE bit = 0 by the UAnPWR bit even if the UAnTXE bit is 1. The transmission operation is enabled when the UAnPWR bit is set to 1 again.

(2/2)

UAnRXE	Reception operation enable
0	Disable reception operation
1	Enable reception operation

<R> • To start reception, set the UAnPWR bit to 1 and then set the UAnRXE bit to 1.
• To initialize the reception unit, clear the UAnRXE bit to 0, wait for two cycles of the base clock, and then set the UAnRXE bit to 1 again. Otherwise, initialization may not be executed (for the base clock, see **13.6 (1) (a) Base clock**).
<R> • When the operation is enabled (UAnPWR bit = 1), the reception operation is enabled after two or more cycles of the base clock (f_{UCLK}) have elapsed since UAnRXE = 1. If the start bit is received before the reception operation is enabled, the start bit is ignored.
<R> • When the UAnPWR bit is cleared to 0, the status of the internal circuit becomes the same status as UAnRXE bit = 0 by the UAnPWR bit even if the UAnRXE bit is 1. The reception operation is enabled when the UAnPWR bit is set to 1 again.

<R>

UAnDIR[Note]	Transfer direction selection
0	MSB-first transfer
1	LSB-first transfer

<R>

UAnPS1[Note]	UAnPS0[Note]	Parity selection during transmission	Parity selection during reception
0	0	No parity output	Reception with no parity
0	1	0 parity output	Reception with 0 parity
1	0	Odd parity output	Odd parity check
1	1	Even parity output	Even parity check

If “Reception with 0 parity” is selected during reception, a parity check is not performed. Therefore, the UAnSTR.UAnPE bit is not set.

<R>

UAnCL[Note]	Specification of data character length of 1 frame of transmit/receive data
0	7 bits
1	8 bits

<R>

UAnSL[Note]	Specification of length of stop bit for transmit data
0	1 bit
1	2 bits

<R> Only the first bit of the receive data stop bits is checked, regardless of the value of the UAnSL bit.

<R> **Note** This register can be rewritten only when the UAnPWR bit = 0 or the UAnTXE bit = UAnRXE bit = 0. However, setting any or all of the UAnPWR, UAnTXE, and UAnRXE bits to 1 at the same time is possible.

Remark For details of parity, see **13.5.6 Parity types and operations**.

(2) UARTAn control register 1 (UAnCTL1)

For details, see **13.6 (2) UARTAn control register 1 (UAnCTL1)**.

(3) UARTAn control register 2 (UAnCTL2)

For details, see **13.6 (3) UARTAn control register 2 (UAnCTL2)**.

(4) UARTAn option control register 0 (UAnOPT0)

The UAnOPT0 register is an 8-bit register that controls the serial transfer operation of the UARTAn register.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 14H.

After reset: 14H R/W Address: UA0OPT0 FFFFFA03H, UA1OPT0 FFFFFA13H

	7	6	5	4	3	2	1	0
UAnOPT0	0	0	0	1	0	1	UAnTDL	UAnRDL

(n = 0, 1)

UAnTDL	Transmit data level bit
0	Normal output of transfer data
1	Inverted output of transfer data

- The output level of the TXDAn pin can be inverted using the UAnTDL bit.
- This register can be set when the UAnPWR bit = 0 or when the UAnTXE bit = 0.

UAnRDL	Receive data level bit
0	Normal input of transfer data
1	Inverted input of transfer data

<R>
- The input level of the RXDAn pin can be inverted using the UAnRDL bit.
- This register can be set when the UAnPWR bit = 0 or the UAnRXE bit = 0.
- When the UAnRDL bit is set to 1 (inverted input of receive data), reception must be enabled (UAnCTL0.UAnRXE bit = 1) after setting the data reception pin to the UART reception pin (RXDAn) when reception is started. When the pin mode is changed after reception is enabled, the start bit will be mistakenly detected if the pin level is high.

Caution Be sure to clear bits 7 to 5 and 3 to "0", and set bits 4 and 2 to "1".

(5) UARTAn status register (UAnSTR)

The UAnSTR register is an 8-bit register that displays the UARTAn transfer status and reception error contents. This register can be read or written in 8-bit or 1-bit units, but the UAnTSF bit is a read-only bit, while the UAnPE, UAnFE, and UAnOVE bits can both be read and written. However, these bits can only be cleared by writing 0; they cannot be set by writing 1 (even if 1 is written to them, the value is retained).

The initialization conditions are shown below.

Register/Bit	Initialization Conditions
UAnSTR register	• Reset • UAnCTL0.UAnPWR = 0
UAnTSF bit	• UAnCTL0.UAnTXE = 0
UAnPE, UAnFE, UAnOVE bits	• 0 write • UAnCTL0.UAnRXE = 0

<R> **Caution Be sure to read the error flags of the UAnPE, UAnFE, and UAnOVE bits to check the flag status, and then clear the flags by writing "0" to them.**

After reset: 00H R/W Address: UA0STR FFFFFA04H, UA1STR FFFFFA14H

	<7>	6	5	4	3	<2>	<1>	<0>
UAnSTR	UAnTSF	0	0	0	0	UAnPE	UAnFE	UAnOVE

(n = 0, 1)

UAnTSF	Transfer status flag
0	• When the UAnPWR bit = 0 or the UAnTXE bit = 0 has been set. • When, following transfer completion, there was no next data transfer from UAnTX register
1	Write to UAnTX register
The UAnTSF bit is always 1 when performing continuous transmission. When initializing the transmission unit, check that the UAnTSF bit = 0 before performing initialization. The transmit data is not guaranteed when initialization is performed while the UAnTSF bit = 1.	

UAnPE	Parity error flag
0	• When the UAnPWR bit = 0 or the UAnRXE bit = 0 has been set. • When 0 has been written
1	When parity of data and parity bit do not match during reception.
• The operation of the UAnPE bit is controlled by the settings of the UAnCTL0.UAnPS1 and UAnCTL0.UAnPS0 bits. • The UAnPE bit can be read and written, but it can only be cleared by writing 0 to it, and it cannot be set by writing 1 to it. When 1 is written to this bit, the value is retained.	

UAnFE	Framing error flag
0	• When the UAnPWR bit = 0 or the UAnRXE bit = 0 has been set • When 0 has been written
1	When no stop bit is detected during reception
• Only the first bit of the receive data stop bits is checked, regardless of the value of the UAnCTL0.UAnSL bit. • The UAnFE bit can be both read and written, but it can only be cleared by writing 0 to it, and it cannot be set by writing 1 to it. When 1 is written to this bit, the value is retained.	

UAnOVE	Overrun error flag
0	• When the UAnPWR bit = 0 or the UAnRXE bit = 0 has been set. • When 0 has been written
1	When receive data has been set to the UAnRX register and the next receive operation is completed before that receive data has been read
• When an overrun error occurs, the data is discarded without the next receive data being written to UAnRx register. • The UAnOVE bit can be both read and written, but it can only be cleared by writing 0 to it. When 1 is written to this bit, the value is retained.	

(6) UARTAn receive data register (UAnRX)

The UAnRX register is an 8-bit buffer register that stores parallel data converted by the receive shift register.

The data stored in the receive shift register is transferred to the UAnRX register upon completion of reception <R> of 1 byte of data. The reception end interrupt request signal (INTUAnR) is generated in this timing.

During LSB-first reception when the data length has been specified as 7 bits, the receive data is transferred to bits 6 to 0 of the UAnRX register and the MSB always becomes 0. During MSB-first reception, the receive data is transferred to bits 7 to 1 of the UAnRX register and the LSB always becomes 0.

When an overrun error occurs (UAnSTR.UAnOVE bit = 1), the receive data at this time is not transferred to the UAnRX register and is discarded.

This register is read-only, in 8-bit units.

In addition to reset input, the UAnRX register can be set to FFH by clearing the UAnCTL0.UAnPWR bit to 0.

After reset: FFH R Address: UA0RX FFFFFA06H, UA1RX FFFFFA16H

	7	6	5	4	3	2	1	0
UAnRX								

(n = 0, 1)

(7) UARTAn transmit data register (UAnTX)

The UAnTX register is an 8-bit register used to set transmit data.

<R> Transmission starts when transmit data is written to the UAnTX register in the transmission enabled status (UAnCTL0.UAnTXE bit = 1). When the data of the UAnTX register has been transferred to the transmit shift register, the transmission enable interrupt request signal (INTUAnT) is generated.

This register can be read or written in 8-bit units.

Reset sets this register to FFH.

After reset: FFH R/W Address: UA0TX FFFFFA07H, UA1TX FFFFFA17H

	7	6	5	4	3	2	1	0
UAnTX								

(n = 0, 1)

13.4 Interrupt Request Signals

The following two interrupt request signals are generated from UARTAn.

- Reception complete interrupt request signal (INTUAnR)
- Transmission enable interrupt request signal (INTUAnT)

The default priority for these two interrupt request signals is reception complete interrupt request signal then transmission enable interrupt request signal.

Table 13-2. Interrupts and Their Default Priorities

Interrupt	Priority
Reception complete	High
Transmission enable	Low

(1) Reception complete interrupt request signal (INTUAnR)

A reception complete interrupt request signal is output when data is shifted into the receive shift register and transferred to the UAnRX register in the reception enabled status.

When a reception complete interrupt request signal is received and the data is read, read the UAnSTR register and check that the reception result is not an error.

No reception complete interrupt request signal is generated in the reception disabled status.

(2) Transmission enable interrupt request signal (INTUAnT)

If transmit data is transferred from the UAnTX register to the UARTAn transmit shift register with transmission enabled, the transmission enable interrupt request signal is generated.

13.5 Operation

13.5.1 Data format

Full-duplex serial data reception and transmission is performed.

As shown in Figure 13-2, one data frame of transmit/receive data consists of a start bit, character bits, parity bit, and stop bit(s).

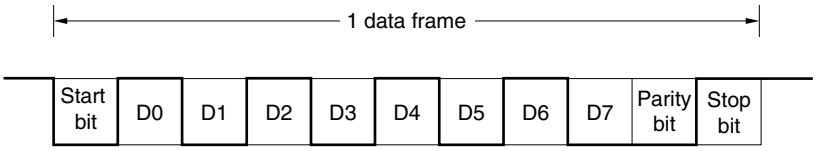
Specification of the character bit length within 1 data frame, parity selection, specification of the stop bit length, and specification of MSB/LSB-first transfer are performed using the UAnCTL0 register.

Moreover, control of UART output/inverted output for the TXDAn bit is performed using the UAnOPT0.UAnTDL bit.

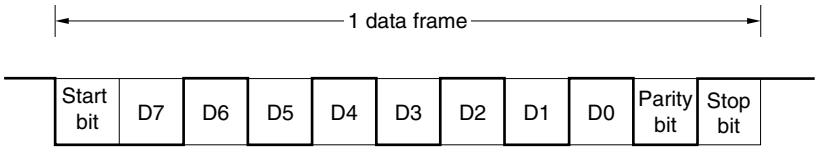
- Start bit ..... 1 bit
- Character bits..... 7 bits/8 bits
- Parity bit ..... Even parity/odd parity/0 parity/no parity
- Stop bit..... 1 bit/2 bits

Figure 13-2. UARTA Transmit/Receive Data Format

(a) 8-bit data length, LSB first, even parity, 1 stop bit, transfer data: 55H

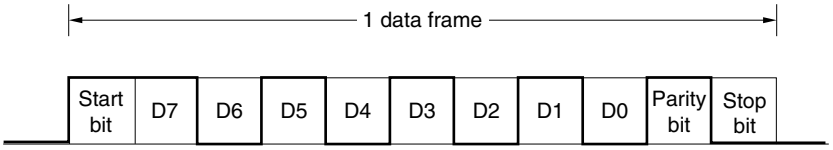


(b) 8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55H

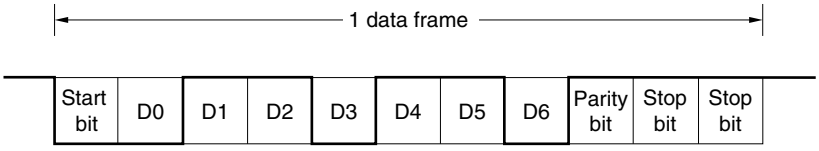


(c) 8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55H, TXDAn inversion

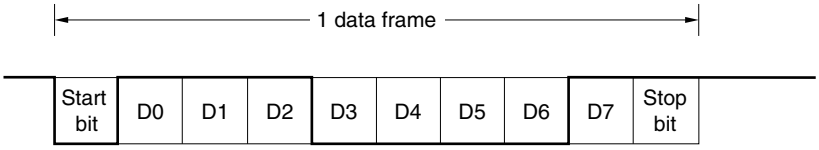
<R>



(d) 7-bit data length, LSB first, odd parity, 2 stop bits, transfer data: 36H



(e) 8-bit data length, LSB first, no parity, 1 stop bit, transfer data: 87H



13.5.2 UART transmission

A high level is output to the TXDAn pin by setting the UAnCTL0.UAnPWR bit to 1.

Next, the transmission enabled status is set by setting the UAnCTL0.UAnTXE bit to 1, and transmission is started by writing transmit data to the UAnTX register. The start bit, parity bit, and stop bit are automatically added.

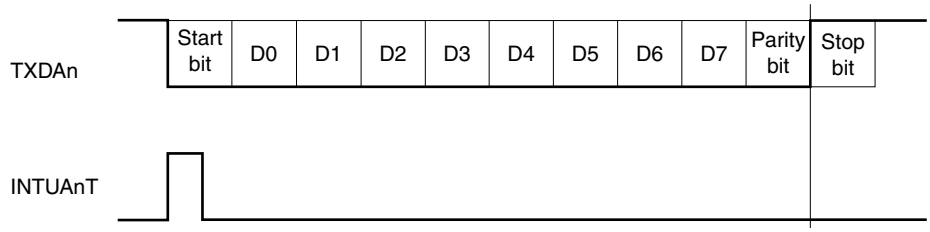
Since the CTS (transmit enable signal) input pin is not provided in UARTAn, use a port to check that reception is enabled at the transmit destination.

The data in the UAnTX register is transferred to the UARTAn transmit shift register upon the start of the transmit operation.

A transmission enable interrupt request signal (INTUAnT) is generated upon completion of transmission of the data of the UAnTX register to the UARTAn transmit shift register, and thereafter the contents of the UARTAn transmit shift register are output to the TXDAn pin.

Write of the next transmit data to the UAnTX register is enabled after the INTUAnT signal is generated.

Figure 13-3. UART Transmission



Remark LSB first

13.5.3 Continuous transmission procedure

UARTAn can write the next transmit data to the UAnTX register when the UARTAn transmit shift register starts the shift operation. The transmit timing of the UARTAn transmit shift register can be judged from the transmission enable interrupt request signal (INTUAnT).

An efficient communication rate is realized by writing the data to be transmitted next to the UAnTX register during transfer.

Caution When initializing transmissions during the execution of continuous transmissions, make sure that the UAnSTR.UAnTSF bit is 0, then perform the initialization. Transmit data that is initialized when the UAnTSF bit is 1 cannot be guaranteed.

Figure 13-4. Continuous Transmission Processing Flow

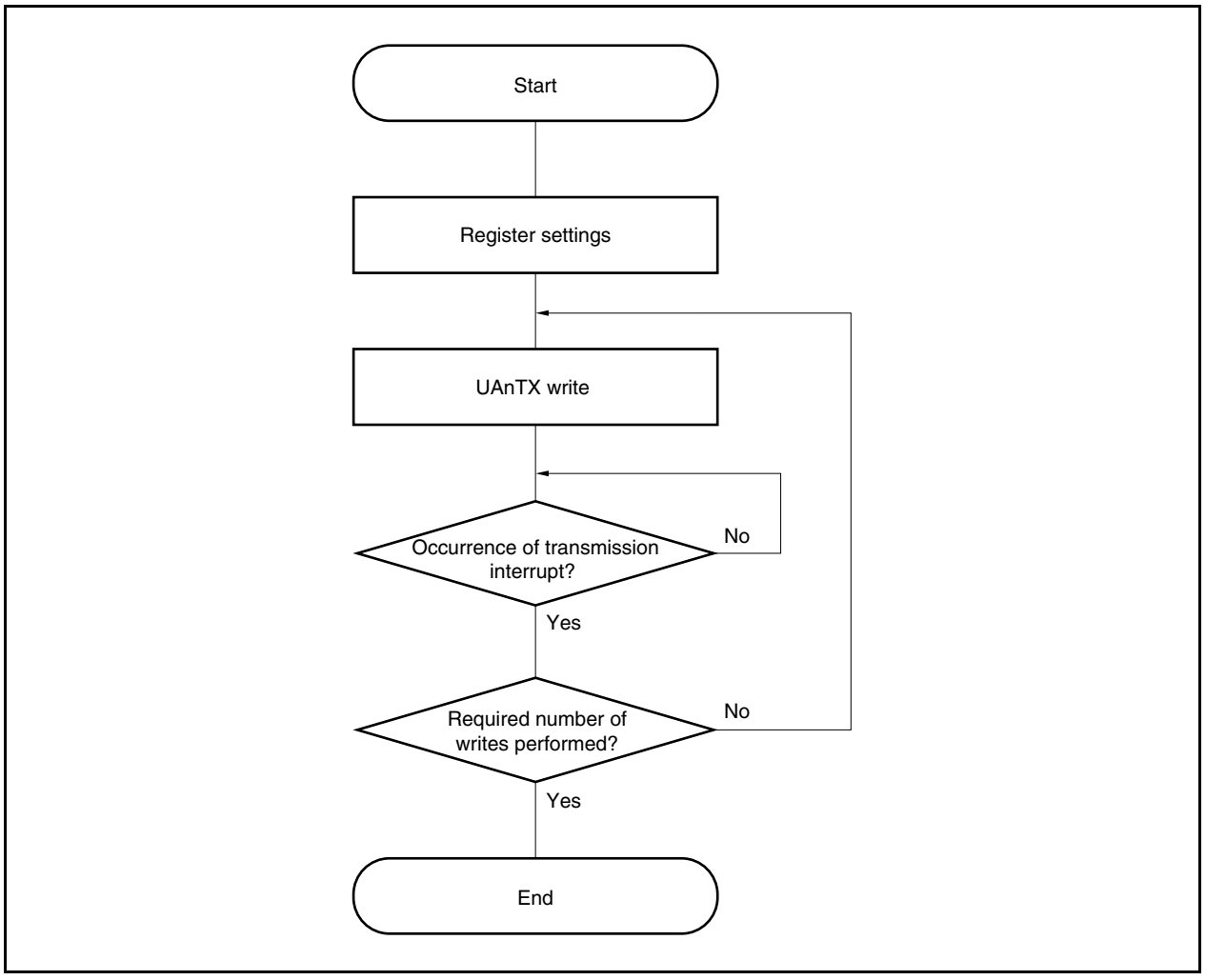
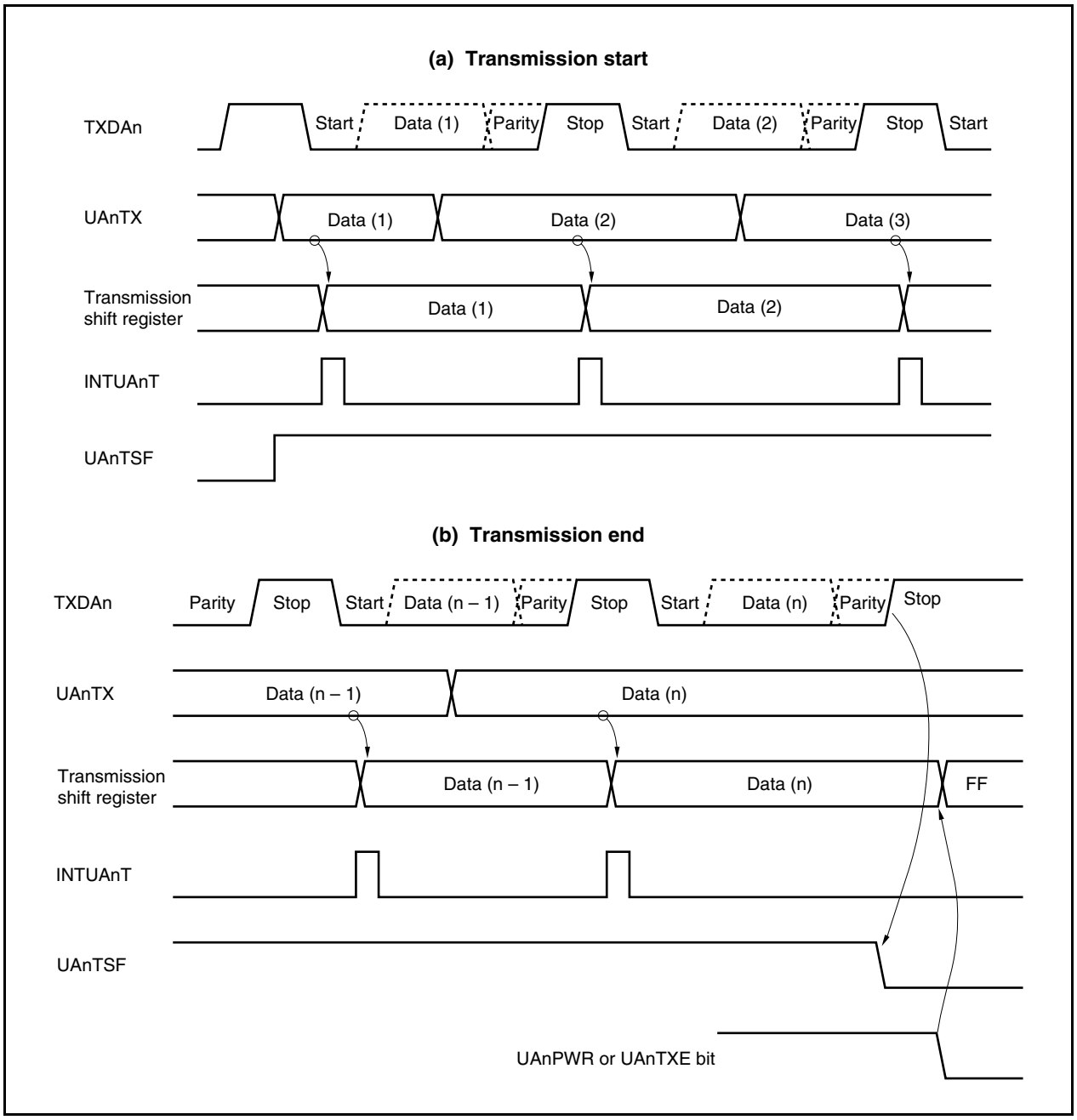


Figure 13-5. Continuous Transmission Operation Timing



13.5.4 UART reception

The reception wait status is set by setting the UAnCTL0.UAnPWR bit to 1 and then setting the UAnCTL0.UAnRXE bit to 1. In the reception wait status, the RXDAn pin is monitored and start bit detection is performed.

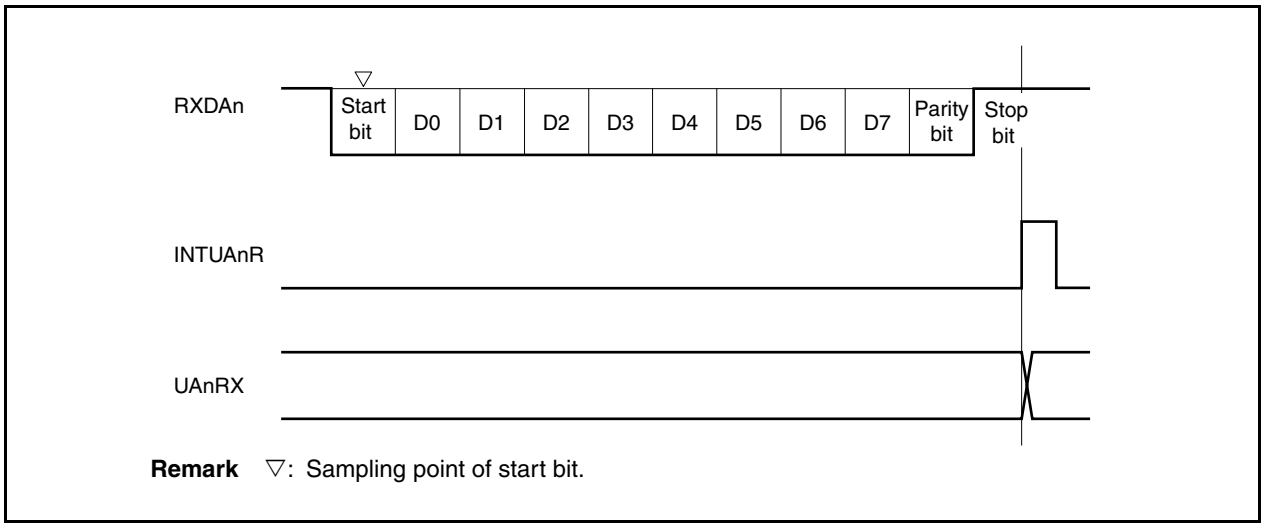
Start bit detection is performed using a two-step detection routine.

First the rising edge of the RXDAn pin is detected and sampling is started at the falling edge. The start bit is recognized if the RXDAn pin is low level at the start bit sampling point. After a start bit has been recognized, the receive operation starts, and serial data is saved to the UARTAn receive shift register according to the set baud rate.

When the reception complete interrupt request signal (INTUAnR) is output upon reception of the stop bit, the data of the UARTAn receive shift register is written to the UAnRX register. However, if an overrun error occurs (UAnSTR.UAnOVE bit = 1), the receive data at this time is not written to the UAnRX register and is discarded.

Even if a parity error (UAnSTR.UAnPE bit) or a framing error (UAnSTR.UAnFE bit = 1) occurs during reception, reception continues until the reception position of the first stop bit, and INTUAnR is output following reception completion.

Figure 13-6. UART Reception



Remark ▽: Sampling point of start bit.

Cautions 1. Be sure to read the UAnRX register even when a reception error occurs. If the UAnRX register is not read, an overrun error occurs during reception of the next data, and reception errors continue occurring indefinitely.

2. The operation during reception is performed assuming that there is only one stop bit. A second stop bit is ignored.

3. When reception is completed, read the UAnRX register after the reception complete interrupt request signal (INTUAnR) has been generated, and clear the UAnPWR or UAnRXE bit to 0. If the UAnPWR or UAnRXE bit is cleared to 0 before the INTUAnR signal is generated, the read value of the UAnRX register cannot be guaranteed.

4. If receive completion processing (INTUAnR signal generation) of UARTAn and the UAnPWR bit = 0 or UAnRXE bit = 0 conflict, the INTUAnR signal may be generated in spite of these being no data stored in the UAnRX register.

To complete reception without waiting INTUAnR signal generation, be sure to clear (0) the interrupt request flag (UAnRIF) of the UAnRIC register, after setting (1) the interrupt mask flag (UAnRMK) of the interrupt control register (UAnRIC) and then set (1) the UAnPWR bit = 0 or UAnRXE bit = 0.

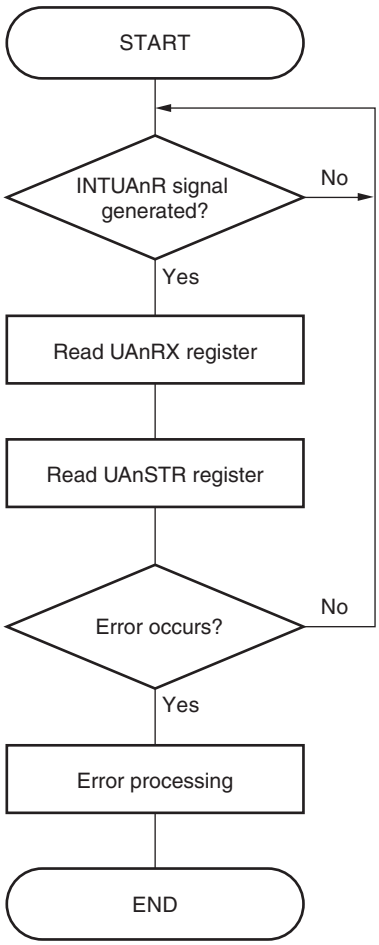
13.5.5 Reception errors

Errors during a receive operation are of three types: parity errors, framing errors, and overrun errors. Data reception result error flags are set in the UAnSTR register and a reception complete interrupt request signal (INTUAnR) is output when an error occurs.

It is possible to ascertain which error occurred during reception by reading the contents of the UAnSTR register.

Clear the reception error flag by writing 0 to it after reading it.

- Receive data read flow



Caution When an INTUAnR signal is generated, the UAnSTR register must be read to check for errors.

- Reception error causes

Error Flag	Reception Error	Cause
UAnPE	Parity error	Received parity bit does not match the setting
UAnFE	Framing error	Stop bit not detected
UAnOVE	Overrun error	Reception of next data completed before data was read from UAnRX register

When reception errors occur, perform the following procedures depending upon the kind of error.

- Parity error
  If false data is received due to problems such as noise in the reception line, discard the received data and retransmit.

- Framing error
  A baud rate error may have occurred between the reception side and transmission side or the start bit may have been erroneously detected. Since this is a fatal error for the communication format, check the operation stop in the transmission side, perform initialization processing each other, and then start the communication again.

- Overrun error
  Since the next reception is completed before reading receive data, 1 frame of data is discarded. If this data was needed, do a retransmission.

Caution If a receive error interrupt occurs during continuous reception, read the contents of the UAnSTR register must be read before the next reception is completed, then perform error processing.

13.5.6 Parity types and operations

The parity bit is used to detect bit errors in the communication data. Normally the same parity is used on the transmission side and the reception side.

In the case of even parity and odd parity, it is possible to detect odd-count bit errors. In the case of 0 parity and no parity, errors cannot be detected.

(a) Even parity

(i) During transmission

The number of bits whose value is “1” among the transmit data, including the parity bit, is controlled so as to be an even number. The parity bit values are as follows.

- Odd number of bits whose value is “1” among transmit data: 1
- Even number of bits whose value is “1” among transmit data: 0

(ii) During reception

The number of bits whose value is “1” among the reception data, including the parity bit, is counted, and if it is an odd number, a parity error is output.

(b) Odd parity

(i) During transmission

Opposite to even parity, the number of bits whose value is “1” among the transmit data, including the parity bit, is controlled so that it is an odd number. The parity bit values are as follows.

- Odd number of bits whose value is “1” among transmit data: 0
- Even number of bits whose value is “1” among transmit data: 1

(ii) During reception

The number of bits whose value is “1” among the receive data, including the parity bit, is counted, and if it is an even number, a parity error is output.

(c) 0 parity

During transmission, the parity bit is always made 0, regardless of the transmit data.

During reception, parity bit check is not performed. Therefore, no parity error occurs, regardless of whether the parity bit is 0 or 1.

(d) No parity

No parity bit is added to the transmit data.

Reception is performed assuming that there is no parity bit. No parity error occurs since there is no parity bit.

13.5.7 Receive data noise filter

This filter samples the RXDAn pin using the base clock (f_{UCLK}) of the prescaler output.

When the same sampling value is read twice, the match detector output changes and the RXDAn signal is sampled as the input data. Therefore, data not exceeding 2 clock width is judged to be noise and is not delivered to the internal circuit (see **Figure 13-8**). See **13.6 (1) (a) Base clock** regarding the base clock.

Moreover, since the circuit is as shown in Figure 13-7, the processing that goes on within the receive operation is delayed by 3 clocks in relation to the external signal status.

Figure 13-7. Noise Filter Circuit

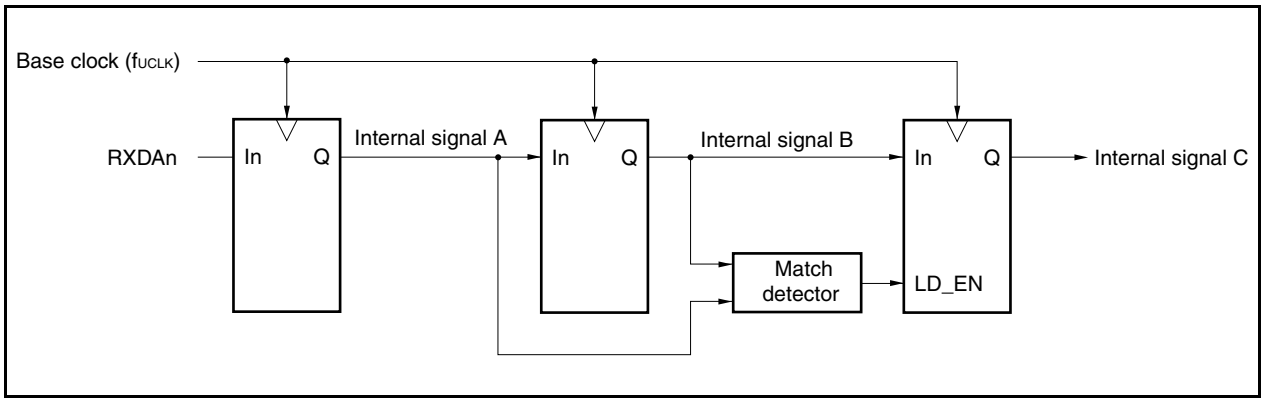
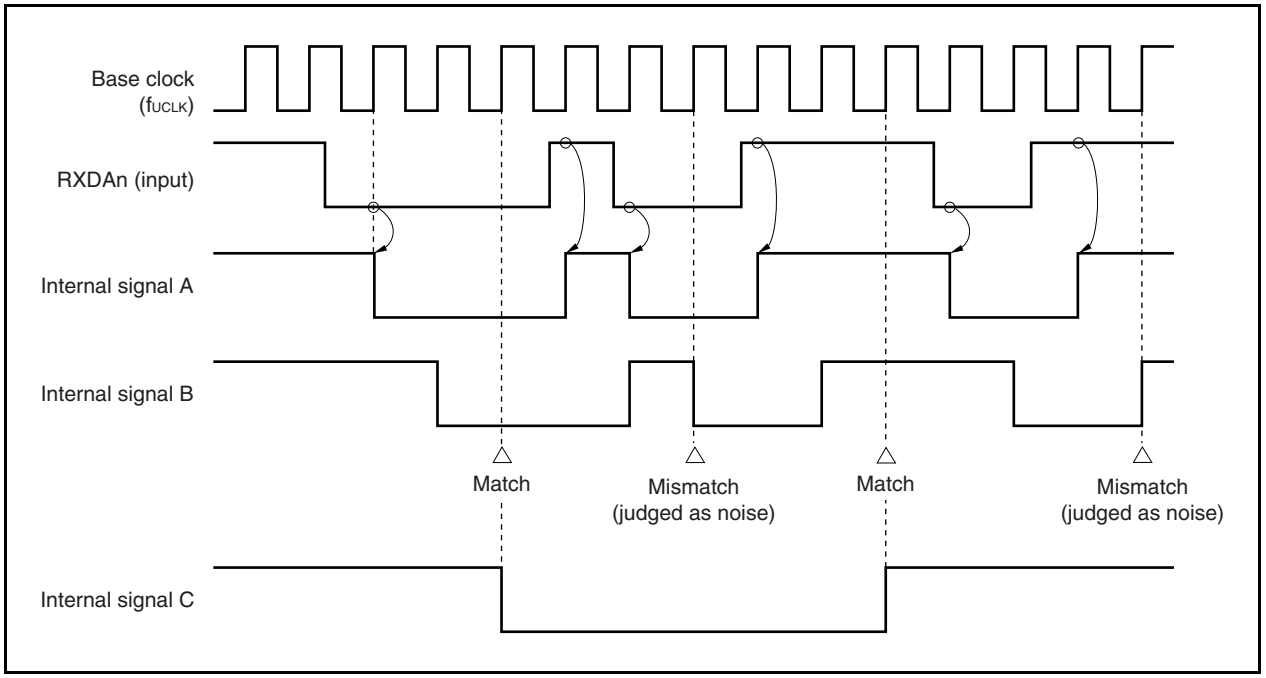


Figure 13-8. Timing of RXDAn Signal Judged as Noise



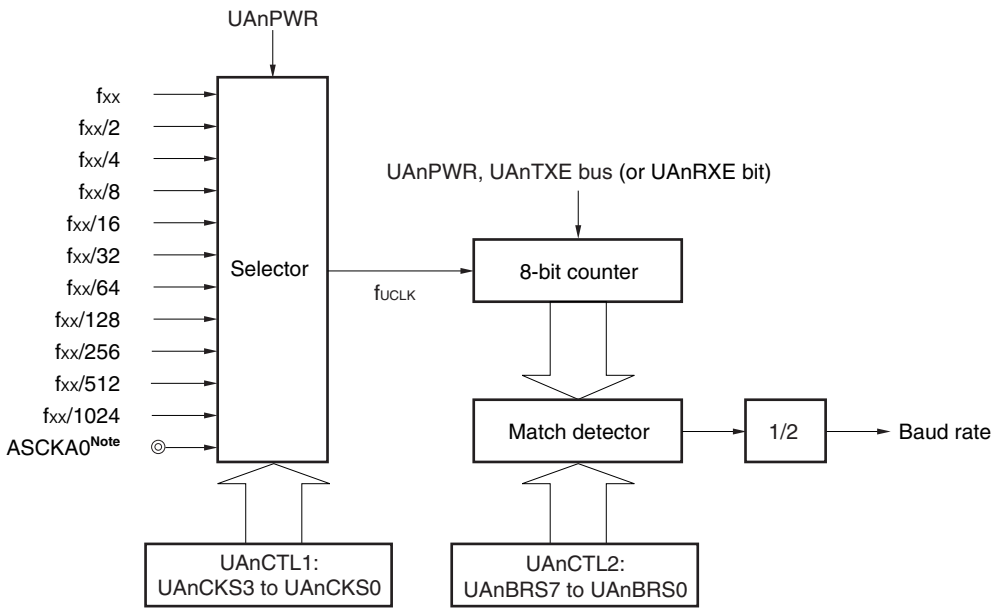
13.6 Dedicated Baud Rate Generator

The dedicated baud rate generator consists of a source clock selector block and an 8-bit programmable counter, and generates a serial clock during transmission and reception with UARTAn. Regarding the serial clock, a dedicated baud rate generator output can be selected for each channel.

There is an 8-bit counter for transmission and another one for reception.

(1) Baud rate generator configuration

Figure 13-9. Configuration of Baud Rate Generator



Note Only UARTA0 is valid; setting UARTA1 is prohibited.

<R> **Caution UARTAn cannot be used if the CPU clock (f_{CPU}) is slower than f_{UCLK}.**

Remarks 1. n = 0, 1

2. f_{XX}: Main clock frequency

f_{UCLK}: Base clock frequency

(a) Base clock

<R> When the UAnCTL0.UAnPWR bit is 1, the clock selected by the UAnCTL1.UAnCKS3 to UAnCTL1.UAnCKS0 bits is supplied to the 8-bit counter. This clock is called the base clock (f_{UCLK}). The base clock f_{UCLK} is fixed to the low level when the UAnPWR bit is 0.

(b) Serial clock generation

A serial clock can be generated by setting the UAnCTL1 register and the UAnCTL2 register (n = 0, 1).

The base clock (f_{UCLK}) is selected by UAnCTL1.UAnCKS3 to UAnCTL1.UAnCKS0 bits.

The frequency division value for the 8-bit counter can be set using the UAnCTL2.UAnBRS7 to UAnCTL2.UAnBRS0 bits.

(2) UARTAn control register 1 (UAnCTL1)

The UAnCTL1 register is an 8-bit register that selects the UARTAn base clock.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

Caution Clear the UAnCTL0.UAnPWR bit to 0 before rewriting the UAnCTL1 register.

After reset: 00H R/W Address: UA0CTL1 FFFFFA01H, UA1CTL1 FFFFFA11H

	7	6	5	4	3	2	1	0
UAnCTL1 (n = 0, 1)	0	0	0	0	UAnCKS3	UAnCKS2	UAnCKS1	UAnCKS0

UAnCKS3	UAnCKS2	UAnCKS1	UAnCKS0	Base clock (f_{UCLK}) selection
0	0	0	0	fxx
0	0	0	1	fxx/2
0	0	1	0	fxx/4
0	0	1	1	fxx/8
0	1	0	0	fxx/16
0	1	0	1	fxx/32
0	1	1	0	fxx/64
0	1	1	1	fxx/128
1	0	0	0	fxx/256
1	0	0	1	fxx/512
1	0	1	0	fxx/1,024
1	0	1	1	External clock[Note] (ASCKA0 pin)
Other than above				Setting prohibited

Note Only UARTA0 is valid; setting UARTA1 is prohibited.

Remark fxx: Main clock frequency

(3) UARTAn control register 2 (UAnCTL2)

The UAnCTL2 register is an 8-bit register that selects the baud rate (serial transfer speed) clock of UARTAn.

This register can be read or written in 8-bit units.

Reset sets this register to FFH.

Caution Clear the UAnCTL0.UAnPWR bit to 0 or clear the UAnTXE and UAnRXE bits to 00 before rewriting the UAnCTL2 register.

After reset FFH R/W Address: UA0CTL2 FFFFFA02H, UA1CTL2 FFFFFA12H

	7	6	5	4	3	2	1	0
UAnCTL2	UAnBRS7	UAnBRS6	UAnBRS5	UAnBRS4	UAnBRS3	UAnBRS2	UAnBRS1	UAnBRS0

(n = 0, 1)

UAn BRS7	UAn BRS6	UAn BRS5	UAn BRS4	UAn BRS3	UAn BRS2	UAn BRS1	UAn BRS0	Default (k)	Serial clock
0	0	0	0	0	0	×	×	×	Setting prohibited
0	0	0	0	0	1	0	0	4	$f_{UCLK}/4$
0	0	0	0	0	1	0	1	5	$f_{UCLK}/5$
0	0	0	0	0	1	1	0	6	$f_{UCLK}/6$
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	0	0	252	$f_{UCLK}/252$
1	1	1	1	1	1	0	1	253	$f_{UCLK}/253$
1	1	1	1	1	1	1	0	254	$f_{UCLK}/254$
1	1	1	1	1	1	1	1	255	$f_{UCLK}/255$

Remark f_{UCLK}: Frequency of basic clock selected by the UAnCTL1.UAnCKS3 to UAnCTL1.UAnCKS0 bits

(4) Baud rate

The baud rate is obtained by the following equation.

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

When using the internal clock, the equation will be as follows (when using the ASCKA0 pin as clock at UARTA0, calculate using the above equation).

$$\text{Baud rate} = \frac{f_{XX}}{2^{m+1} \times k} \text{ [bps]}$$

Remark f_{UCLK} = Frequency of base clock selected by the UAnCTL1.UAnCKS3 to UAnCTL1.UAnCKS0 bits
f_{XX}: Main clock frequency
m = Value set using the UAnCTL1.UAnCKS3 to UAnCTL1.UAnCKS0 bits (m = 0 to 10)
k = Value set using the UAnCTL2.UAnBRS7 to UAnCTL2.UAnBRS0 bits (k = 4 to 255)

The baud rate error is obtained by the following equation.

$$\text{Error (\%)} = \left(\frac{\text{Actual baud rate (baud rate with error)}}{\text{Target baud rate (correct baud rate)}} - 1 \right) \times 100 \text{ [\%]}$$

$$= \left(\frac{f_{UCLK}}{2 \times k \times \text{Target baud rate}} - 1 \right) \times 100 \text{ [\%]}$$

When using the internal clock, the equation will be as follows (when using the ASCKA0 pin as clock at UARTA0, calculate the baud rate error using the above equation).

$$\text{Error (\%)} = \left(\frac{f_{XX}}{2^{m+1} \times k \times \text{Target baud rate}} - 1 \right) \times 100 \text{ [\%]}$$

Cautions 1. The baud rate error during transmission must be within the error tolerance on the receiving side.

2. The baud rate error during reception must satisfy the range indicated in (5) Allowable baud rate range during reception.

To set the baud rate, perform the following calculation for setting the UAnCTL1 and UAnCTL2 registers (when using internal clock).

<1> Set k to fxx/(2 × target baud rate) and m to 0.
<2> If k is 256 or greater (k ≥ 256), reduce k to half (k/2) and increment m by 1 (m + 1).
<3> Repeat Step <2> until k becomes less than 256 (k < 256).
<4> Round off the first decimal point of k to the nearest whole number.
If k becomes 256 after round-off, perform Step <2> again to set k to 128.
<5> Set the value of m to UAnCTL1 register and the value of k to the UAnCTL2 register.

Example: When fxx = 20 MHz and target baud rate = 153,600 bps
<1> k = 20,000,000/(2 × 153,600) = 65.10…, m = 0
<2>, <3> k = 65.10… < 256, m = 0
<4> Set value of UAnCTL2 register: k = 65 = 41H, set value of UAnCTL1 register: m = 0

Actual baud rate = 20,000,000/(2 × 65)
= 153,846 [bps]

Baud rate error = {20,000,000/(2 × 65 × 153,600) − 1} × 100
= 0.160 [%]

The representative examples of baud rate settings are shown below.

Table 13-3. Baud Rate Generator Setting Data

Baud Rate	fxx = 20 MHz			fxx = 18.874 MHz			fxx = 16 MHz			fxx = 10 MHz		
(bps)	UAnCTL1	UAnCTL2	ERR (%)	UAnCTL1	UAnCTL2	ERR (%)	UAnCTL1	UAnCTL2	ERR (%)	UAnCTL1	UAnCTL2	ERR (%)
300	08H	82H	0.16	07H	F6H	−0.10	07H	D0H	0.16	07H	82H	0.16
600	07H	82H	0.16	06H	F6H	−0.10	06H	D0H	0.16	06H	82H	0.16
1,200	06H	82H	0.16	05H	F6H	−0.10	05H	D0H	0.16	05H	82H	0.16
2,400	05H	82H	0.16	04H	F6H	−0.10	04H	D0H	0.16	04H	82H	0.16
4,800	04H	82H	0.16	03H	F6H	−0.10	03H	D0H	0.16	03H	82H	0.16
9,600	03H	82H	0.16	02H	F6H	−0.10	02H	D0H	0.16	02H	82H	0.16
19,200	02H	82H	0.16	01H	F6H	−0.10	01H	D0H	0.16	01H	82H	0.16
31,250	01H	A0H	0.00	01H	97H	−0.01	01H	80H	0.00	00H	A0H	0.00
38,400	01H	82H	0.16	00H	F6H	−0.10	00H	D0H	0.16	00H	82H	0.16
76,800	00H	82H	0.16	00H	7BH	−0.10	00H	68H	0.16	00H	41H	0.16
153,600	00H	41H	0.16	00H	3DH	0.72	00H	34H	0.16	00H	21H	−1.36
312,500	00H	20H	0.00	00H	1EH	0.66	00H	1AH	−1.54	00H	10H	0.00

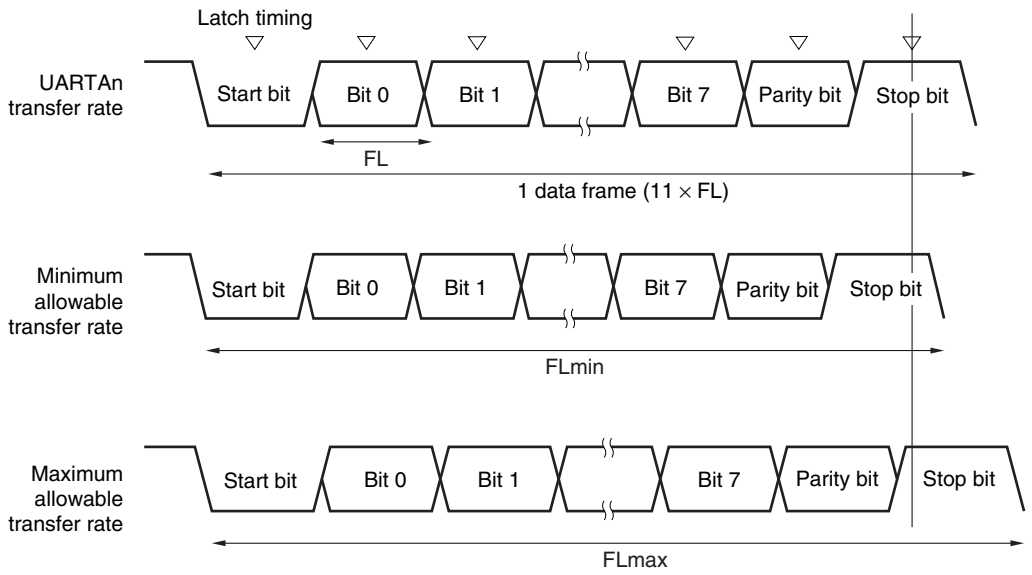
Remark fxx: Main clock frequency
ERR: Baud rate error (%)

(5) Allowable baud rate range during reception

The baud rate error range at the destination that is allowable during reception is shown below.

Caution The baud rate error during reception must be set within the allowable error range using the following equation.

Figure 13-10. Allowable Baud Rate Range During Reception



Remark n = 0, 1

As shown in Figure 13-10, the receive data latch timing is determined by the counter set using the UAnCTL2 register following start bit detection. The transmit data can be normally received if up to the last data (stop bit) can be received in time for this latch timing.

When this is applied to 11-bit reception, the following is the theoretical result.

$FL = (Brate)^{-1}$

Brate: UARTAn baud rate (n = 0, 1)

k: Setting value of UAnCTL2.UAnBRS7 to UAnCTL2.UAnBRS0 bits (n = 0, 1)

FL: 1-bit data length

Latch timing margin: 2 clocks

Minimum allowable transfer rate: $FLmin = 11 \times FL - \frac{k-2}{2k} \times FL = \frac{21k+2}{2k} FL$

Therefore, the maximum baud rate that can be received by the destination is as follows.

$$BRmax = (FLmin/11)^{-1} = \frac{22k}{21k + 2} \text{ Brate}$$

Similarly, obtaining the following maximum allowable transfer rate yields the following.

$$\frac{10}{11} \times FLmax = 11 \times FL - \frac{k + 2}{2 \times k} \times FL = \frac{21k - 2}{2 \times k} FL$$

$$FLmax = \frac{21k - 2}{20\ k} FL \times 11$$

Therefore, the minimum baud rate that can be received by the destination is as follows.

$$BRmin = (FLmax/11)^{-1} = \frac{20k}{21k - 2} \text{ Brate}$$

Obtaining the allowable baud rate error for UARTAn and the destination from the above-described equations for obtaining the minimum and maximum baud rate values yields the following.

Table 13-4. Maximum/Minimum Allowable Baud Rate Error

Division Ratio (k)	Maximum Allowable Baud Rate Error	Minimum Allowable Baud Rate Error
4	+2.32%	–2.43%
8	+3.52%	–3.61%
20	+4.26%	–4.30%
50	+4.56%	–4.58%
100	+4.66%	–4.67%
255	+4.72%	–4.72%

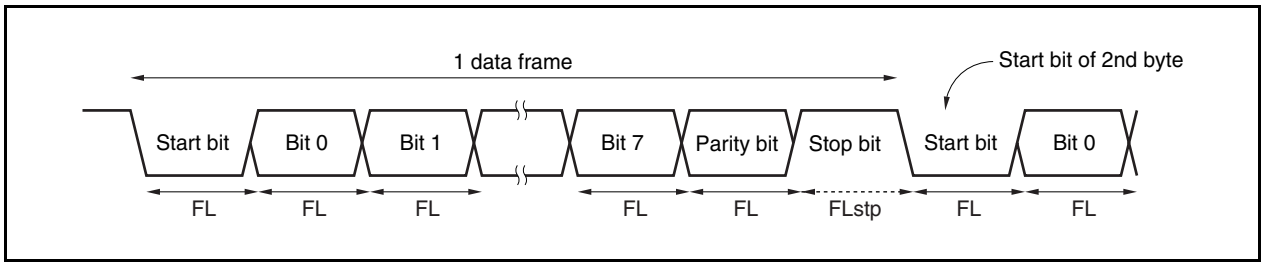
Remarks 1. The reception accuracy depends on the bit count in 1 frame, the input clock frequency, and the division ratio (k). The higher the input clock frequency and the larger the division ratio (k), the higher the accuracy.

2. k: Setting value of UAnCTL2.UAnBRS7 to UAnCTL2.UAnBRS0 bits (n = 0, 1)

(6) Baud rate during continuous transmission

During continuous transmission, the transfer rate from the stop bit to the next start bit is usually 2 base clocks longer. However, timing initialization is performed via start bit detection by the receiving side, so this has no influence on the transfer result.

Figure 13-11. Transfer Rate During Continuous Transfer



Assuming 1 bit data length: FL; stop bit length: FLstp; and base clock frequency: f_{UCLK}, we obtain the following equation.

$$FLstp = FL + 2/f_{UCLK}$$

Therefore, the transfer rate during continuous transmission is as follows.

$$\text{Transfer rate} = 11 \times FL + (2/f_{UCLK})$$

13.7 Cautions

(1) When the clock supply to UARTAn is stopped (for example, in IDLE1, IDLE2, or STOP mode), the operation stops with each register retaining the value it had immediately before the clock supply was stopped. The TXDAn pin output also holds and outputs the value it had immediately before the clock supply was stopped. However, the operation is not guaranteed after the clock supply is resumed. Therefore, after the clock supply is resumed, the circuits should be initialized by setting the UAnCTL0.UAnPWR, UAnCTL0.UAnRXEn, and UAnCTL0.UAnTXEn bits to 000.

(2) The RXDA1 and KR7 pins must not be used at the same time. To use the RXDA1 pin, do not use the KR7 pin. To use the KR7 pin, do not use the RXDA1 pin (it is recommended to set the PFC91 bit to 1 and clear PFCE91 bit to 0).

<R> (3) In UARTAn, the interrupt caused by a communication error does not occur. When performing the transfer of transmit data and receive data using DMA transfer (μPD703253Y only), error processing cannot be performed even if errors (parity, overrun, framing) occur during transfer. Either read the UAnSTR register after DMA transfer has been completed to make sure that there are no errors, or read the UAnSTR register during communication to check for errors.

(4) Start up the UARTAn in the following sequence.
<1> Set the UAnCTL0.UAnPWR bit to 1.
<2> Set the ports.
<3> Set the UAnCTL0.UAnTXE bit to 1, UAnCTL0.UAnRXE bit to 1.

(5) Stop the UARTAn in the following sequence.
<1> Set the UAnCTL0.UAnTXE bit to 0, UAnCTL0.UAnRXE bit to 0.
<2> Set the ports and set the UAnCTL0.UAnPWR bit to 0 (it is not a problem if port setting is not changed).

(6) In transmit mode (UAnCTL0.UAnPWR bit = 1 and UAnCTL0.UAnTXE bit = 1), do not overwrite the same value to the UAnTX register by software because transmission starts by writing to this register. To transmit the same value continuously, overwrite the same value.

(7) In continuous transmission, the communication rate from the stop bit to the next start bit is extended 2 base clocks more than usual. However, the reception side initializes the timing by detecting the start bit, so the reception result is not affected.

14.1 Mode Switching of CSIB and Other Serial Interfaces

14.1.1 CSIB0 and I²C01 mode switching

In the V850ES/SG1, CSIB0 and I²C01 are alternate functions of the same pin and therefore cannot be used simultaneously. Set CSIB0 in advance, using the PMC4 and PFC4 registers, before use.

Caution The transmit/receive operation of CSIB0 and I²C01 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 14-1. CSIB0 and I²C01 Mode Switch Settings

After reset: 00H R/W Address: FFFFF448H

	7	6	5	4	3	2	1	0
PMC4	0	0	0	0	0	PMC42	PMC41	PMC40

After reset: 00H R/W Address: FFFFF468H

	7	6	5	4	3	2	1	0
PFC4	0	0	0	0	0	0	PFC41	PFC40

PMC4n	PFC4n	Operation mode
0	×	Port I/O mode
1	0	CSIB0 mode
1	1	I²C01 mode

Remarks 1. n = 0, 1

2. × = don't care

14.2 Features

- Transfer rate: 8 Mbps Max. (f_{XX} = 20 MHz, using internal clock)
- Master mode and slave mode selectable
- 8-bit to 16-bit transfer, 3-wire serial interface
- Interrupt request signals (INTCBnT, INTCBnR)
- Serial clock and data phase switchable
- Transfer data length selectable in 1-bit units between 8 and 16 bits
- Transfer data MSB-first/LSB-first switchable
- 3-wire transfer SOBn: Serial data output
  SIBn: Serial data input
  $\overline{\text{SCKBn}}$: Serial clock output

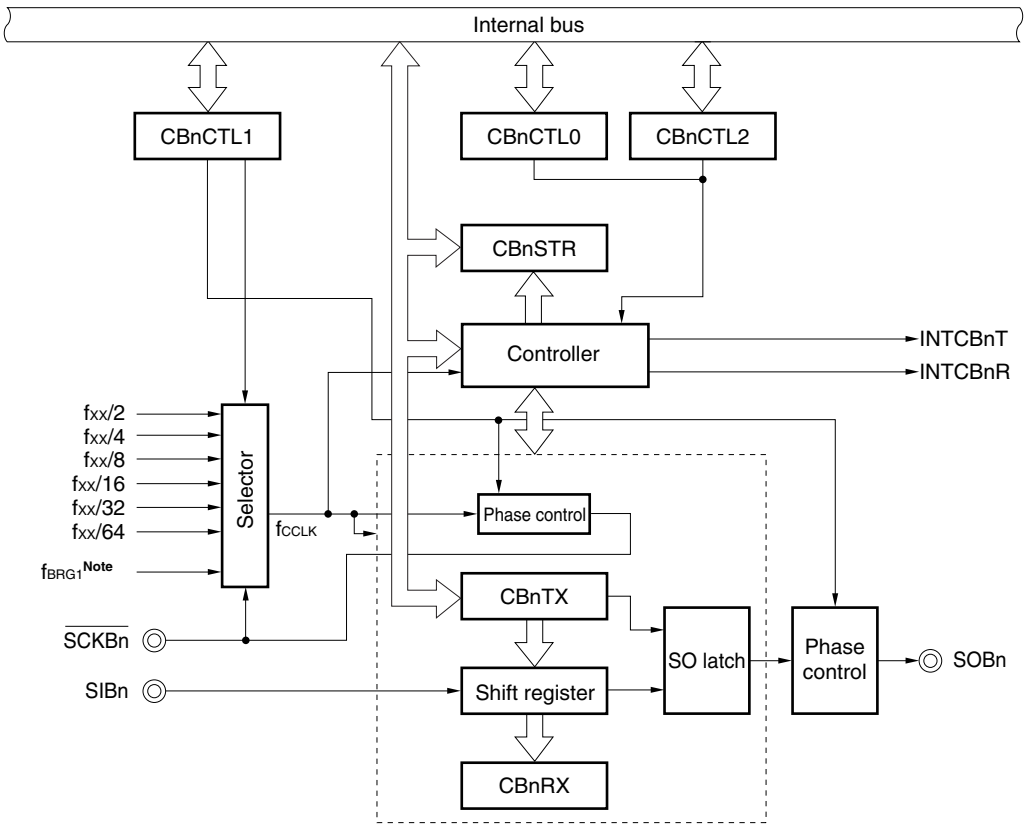
  Transmission mode, reception mode, and transmission/reception mode specifiable

Remark n = 0, 1, 3

14.3 Configuration

The following shows the block diagram of CSIBn.

Figure 14-2. Block Diagram of CSIBn



Note Available only in CSIB0 and CSIB1

Remark f_{CCLK}: Communication clock
f_{XX}: Main clock frequency
f_{BRG1}: BRG1 count clock
n = 0, 1, 3

CSIBn includes the following hardware.

Table 14-1. Configuration of CSIBn

Item	Configuration
Registers	CSIBn receive data register (CBnRX) CSIBn transmit data register (CBnTX)
Control registers	CSIBn control register 0 (CBnCTL0) CSIBn control register 1 (CBnCTL1) CSIBn control register 2 (CBnCTL2) CSIBn status register (CBnSTR)

(1) CSIBn receive data register (CBnRX)

The CBnRX register is a 16-bit buffer register that holds receive data.

This register is read-only, in 16-bit units.

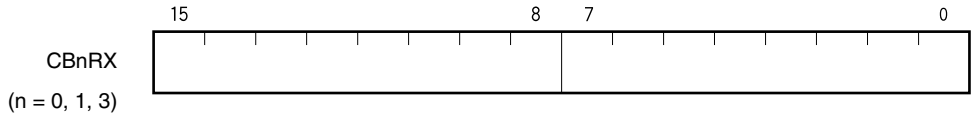
The receive operation is started by reading the CBnRX register in the reception enabled status.

If the transfer data length is 8 bits, the lower 8 bits of this register are read-only in 8-bit units as the CBnRXL register.

Reset sets this register to 0000H.

In addition to reset input, the CBnRX register can be initialized by clearing (to 0) the CBnPWR bit of the CBnCTL0 register.

After reset: 0000H R Address: CB0RX FFFFFD04H, CB1RX FFFFFD14H, CB3RX FFFFFD34H



(2) CSIBn transmit data register (CBnTX)

The CBnTX register is a 16-bit buffer register used to write the CSIBn transfer data.

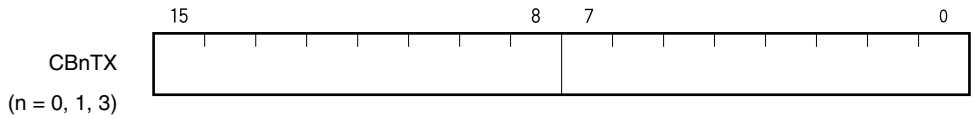
This register can be read or written in 16-bit units.

The transmit operation is started by writing data to the CBnTX register in the transmission enabled status.

If the transfer data length is 8 bits, the lower 8 bits of this register are read-only in 8-bit units as the CBnTXL register.

Reset sets this register to 0000H.

After reset 0000H R/W Address: CB0TX FFFFFD06H, CB1TX FFFFFD16H, CB3TX FFFFFD36H



Remark The communication start conditions are shown below.

Transmission mode (CBnTXE bit = 1, CBnRXE bit = 0):	Write to CBnTX register
Transmission/reception mode (CBnTXE bit = 1, CBnRXE bit = 1):	Write to CBnTX register
Reception mode (CBnTXE bit = 0, CBnRXE bit = 1):	Read from CBnRX register

14.4 Registers

The following registers are used to control CSIBn.

- CSIBn control register 0 (CBnCTL0)
- CSIBn control register 1 (CBnCTL1)
- CSIBn control register 2 (CBnCTL2)
- CSIBn status register (CBnSTR)

(1) CSIBn control register 0 (CBnCTL0)

CBnCTL0 is a register that controls the CSIBn serial transfer operation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 01H.

After reset: 01H R/W Address: CB0CTL0 FFFFFD00H, CB1CTL0 FFFFFD10H, CB3CTL0 FFFFFD30H

CBnCTL0 (n = 0, 1, 3)

<7>	<6>	<5>	<4>	3	2	1	<0>
CBnPWR	CBnTXE[Note]	CBnRXE[Note]	CBnDIR[Note]	0	0	CBnTMS[Note]	CBnSCE

CBnPWR	Specification of CSIBn operation disable/enable
0	Disable CSIBn operation and reset the CBnSTR register
1	Enable CSIBn operation
• The CBnPWR bit controls the CSIBn operation and resets the internal circuit.	

CBnTXE[Note]	Specification of transmit operation disable/enable
0	Disable transmit operation
1	Enable transmit operation
• The SOBn output is low level when the CBnTXE bit is 0.	

CBnRXE[Note]	Specification of receive operation disable/enable
0	Disable receive operation
1	Enable receive operation
• When the CBnRXE bit is cleared to 0, no reception complete interrupt is output even when the prescribed data is transferred in order to disable the receive operation, and the receive data (CBnRX register) is not updated.	

Note These bits can only be rewritten when the CBnPWR bit = 0. However, CBnPWR bit = 1 can also be set at the same time as rewriting these bits.

Cautions 1. To forcibly suspend transmission/reception, clear the CBnPWR bit instead of the CBnTXE and CBnRXE bit to 0. At this time, the clock output is stopped.

<R> **2. Be sure to set bits 3 and 2 to "0".**

(2/3)

CBnDIR[Note]	Specification of transfer direction mode (MSB/LSB)
0	MSB-first transfer
1	LSB-first transfer

CBnTMS[Note]	Transfer mode specification
0	Single transfer mode
1	Continuous transfer mode

[In single transfer mode]
The reception complete interrupt (INTCBnR) occurs when communication is complete.
Even if transmission is enabled (CBnTXE bit = 1), the transmission enable interrupt (INTCBnT) does not occur.
If the next transmit data is written during communication (CBnSTR.CBnTSF bit = 1), it is ignored and the next communication is not started. Also, if reception-only communication is set (CBnTXE bit = 0, CBnRXE bit = 1), the next communication is not started even if the receive data is read during communication (CBnSTR. CBnTSF bit = 1).

[In continuous transfer mode]
The continuous transmission is enabled by writing the next transmit data during communication (CBnSTR.CBnTSF bit = 1). Writing the next transmission data is enabled after a transmission enable interrupt (INTCBnT) occurrence.
If reception-only communication is set (CBnTXE bit = 0, CBnRXE bit = 1) in the continuous transfer mode, the next reception is started continuously after a reception complete interrupt (INTCBnR) regardless of the read operation of the CBnRX register.
Therefore, read immediately the receive data from the CBnRX register. If this read operation is delayed, an overrun error (CBnOVE bit = 1) occurs.

Note These bits can only be rewritten when the CBnPWR bit = 0. However, the CBnPWR can be set to 1 at the same time as these bits are rewritten.

(3/3)

CBnSCE	Specification of start transfer disable/enable
0	Communication start trigger invalid
1	Communication start trigger valid

- In master mode
  This bit enables or disables the communication start trigger.
  (a) In single transmission or transmission/reception mode, or continuous transmission or continuous transmission/reception mode
  The setting of the CBnSCE bit has no influence on communication operation.
  (b) In single reception mode
  Clear the CBnSCE bit to 0 before reading the last receive data because reception is started by reading the receive data (CBnRX register) to disable the reception startup[Note 1].
  (c) In continuous reception mode
  Clear the CBnSCE bit to 0 one communication clock before reception of the last data is completed to disable the reception startup after the last data is received[Note 2].
- In slave mode
  This bit enables or disables the communication start trigger.
  Set the CBnSCE bit to 1.[Note 3] <R>

Notes 1. If the CBnSCE bit is read while it is 1, the next communication operation is started.

2. The CBnSCE bit is not cleared to 0 one communication clock before the completion of the last data reception, the next communication operation is automatically started.
To start the communication operation again after the last data has been read, set the CBnSCE bit to “1” and dummy-read the CBnRX register. <R>

3. To start the reception, dummy reading is necessary. <R>

<R>

(a) How to use CBnSCE bit

(i) In single reception mode

<1> When the reception of the last data is completed with INTCBnR interrupt servicing, clear the CBnSCE bit to 0, and then read the CBnRX register.

<2> When the reception is disabled after the reception of the last data has been completed, check that the CBnSTR.CBnTSF bit is 0, and then clear the CBnPWR and CBnRXE bits to 0. To continue reception, set the CBnSCE bit to 1 and start the next receive operation by performing a dummy read of the CBnRX register.

(ii) In continuous reception mode

<1> Clear the CBnSCE bit to 0 during reception of the last data with INTCBnR interrupt servicing by the reception before the last reception, and then read the CBnRX register.

<2> After receiving the INTCBnR signal of the last reception, read the last data from the CBnRX register.

<3> When the reception is disabled after the reception of the last data has been completed, check that the CBnSTR.CBnTSF bit is 0, and then clear the CBnPWR and CBnRXE bits to 0. To continue reception, set the CBnSCE bit to 1 and start the next receive operation by performing a dummy read of the CBnRX register.

Caution In continuous reception mode, the serial clock is not stopped until the reception executed when the CBnSCE bit is cleared to 0 is completed after the reception is started by a dummy read.

(2) CSIBn control register 1 (CBnCTL1)

CBnCTL1 is an 8-bit register that controls the CSIBn serial transfer operation.

This register can be read or written in 8-bit or 1-bit units.

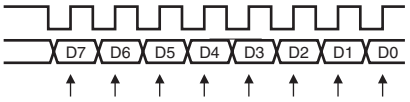
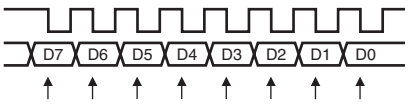
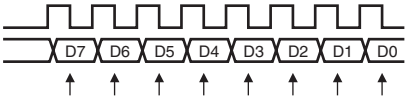
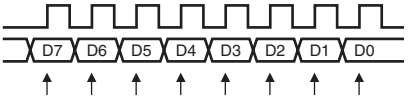
Reset sets this register to 00H.

Caution The CBnCTL1 register can be rewritten only when the CBnCTL0.CBnPWR bit = 0.

After reset 00H R/W Address: CB0CTL1 FFFFFD01H, CB1CTL1 FFFFFD11H, CB3CTL1 FFFFFD31H

	7	6	5	4	3	2	1	0
CBnCTL1	0	0	0	CBnCKP	CBnDAP	CBnCKS2	CBnCKS1	CBnCKS0

(n = 0, 1, 3)

	CBnCKP	CBnDAP	Specification of data transmission/ reception timing in relation to $\overline{SCKBn}$
Communication type 1	0	0	
Communication type 2	0	1	
Communication type 3	1	0	
Communication type 4	1	1	

<R>

CBnCKS2	CBnCKS1	CBnCKS0	Communication clock (f_{CCLK})Note 1	Mode
0	0	0	$f_{XX}/2$	Master mode
0	0	1	$f_{XX}/4$	Master mode
0	1	0	$f_{XX}/8$	Master mode
0	1	1	$f_{XX}/16$	Master mode
1	0	0	$f_{XX}/32$	Master mode
1	0	1	$f_{XX}/64$	Master mode
1	1	0	f_{BRG1}Note 2	Master mode
1	1	1	External clock ($\overline{SCKBn}$)	Slave mode

<R>

Notes 1. Set the communication clock (f_{CCLK}) to 8 MHz or lower.

2. Available only in CSIB0 and CSIB1

For details of f_{BRG1}, see **14.8 Baud Rate Generator**.

(3) CSIBn control register 2 (CBnCTL2)

CBnCTL2 is an 8-bit register that controls the number of CSIBn serial transfer bits.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

Caution The CBnCTL2 register can be rewritten only when the CBnCTL0.CBnPWR bit = 0 or when both the CBnTXE and CBnRXE bits = 0.

After reset: 00H R/W Address: CB0CTL2 FFFFFD02H, CB1CTL2 FFFFFD12H, CB3CTL2 FFFFFD32H

CBnCTL2 (n = 0, 1, 3)

7	6	5	4	3	2	1	0
0	0	0	0	CBnCL3	CBnCL2	CBnCL1	CBnCL0

CBnCL3	CBnCL2	CBnCL1	CBnCL0	Serial register bit length
0	0	0	0	8 bits
0	0	0	1	9 bits
0	0	1	0	10 bits
0	0	1	1	11 bits
0	1	0	0	12 bits
0	1	0	1	13 bits
0	1	1	0	14 bits
0	1	1	1	15 bits
1	×	×	×	16 bits

Remarks 1. If the number of transfer bits is other than 8 or 16, prepare and use data stuffed from the LSB of the CBnTX and CBnRX registers.

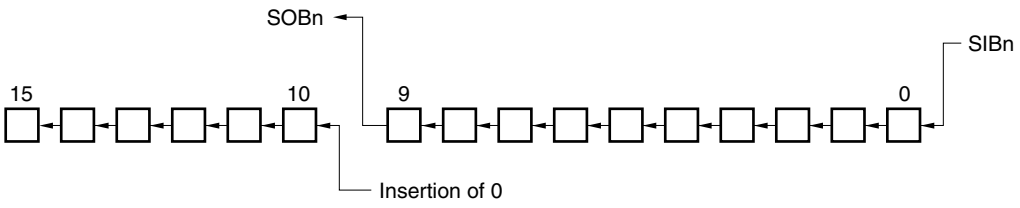
2. ×: don't care

(a) Transfer data length change function

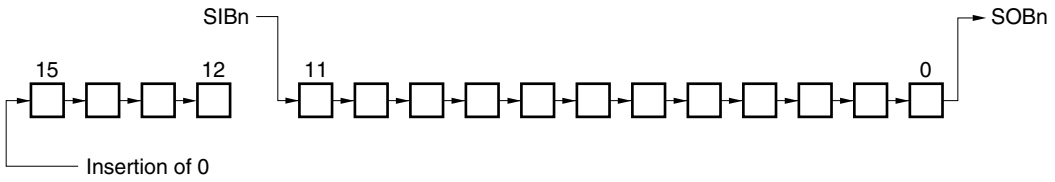
The CSIBn transfer data length can be set in 1-bit units between 8 and 16 bits using the CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits.

When the transfer bit length is set to a value other than 16 bits, set the data to the CBnTX or CBnRX register starting from the LSB, regardless of whether the transfer start bit is the MSB or LSB. Any data can be set for the higher bits that are not used, but the receive data becomes 0 following serial transfer.

(i) Transfer bit length = 10 bits, MSB first



(ii) Transfer bit length = 12 bits, LSB first



(4) CSIBn status register (CBnSTR)

CBnSTR is an 8-bit register that displays the CSIBn status.

This register can be read or written in 8-bit or 1-bit units, but the CBnTSF flag is read-only.

Reset sets this register to 00H.

In addition to reset input, the CBnSTR register can be initialized by clearing (0) the CBnCTL0.CBnPWR bit.

After reset 00H R/W Address: CB0STR FFFFFD03H, CB1STR FFFFFD13H, CB3STR FFFFFD33H

CBnSTR (n = 0, 1, 3)

<7>	6	5	4	3	2	1	<0>
CBnTSF	0	0	0	0	0	0	CBnOVE

CBnTSF	Communication status flag
0	Communication stopped
1	Communicating

- During transmission, this register is set when data is prepared in the CBnTX register, and during reception, it is set when a dummy read of the CBnRX register is performed.
  When transfer ends, this flag is cleared to 0 at the last edge of the clock.

CBnOVE	Overrun error flag
0	No overrun
1	Overrun

- An overrun error occurs when the next reception starts without reading the value of the CBnRX register by CPU, upon completion of the receive operation.
  The CBnOVE flag displays the overrun error occurrence status in this case.
- The CBnOVE bit is valid also in the single transfer mode. Therefore, when only using transmission, note the following.
  - Do not check the CBnOVE flag. (recommended)
  - Read this bit even if reading the reception data is not required.
- The CBnOVE flag is cleared by writing 0 to it. It cannot be set even by writing 1 to it.

<R> **Caution In single transfer mode, writing to the CBnTX register with the CBnTSF bit set to 1 is ignored. This has no influence on the operation during transfer.**

For example, if the next data is written to the CBnTX register when DMA is started by generating the INTCBnR signal (μPD703253Y only), the written data is not transferred because the CBnTSF bit is set to 1.

Use the continuous transfer mode, not the single transfer mode, for such applications.

<R> 14.5 Interrupt Request Signals

CSIBn can generate the following two types of interrupt request signals.

- Reception completion interrupt request signal (INTCBnR)
- Transmission enable interrupt request signal (INTCBnT)

Of these two interrupt request signals, the reception completion interrupt request signal has the higher priority by default, and the priority of the transmission enable interrupt request signal is lower.

Table 14-2. Interrupts and Their Default Priority

Interrupt	Priority
Reception complete	High
Transmission enable	Low

(1) Reception completion interrupt request signal (INTCBnR)

When receive data is transferred to the CBnRX register while reception is enabled, the reception completion interrupt request signal is generated.

This interrupt request signal can also be generated if an overrun error occurs.

When the reception completion interrupt request signal is acknowledged and the data is read, read the CBnSTR register to check that the result of reception is not an error.

In the single transfer mode, the INTCBnR interrupt request signal is generated upon completion of transmission, even when only transmission is executed.

(2) Transmission enable interrupt request signal (INTCBnT)

In the continuous transmission or continuous transmission/reception mode, transmit data is transferred from the CBnTX register and, as soon as writing to CBnTX has been enabled, the transmission enable interrupt request signal is generated.

In the single transmission and single transmission/reception modes, the INTCBnT interrupt is not generated.

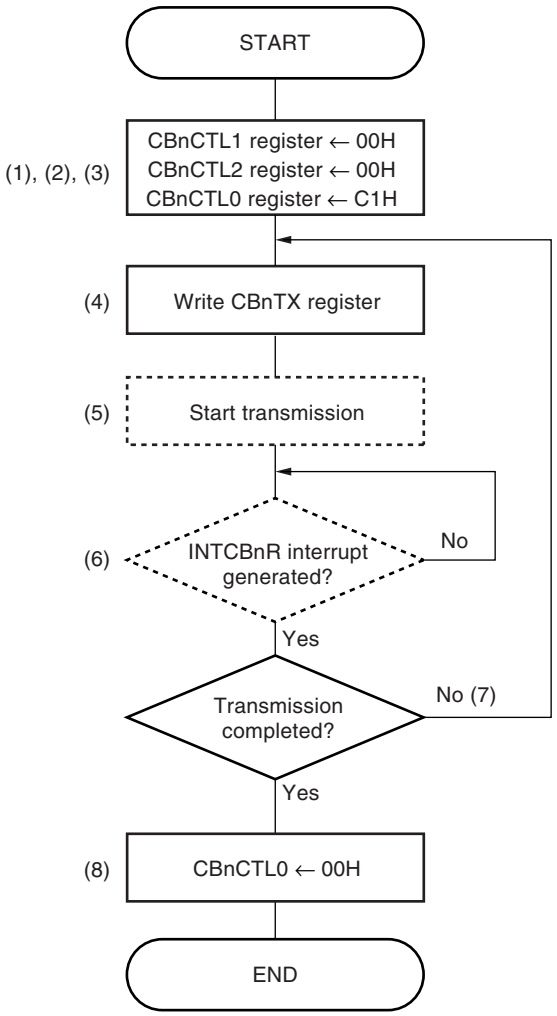
14.6 Operation

<R>

14.6.1 Single transfer mode (master mode, transmission mode)

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = f_{XX}/2 (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 000), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

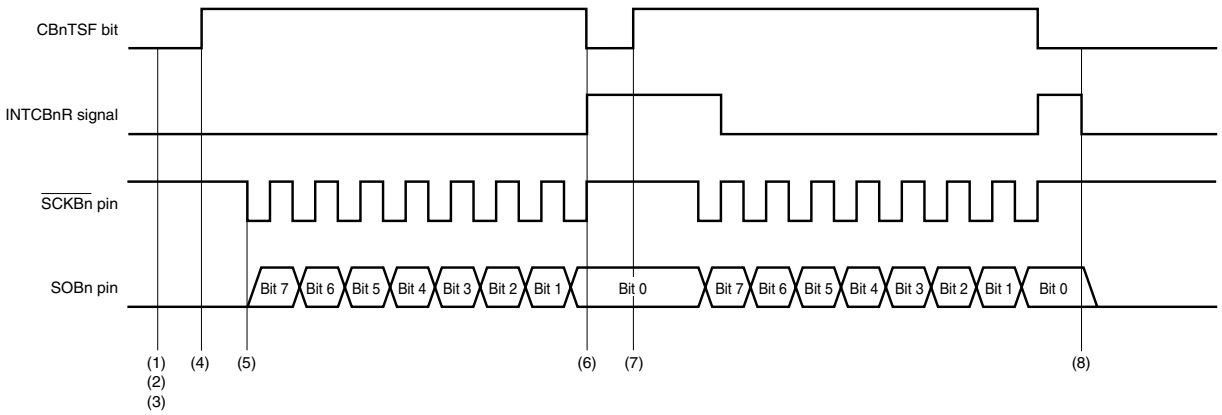


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 00H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = f_{XX}/2, and master mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write C1H to the CBnCTL0 register, and select the transmission mode and MSB first at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by writing the transmit data to the CBnTX register, and transmission is started.

(5) When transmission is started, output the serial clock to the $\overline{\text{SCKBn}}$ pin, and output the transmit data from the SOBn pin in synchronization with the serial clock.

(6) When transmission of the transfer data length set with the CBnCTL2 register is completed, stop the serial clock output and transmit data output, generate the reception completion interrupt request signal (INTCBnR) at the last edge of the serial clock, and clear the CBnTSF bit to 0.

(7) To continue transmission, start the next transmission by writing the transmit data to the CBnTX register again after the INTCBnR signal is generated.

(8) To end transmission, write the CBnCTL0.CBnPWR bit = 0 and the CBnCTL0.CBnTXE bit = 0.

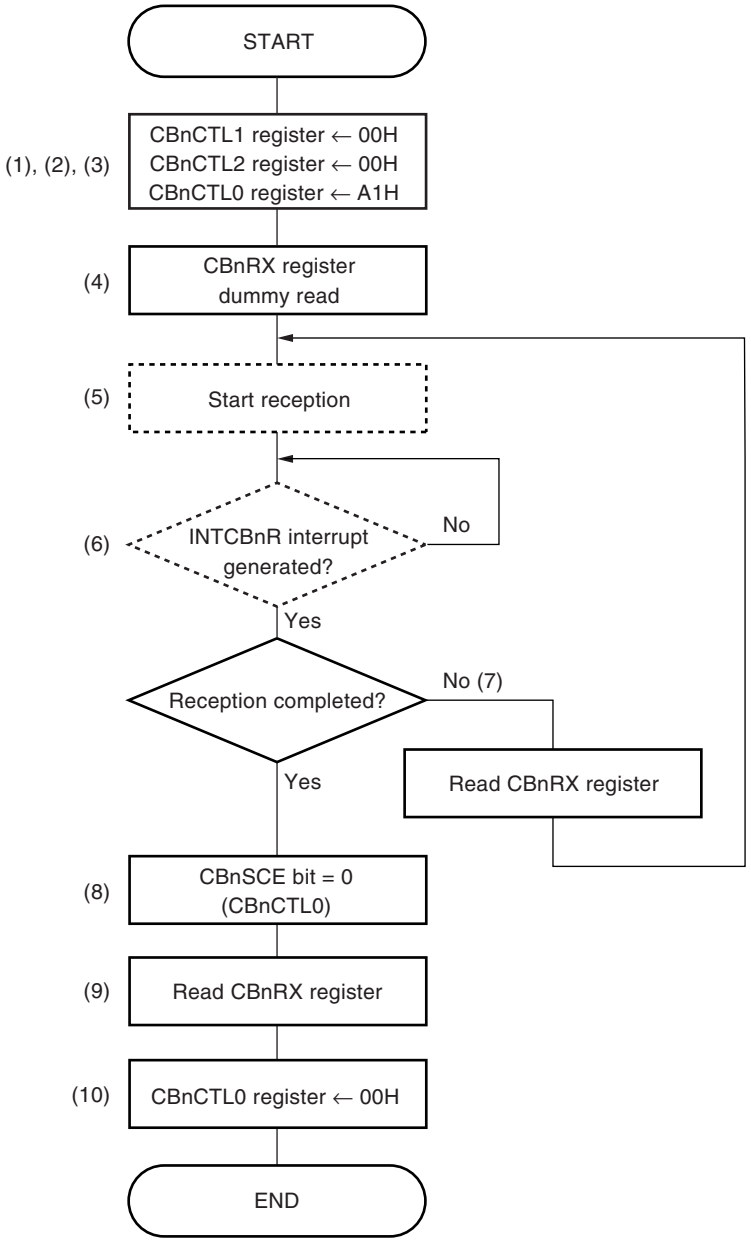
Remark n = 0, 1, and 3

<R>

14.6.2 Single transfer mode (master mode, reception mode)

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = f_{XX}/2 (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 000), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

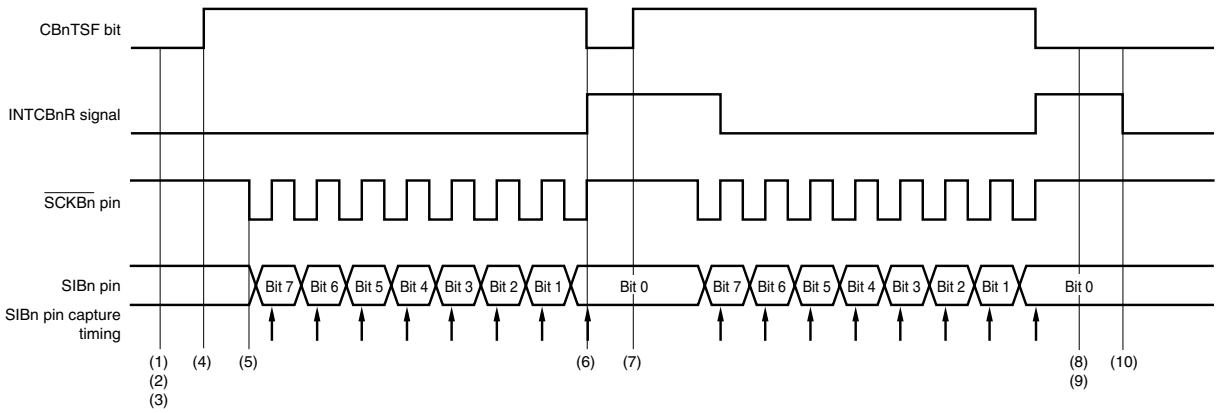


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 00H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = f_{XX}/2, and master mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write A1H to the CBnCTL0 register, and select the reception mode and MSB first at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by performing a dummy read of the CBnRX register, and reception is started.

(5) When reception is started, output the serial clock to the $\overline{\text{SCKBn}}$ pin, and capture the receive data of the SIBn pin in synchronization with the serial clock.

(6) When reception of the transfer data length set with the CBnCTL2 register is completed, stop the serial clock output and data capturing, generate the reception completion interrupt request signal (INTCBnR) at the last edge of the serial clock, and clear the CBnTSF bit to 0.

(7) To continue reception, read the CBnRX register with the CBnCTL0.CBnSCE bit = 1 remained after the INTCBnR signal is generated.

(8) To read the CBnRX register without starting the next reception, write the CBnSCE bit = 0.

(9) Read the CBnRX register.

(10) To end reception, write the CBnCTL0.CBnPWR bit = 0 and the CBnCTL0.CBnRXE bit = 0.

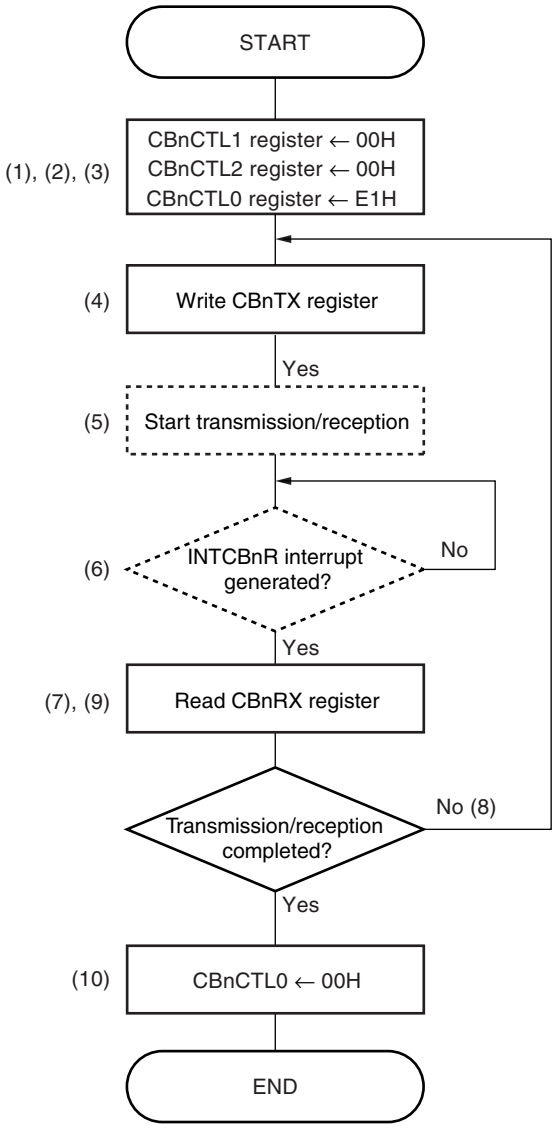
Remark n = 0, 1, and 3

<R>

14.6.3 Single transfer mode (master mode, transmission/reception mode)

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = f_{XX}/2 (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 000), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

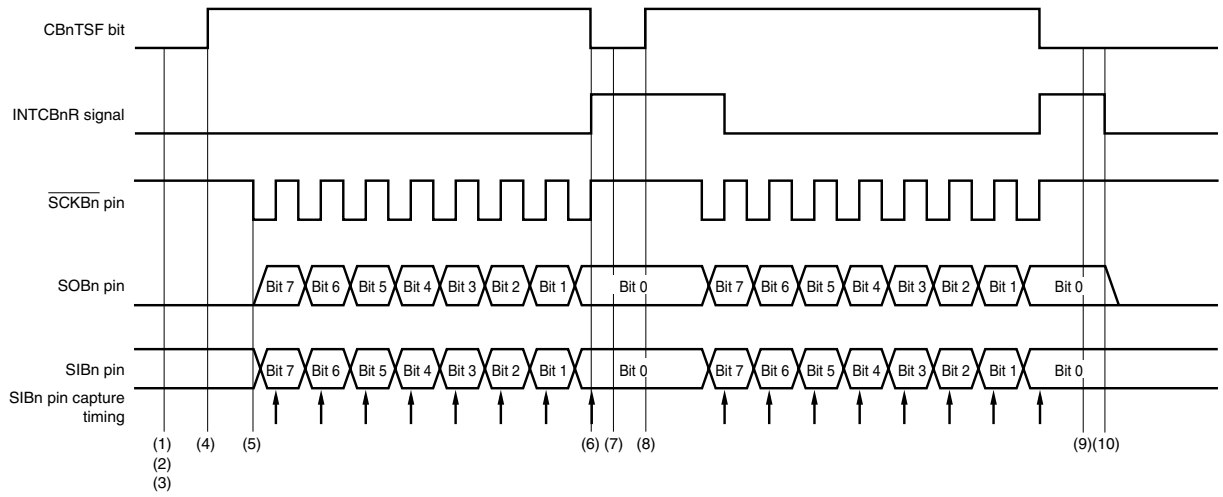


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 00H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = f_{XX}/2, and master mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write E1H to the CBnCTL0 register, and select the transmission/reception mode and MSB first at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by writing the transmit data to the CBnTX register, and transmission/reception is started.

(5) When transmission/reception is started, output the serial clock to the $\overline{\text{SCKBn}}$ pin, output the transmit data to the SOBn pin in synchronization with the serial clock, and capture the receive data of the SIBn pin.

(6) When transmission/reception of the transfer data length set with the CBnCTL2 register is completed, stop the serial clock output, transmit data output, and data capturing, generate the reception completion interrupt request signal (INTCBnR) at the last edge of the serial clock, and clear the CBnTSF bit to 0.

(7) Read the CBnRX register.

(8) To continue transmission/reception, write the transmit data to the CBnTX register again.

(9) Read the CBnRX register.

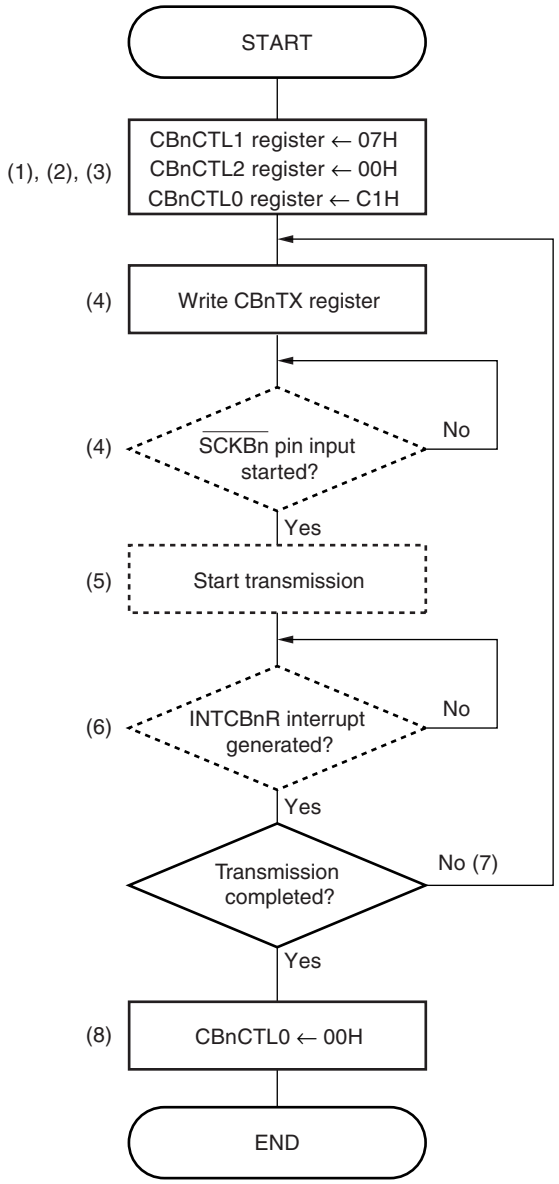
(10) To end transmission/reception, write the CBnCTL0.CBnPWR bit = 0, the CBnCTL0.CBnTXE bit = 0, and the CBnCTL0.CBnRXE bit = 0.

Remark n = 0, 1, and 3

<R> **14.6.4 Single transfer mode (slave mode, transmission mode)**

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$) (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 111), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

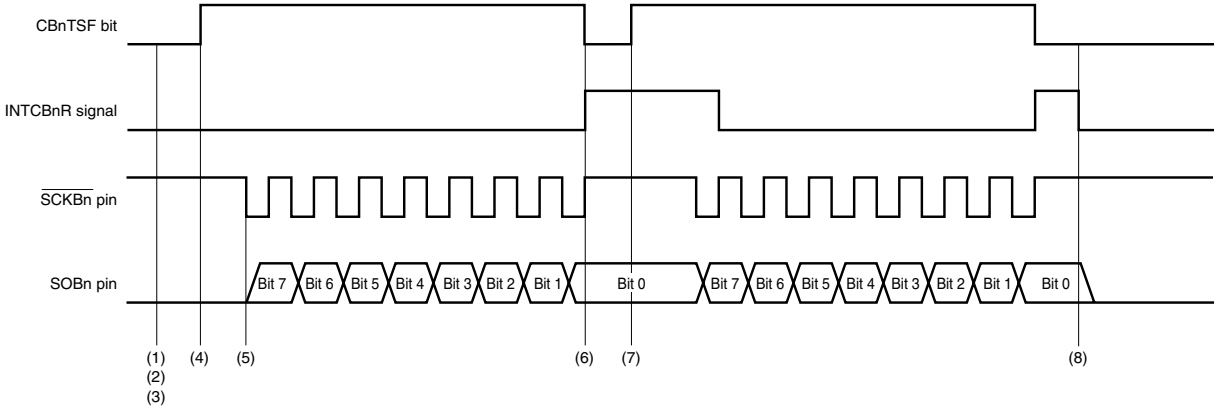


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 07H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$), and slave mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write C1H to the CBnCTL0 register, and select the transmission mode and MSB first at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by writing the transmit data to the CBnTX register, and the device waits for a serial clock input.

(5) When a serial clock is input, output the transmit data from the SOBn pin in synchronization with the serial clock.

(6) When transmission of the transfer data length set with the CBnCTL2 register is completed, stop the serial clock input and transmit data output, generate the reception completion interrupt request signal (INTCBnR) at the last edge of the serial clock, and clear the CBnTSF bit to 0.

(7) To continue transmission, write the transmit data to the CBnTX register again after the INTCBnR signal is generated, and wait for a serial clock input.

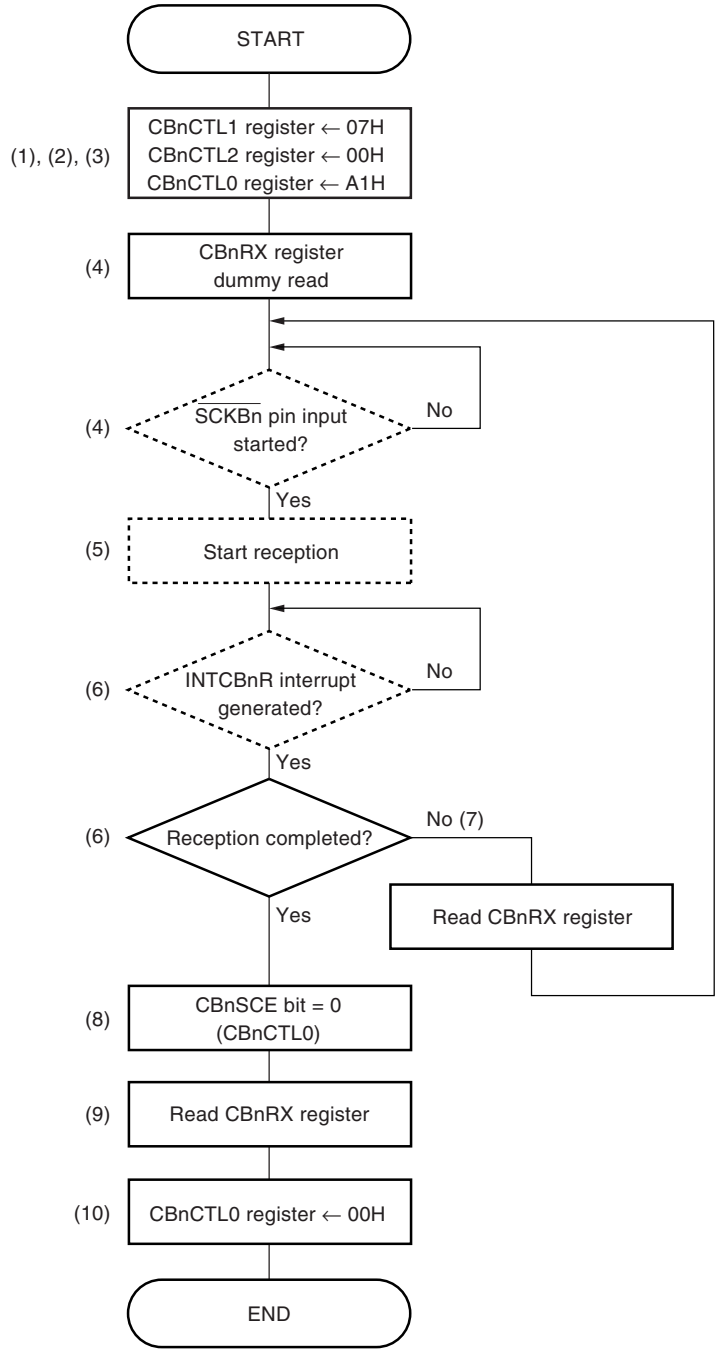
(8) To end transmission, write the CBnCTL0.CBnPWR bit = 0 and the CBnCTL0.CBnTXE bit = 0.

Remark n = 0, 1, and 3

<R> **14.6.5 Single transfer mode (slave mode, reception mode)**

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$) (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 111), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

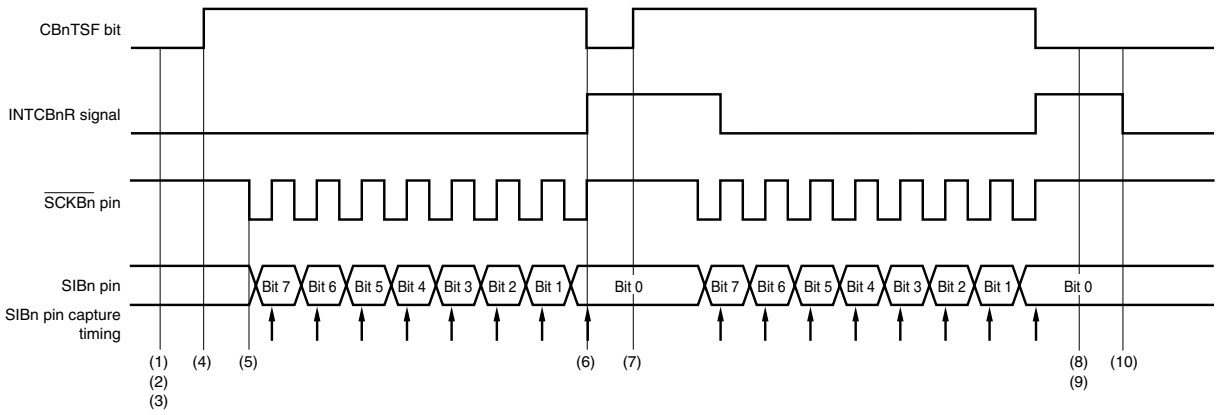


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 07H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$), and slave mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write A1H to the CBnCTL0 register, and select the reception mode and MSB first at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by performing a dummy read of the CBnRX register, and the device waits for a serial clock input.

(5) When a serial clock is input, capture the receive data of the SIBn pin in synchronization with the serial clock.

(6) When reception of the transfer data length set with the CBnCTL2 register is completed, stop the serial clock input and data capturing, generate the reception completion interrupt request signal (INTCBnR) at the last edge of the serial clock, and clear the CBnTSF bit to 0.

(7) To continue reception, read the CBnRX register with the CBnCTL0.CBnSCE bit = 1 remained after the INTCBnR signal is generated, and wait for a serial clock input.

(8) To end reception, write the CBnSCE bit = 0.

(9) Read the CBnRX register.

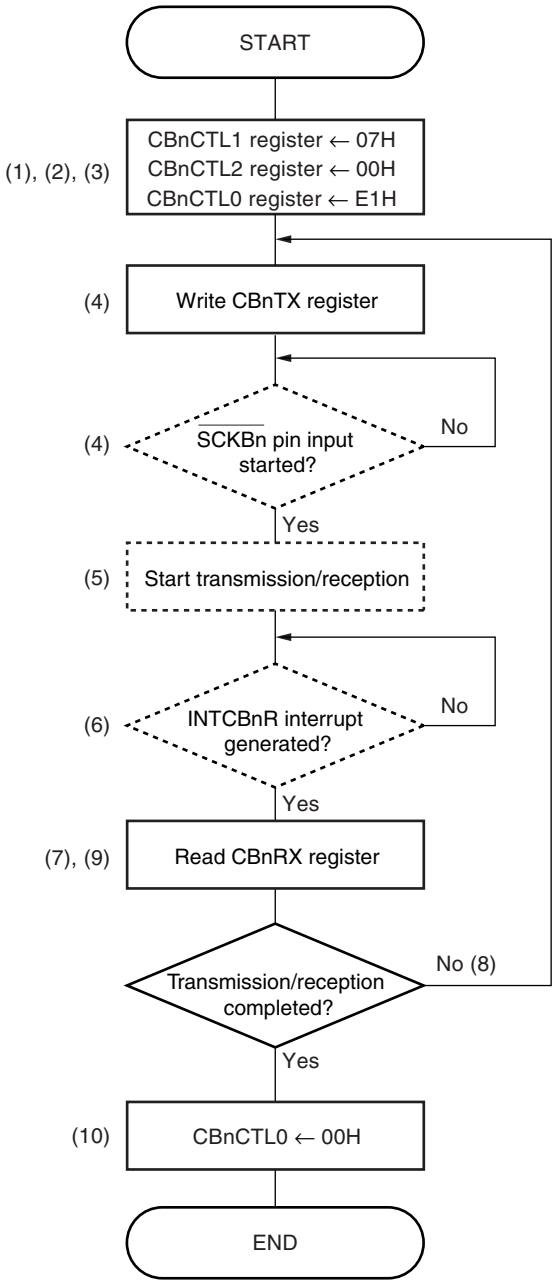
(10) To end reception, write the CBnCTL0.CBnPWR bit = 0 and the CBnCTL0.CBnRXE bit = 0.

Remark n = 0, 1, and 3

<R> **14.6.6 Single transfer mode (slave mode, transmission/reception mode)**

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$) (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 111), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

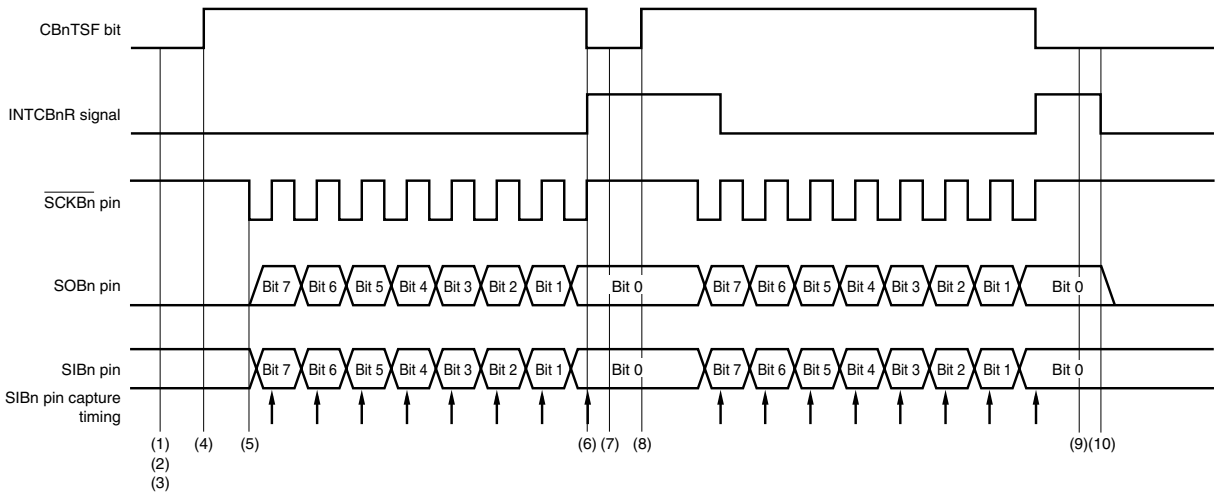


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 07H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$), and slave mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write E1H to the CBnCTL0 register, and select the transmission/reception mode and MSB first at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by writing the transmit data to the CBnTX register, and the device waits for a serial clock input.

(5) When a serial clock is input, output the transmit data to the SOBn pin in synchronization with the serial clock, and capture the receive data of the SIBn pin.

(6) When transmission/reception of the transfer data length set with the CBnCTL2 register is completed, stop the serial clock input, transmit data output, and data capturing, generate the reception completion interrupt request signal (INTCBnR) at the last edge of the serial clock, and clear the CBnTSF bit to 0.

(7) Read the CBnRX register.

(8) To continue transmission/reception, write the transmit data to the CBnTX register again, and wait for a serial clock input.

(9) Read the CBnRX register.

(10) To end transmission/reception, write the CBnCTL0.CBnPWR bit = 0, the CBnCTL0.CBnTXE bit = 0, and the CBnCTL0.CBnRXE bit = 0.

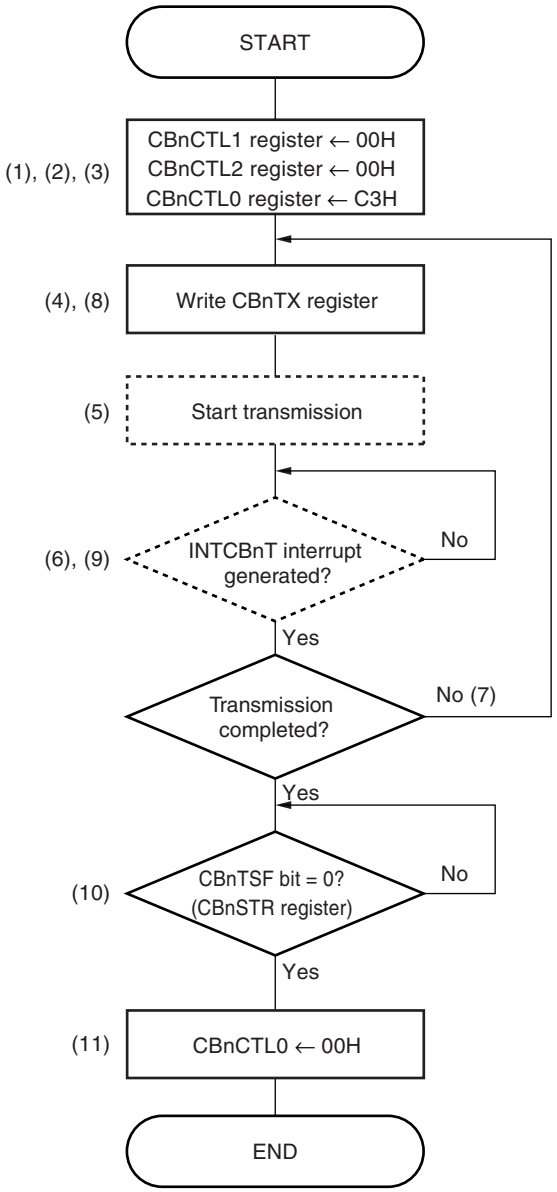
Remark n = 0, 1, and 3

<R>

14.6.7 Continuous transfer mode (master mode, transmission mode)

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = f_{XX}/2 (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 000), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

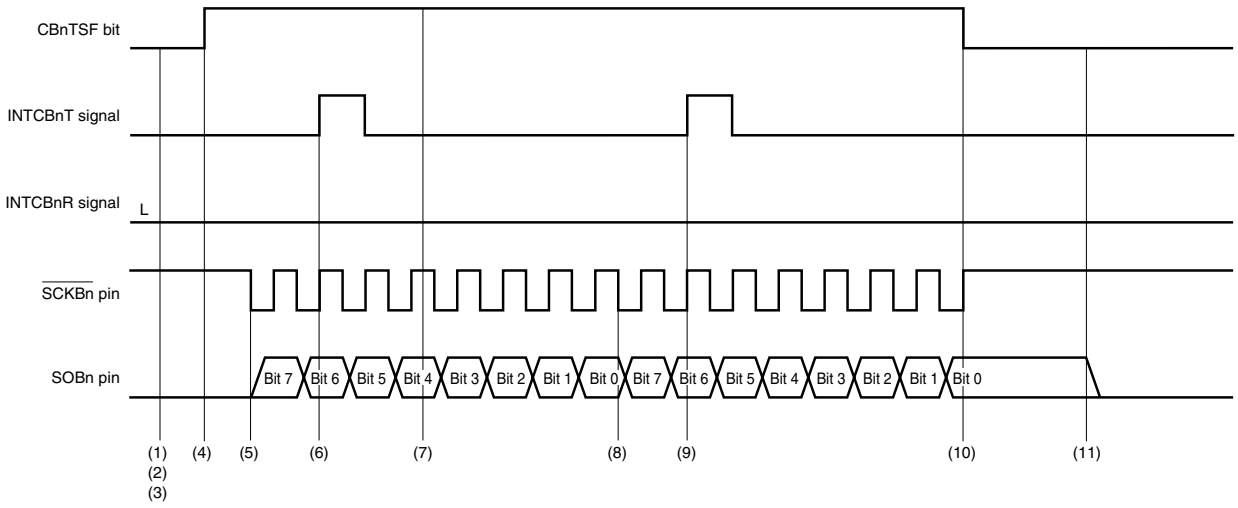


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 00H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = f_{XX}/2, and master mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write C3H to the CBnCTL0 register, and select the transmission mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by writing the transmit data to the CBnTX register, and transmission is started.

(5) When transmission is started, output the serial clock to the $\overline{\text{SCKBn}}$ pin, and output the transmit data from the SOBn pin in synchronization with the serial clock.

(6) When transfer of the transmit data from the CBnTX register to the shift register is completed and writing to the CBnTX register is enabled, the transmission enable interrupt request signal (INTCBnT) is generated.

(7) To continue transmission, write the transmit data to the CBnTX register again after the INTCBnT signal is generated.

(8) When a new transmit data is written to the CBnTX register before communication completion, the next communication is started following communication completion.

(9) The transfer of the transmit data from the CBnTX register to the shift register is completed and the INTCBnT signal is generated. To end continuous transmission with the current transmission, do not write to the CBnTX register.

(10) When the next transmit data is not written to the CBnTX register before transfer completion, stop the serial clock output to the $\overline{\text{SCKBn}}$ pin after transfer completion, and clear the CBnTSF bit to 0.

(11) To release the transmission enable status, write the CBnCTL0.CBnPWR bit = 0 and the CBnCTL0.CBnTXE bit = 0 after checking that the CBnTSF bit = 0.

Caution In continuous transmission mode, the reception completion interrupt request signal (INTCBnR) is not generated.

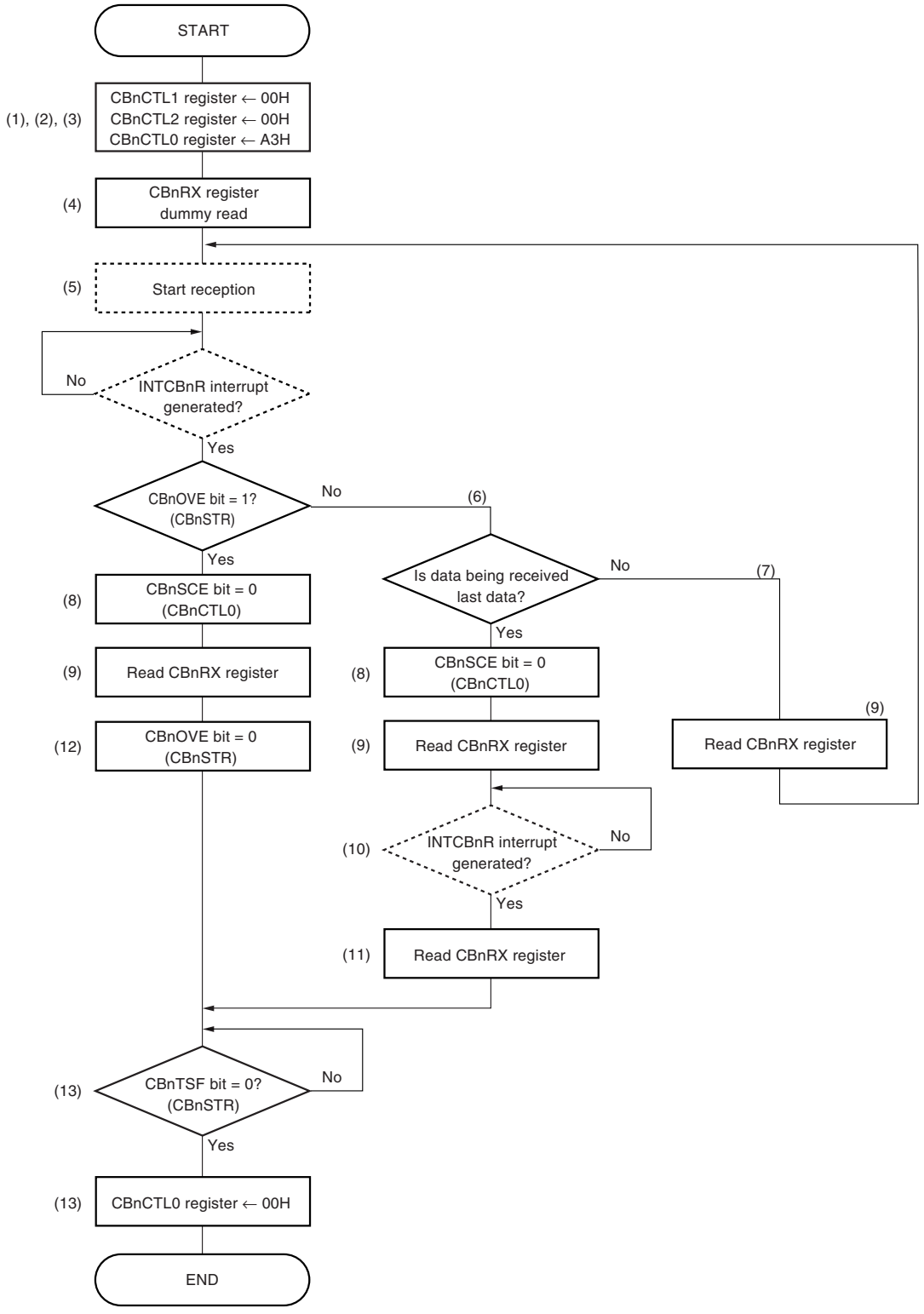
Remark n = 0, 1, and 3

<R>

14.6.8 Continuous transfer mode (master mode, reception mode)

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = f_{XX}/2 (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 000), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

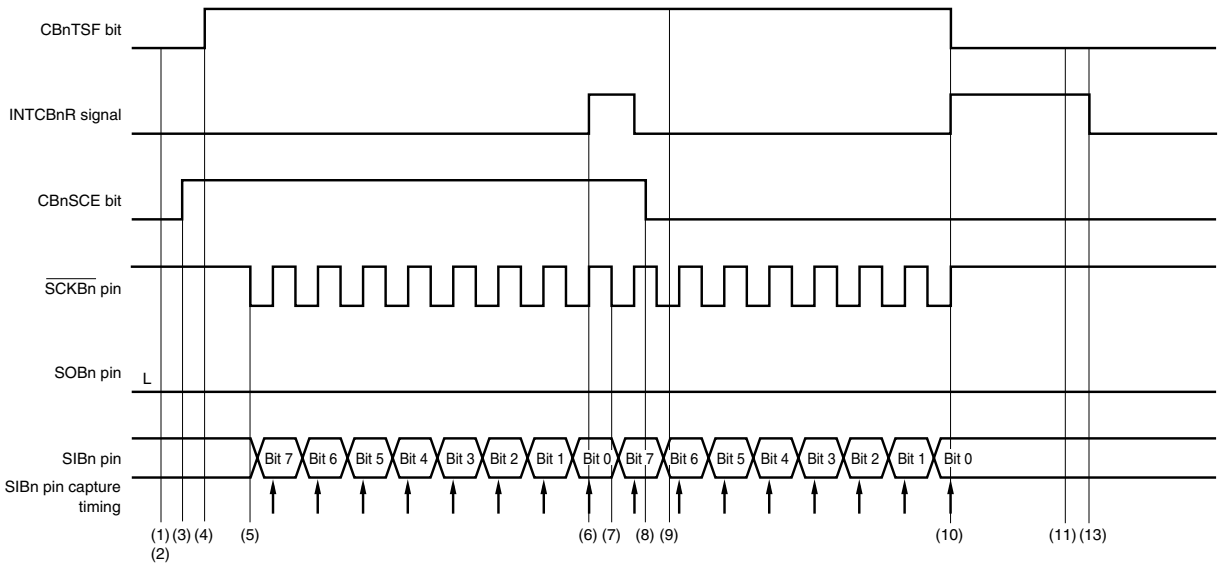


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 00H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = f_{XX}/2, and master mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write A3H to the CBnCTL0 register, and select the reception mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by performing a dummy read of the CBnRX register, and reception is started.

(5) When reception is started, output the serial clock to the $\overline{\text{SCKBn}}$ pin, and capture the receive data of the SIBn pin in synchronization with the serial clock.

(6) When reception is completed, the reception completion interrupt request signal (INTCBnR) is generated, and reading of the CBnRX register is enabled.

(7) When the CBnCTL0.CBnSCE bit = 1 upon communication completion, the next communication is started following communication completion.

(8) To end continuous reception with the current reception, write the CBnSCE bit = 0.

(9) Read the CBnRX register.

(10) When reception is completed, the INTCBnR signal is generated, and reading of the CBnRX register is enabled. When the CBnSCE bit = 0 is set before communication completion, stop the serial clock output to the $\overline{\text{SCKBn}}$ pin, and clear the CBnTSF bit to 0, to end the receive operation.

(11) Read the CBnRX register.

(12) If an overrun error occurs, write the CBnSTR.CBnOVE bit = 0, and clear the error flag.

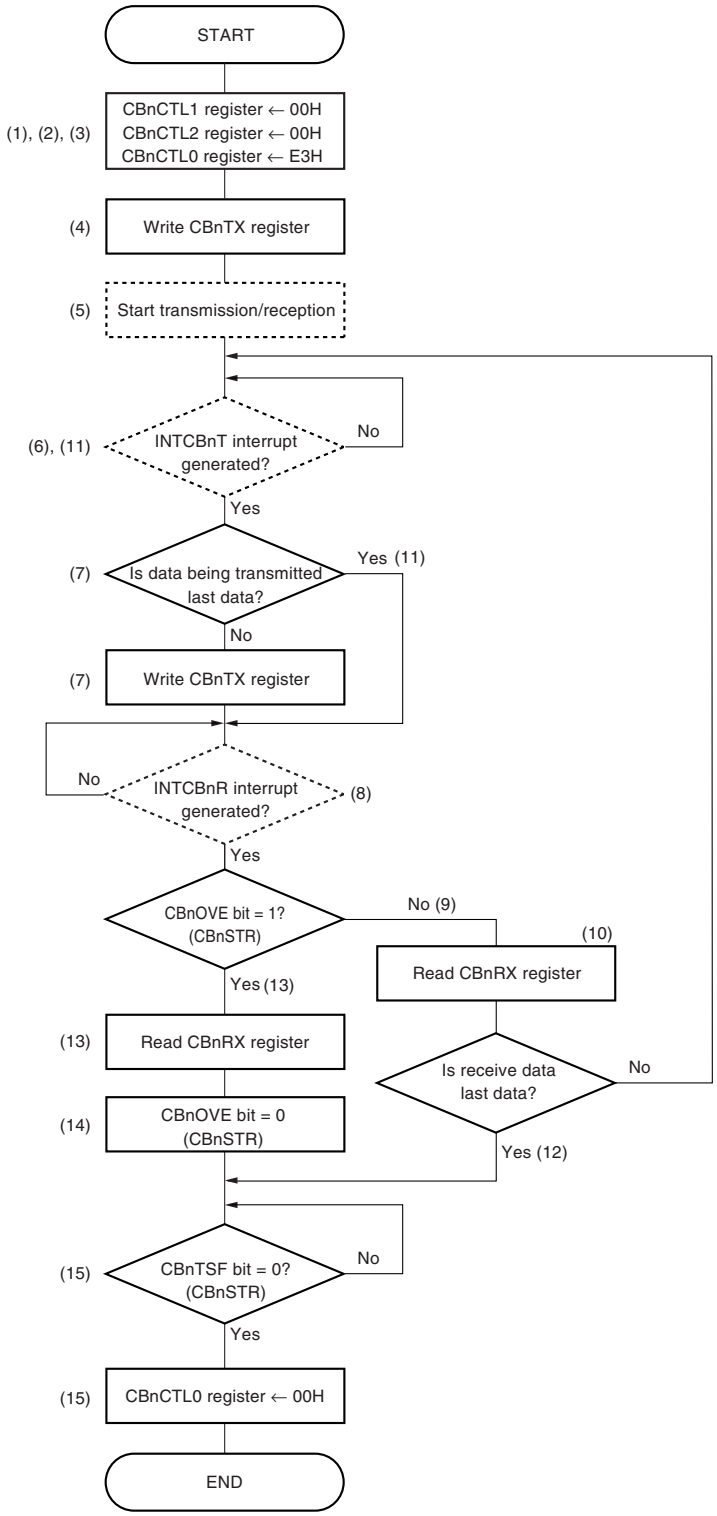
(13) To release the reception enable status, write the CBnCTL0.CBnPWR bit = 0 and the CBnCTL0.CBnRXE bit = 0 after checking that the CBnTSF bit = 0.

Remark n = 0, 1, and 3

<R> **14.6.9 Continuous transfer mode (master mode, transmission/reception mode)**

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = f_{XX}/2 (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 000), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow



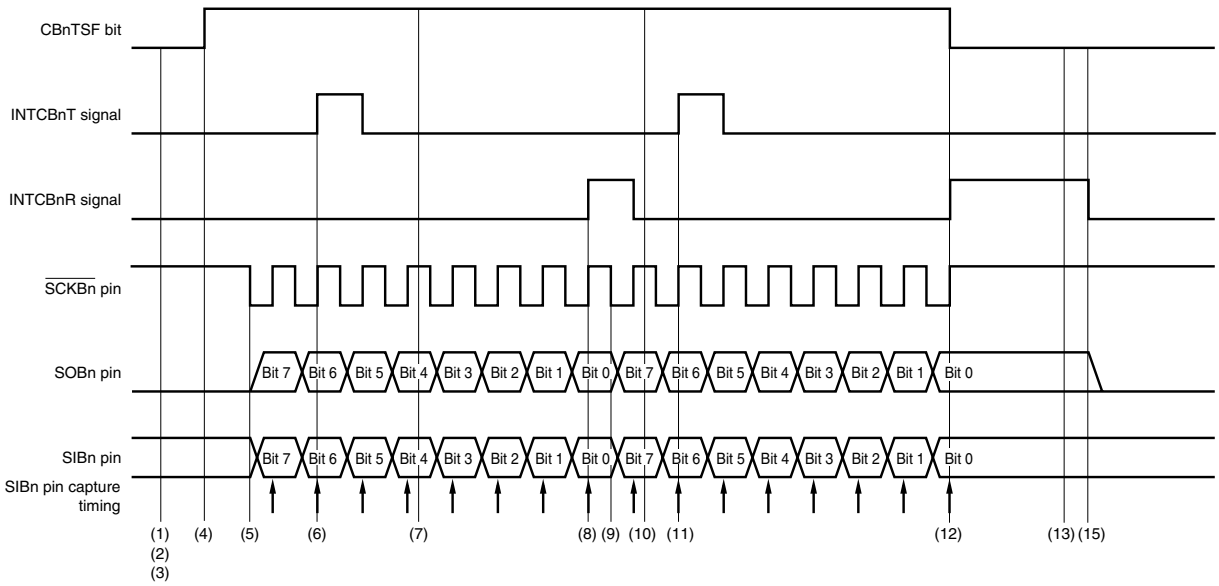
Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing

(1/2)



(1) Write 00H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = f_{XX}/2, and master mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write E3H to the CBnCTL0 register, and select the transmission/reception mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by writing the transmit data to the CBnTX register, and transmission/reception is started.

(5) When transmission/reception is started, output the serial clock to the $\overline{\text{SCKBn}}$ pin, output the transmit data to the SOBn pin in synchronization with the serial clock, and capture the receive data of the SIBn pin.

(6) When transfer of the transmit data from the CBnTX register to the shift register is completed and writing to the CBnTX register is enabled, the transmission enable interrupt request signal (INTCBnT) is generated.

(7) To continue transmission/reception, write the transmit data to the CBnTX register again after the INTCBnT signal is generated.

(8) When one transmission/reception is completed, the reception completion interrupt request signal (INTCBnR) is generated, and reading of the CBnRX register is enabled.

(9) When a new transmit data is written to the CBnTX register before communication completion, the next communication is started following communication completion.

(10) Read the CBnRX register.

Remark n = 0, 1, and 3

(2/2)

(11) The transfer of the transmit data from the CBnTX register to the shift register is completed and the INTCBnT signal is generated. To end continuous transmission/reception with the current transmission/reception, do not write to the CBnTX register.

(12) When the next transmit data is not written to the CBnTX register before transfer completion, stop the serial clock output to the $\overline{\text{SCKBn}}$ pin after transfer completion, and clear the CBnTSF bit to 0.

(13) When the reception error interrupt request signal (INTCBnR) is generated, read the CBnRX register.

(14) If an overrun error occurs, write the CBnSTR.CBnOVE bit = 0, and clear the error flag.

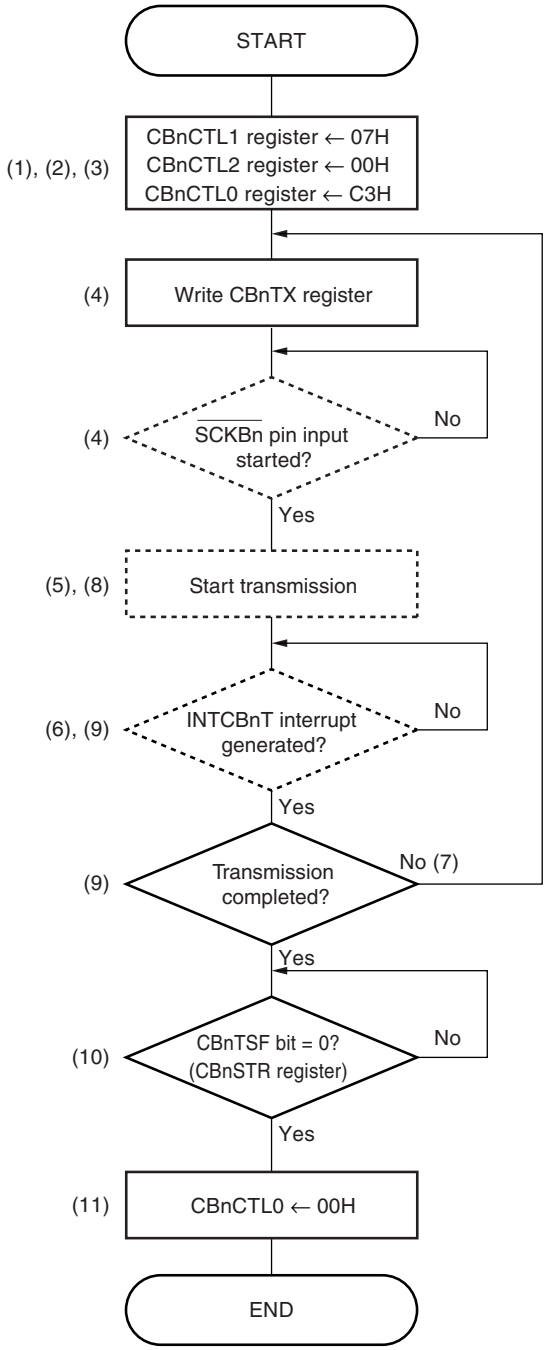
(15) To release the transmission/reception enable status, write the CBnCTL0.CBnPWR bit = 0, the CBnCTL0.CBnTXE bit = 0, and the CBnCTL0.CBnRXE bit = 0 after checking that the CBnTSF bit = 0.

Remark n = 0, 1, and 3

<R> 14.6.10 Continuous transfer mode (slave mode, transmission mode)

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$) (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 111), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

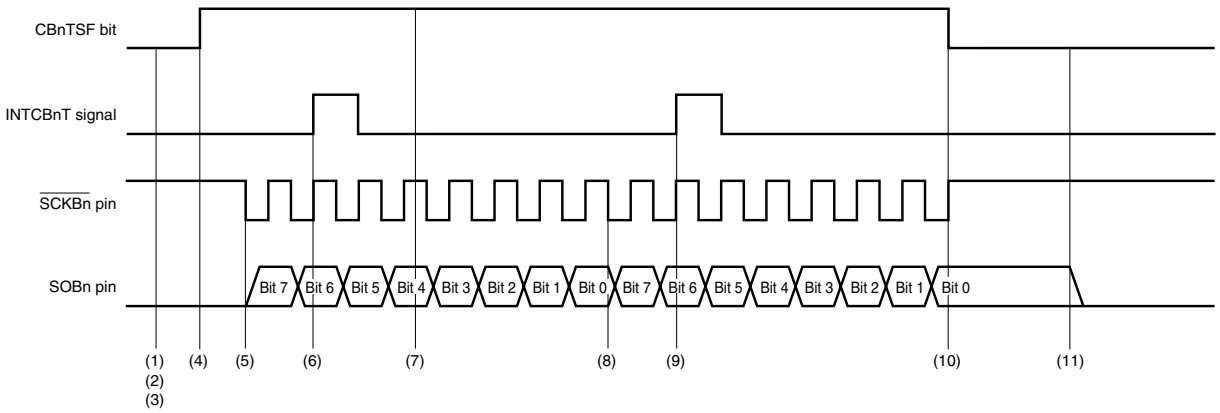


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 07H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$), and slave mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write C3H to the CBnCTL0 register, and select the transmission mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by writing the transmit data to the CBnTX register, and the device waits for a serial clock input.

(5) When a serial clock is input, output the transmit data from the SOBn pin in synchronization with the serial clock.

(6) When transfer of the transmit data from the CBnTX register to the shift register is completed and writing to the CBnTX register is enabled, the transmission enable interrupt request signal (INTCBnT) is generated.

(7) To continue transmission, write the transmit data to the CBnTX register again after the INTCBnT signal is generated.

(8) When a serial clock is input following completion of the transmission of the transfer data length set with the CBnCTL2 register, continuous transmission is started.

(9) When transfer of the transmit data from the CBnTX register to the shift register is completed and writing to the CBnTX register is enabled, the INTCBnT signal is generated. To end continuous transmission with the current transmission, do not write to the CBnTX register.

(10) When the clock of the transfer data length set with the CBnCTL2 register is input without writing to the CBnTX register, clear the CBnTSF bit to 0 to end transmission.

(11) To release the transmission enable status, write the CBnCTL0.CBnPWR bit = 0 and the CBnCTL0.CBnTXE bit = 0 after checking that the CBnTSF bit = 0.

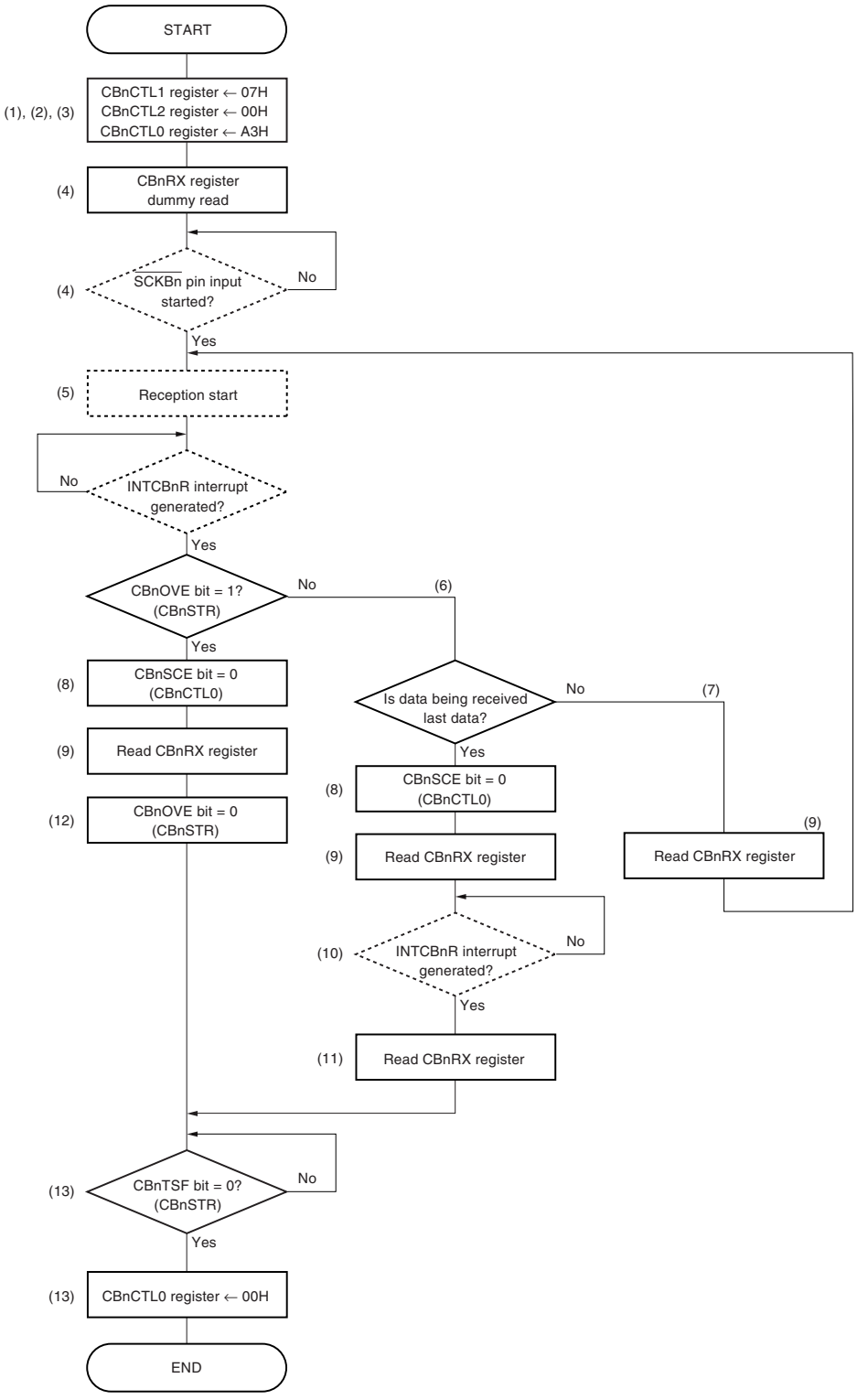
Caution In continuous transmission mode, the reception completion interrupt request signal (INTCBnR) is not generated.

Remark n = 0, 1, and 3

<R> 14.6.11 Continuous transfer mode (slave mode, reception mode)

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$) (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 111), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow

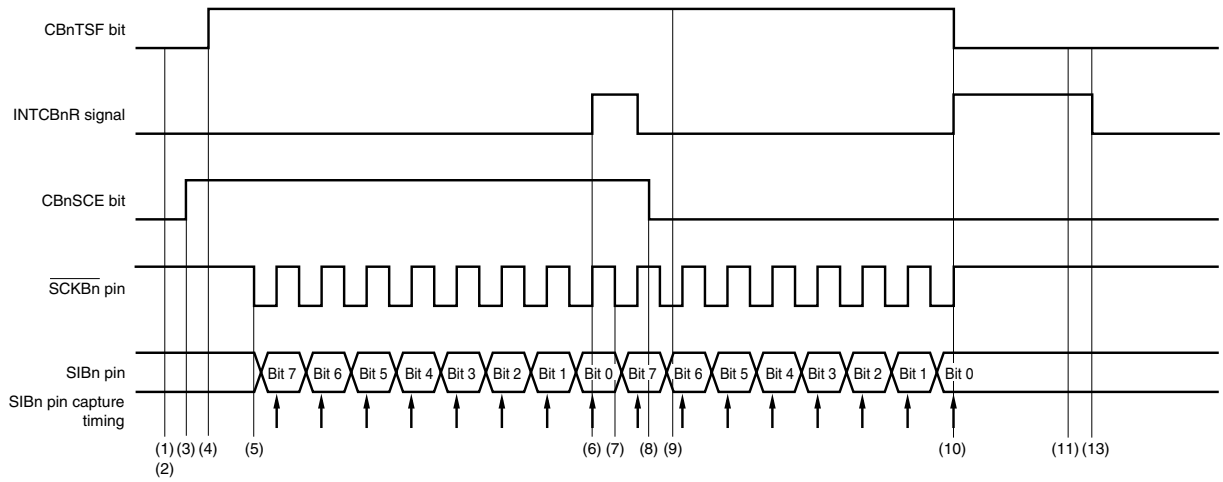


Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing



(1) Write 07H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$), and slave mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write A3H to the CBnCTL0 register, and select the reception mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by performing a dummy read of the CBnRX register, and the device waits for a serial clock input.

(5) When a serial clock is input, capture the receive data of the SIBn pin in synchronization with the serial clock.

(6) When reception is completed, the reception completion interrupt request signal (INTCBnR) is generated, and reading of the CBnRX register is enabled.

(7) When a serial clock is input in the CBnCTL0.CBnSCE bit = 1 status, continuous reception is started.

(8) To end continuous reception with the current reception, write the CBnSCE bit = 0.

(9) Read the CBnRX register.

(10) When reception is completed, the INTCBnR signal is generated, and reading of the CBnRX register is enabled. When the CBnSCE bit = 0 is set before communication completion, clear the CBnTSF bit to 0 to end the receive operation.

(11) Read the CBnRX register.

(12) If an overrun error occurs, write the CBnSTR.CBnOVE bit = 0, and clear the error flag.

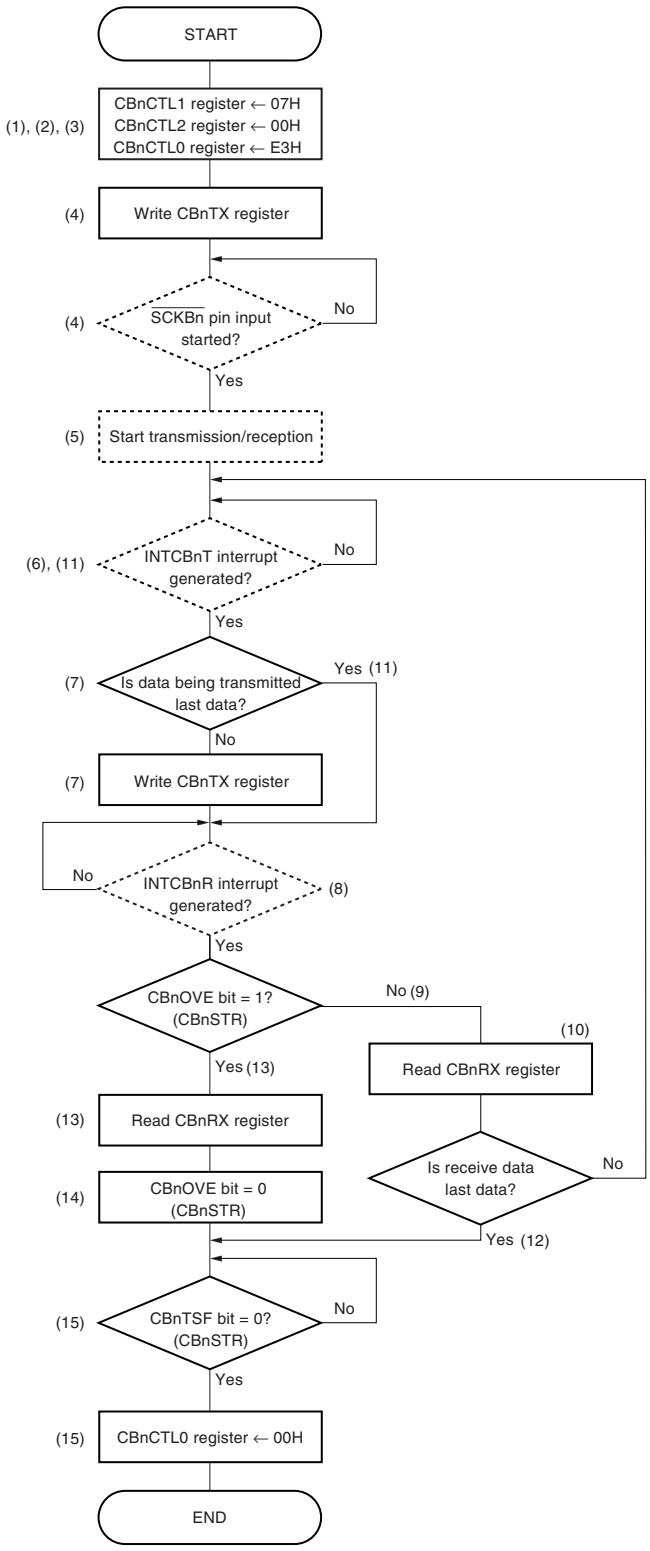
(13) To release the reception enable status, write the CBnCTL0.CBnPWR bit = 0 and the CBnCTL0.CBnRXE bit = 0 after checking that the CBnTSF bit = 0.

Remark n = 0, 1, and 3

<R> **14.6.12 Continuous transfer mode (slave mode, transmission/reception mode)**

MSB first (CBnCTL0.CBnDIR bit = 0), communication type 1 (CBnCTL1.CBnCKP and CBnCTL1.CBnDAP bits = 00), communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$) (CBnCTL1.CBnCKS2 to CBnCTL1.CBnCKS0 bits = 111), transfer data length = 8 bits (CBnCTL2.CBnCL3 to CBnCTL2.CBnCL0 bits = 0000)

(1) Operation flow



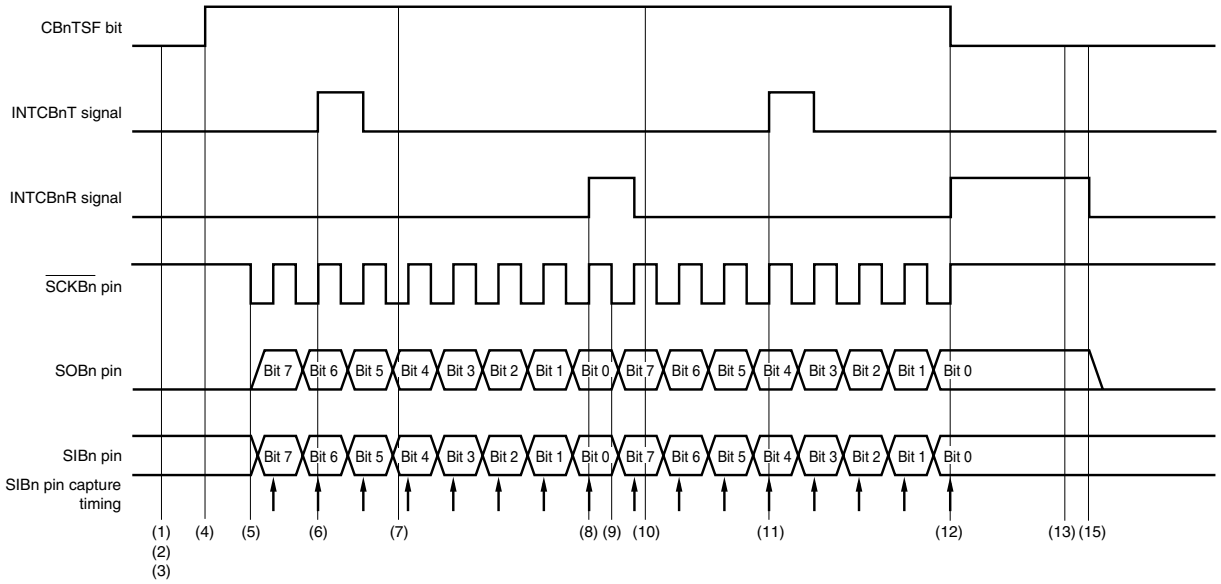
Remarks 1. The broken lines indicate the hardware processing.

2. The numbers in this figure correspond to the processing numbers in **(2) Operation timing**.

3. n = 0, 1, and 3

(2) Operation timing

(1/2)



(1) Write 07H to the CBnCTL1 register, and select communication type 1, communication clock (f_{CCLK}) = external clock ($\overline{SCKBn}$), and slave mode.

(2) Write 00H to the CBnCTL2 register, and set the transfer data length to 8 bits.

(3) Write E3H to the CBnCTL0 register, and select the transmission/reception mode, MSB first, and continuous transfer mode at the same time as enabling the operation of the communication clock (f_{CCLK}).

(4) The CBnSTR.CBnTSF bit is set to 1 by writing the transmit data to the CBnTX register, and the device waits for a serial clock input.

(5) When a serial clock is input, output the transmit data to the SOBn pin in synchronization with the serial clock, and capture the receive data of the SIBn pin.

(6) When transfer of the transmit data from the CBnTX register to the shift register is completed and writing to the CBnTX register is enabled, the transmission enable interrupt request signal (INTCBnT) is generated.

(7) To continue transmission, write the transmit data to the CBnTX register again after the INTCBnT signal is generated.

(8) When reception of the transfer data length set with the CBnCTL2 register is completed, the reception completion interrupt request signal (INTCBnR) is generated, and reading of the CBnRX register is enabled.

(9) When a serial clock is input continuously, continuous transmission/reception is started.

(10) Read the CBnRX register.

(11) When transfer of the transmit data from the CBnTX register to the shift register is completed and writing to the CBnTX register is enabled, the INTCBnT signal is generated. To end continuous transmission/reception with the current transmission/reception, do not write to the CBnTX register.

Remark n = 0, 1, and 3

(2/2)

(12) When the clock of the transfer data length set with the CBnCTL2 register is input without writing to the CBnTX register, the INTCBnR signal is generated. Clear the CBnTSF bit to 0 to end transmission/reception.
(13) When the INTCBnR signal is generated, read the CBnRX register.
(14) If an overrun error occurs, write the CBnSTR.CBnOVE bit = 0, and clear the error flag.
(15) To release the transmission/reception enable status, write the CBnCTL0.CBnPWR bit = 0, the CBnCTL0.CBnTXE bit = 0, and the CBnCTL0.CBnRXE bit = 0 after checking that the CBnTSF bit = 0.

Remark n = 0, 1, and 3

<R>

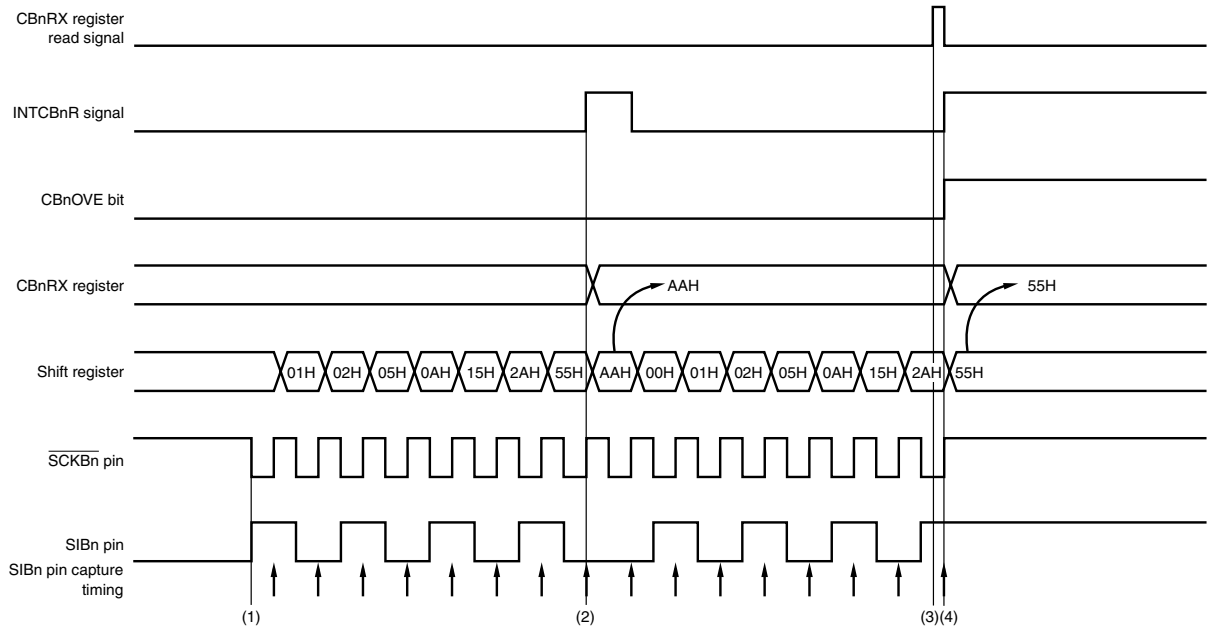
14.6.13 Reception error

When transfer is performed with reception enabled (CBnCTL0.CBnRXE bit = 1) in the continuous transfer mode, the reception completion interrupt request signal (INTCBnR) is generated again when the next receive operation is completed before the CBnRX register is read after the INTCBnR signal is generated, and the overrun error flag (CBnSTR.CBnOVE) is set to 1.

Even if an overrun error has occurred, the previous receive data is lost since the CBnRX register is updated. Even if a reception error has occurred, the INTCBnR signal is generated again upon the next reception completion if the CBnRX register is not read.

To avoid an overrun error, complete reading the CBnRX register until one half clock before sampling the last bit of the next receive data from the INTCBnR signal generation.

(1) Operation timing

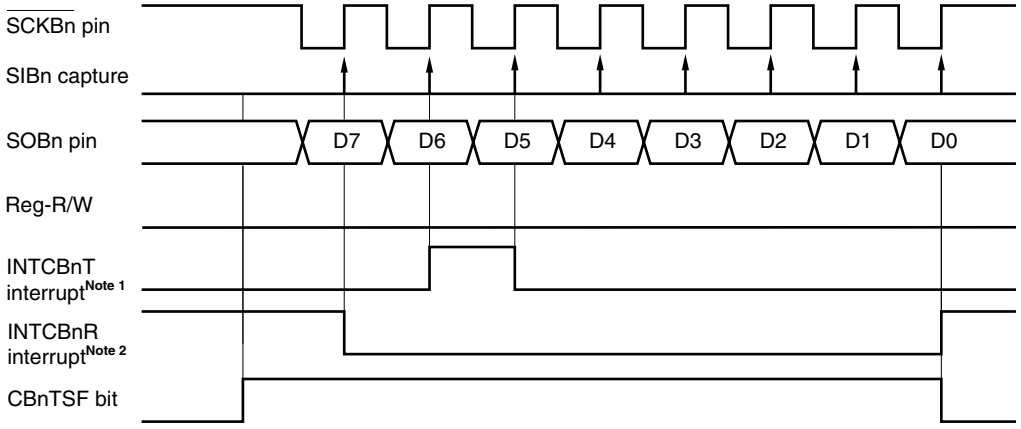


(1) Start continuous transfer.
(2) Completion of the first transfer
(3) The CBnRX register cannot be read until one half clock before the completion of the second transfer.
(4) An overrun error occurs, and the reception completion interrupt request signal (INTCBnR) is generated, and then the overrun error flag (CBnSTR.CBnOVE) is set to 1. The receive data is overwritten.

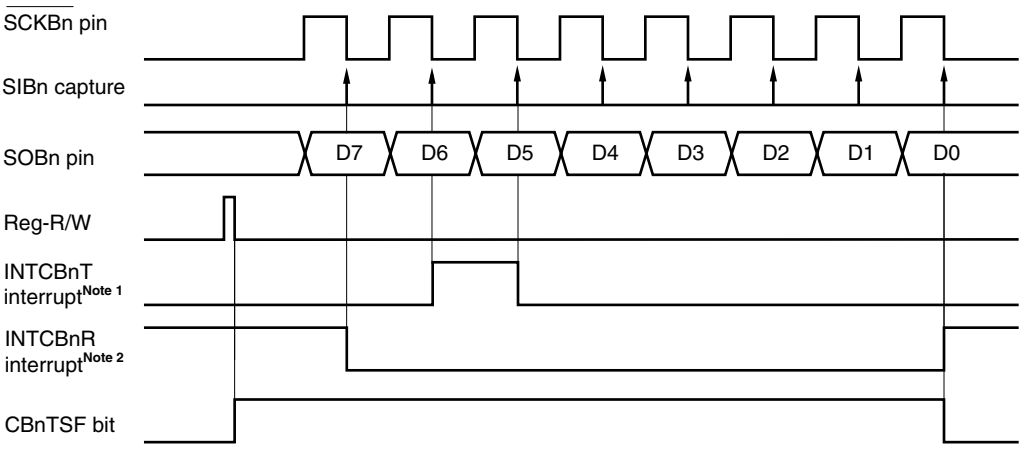
Remark n = 0, 1, and 3

14.6.14 Clock timing

(i) Communication type 1 (CBnCKP and CBnDAP bits = 00)



(ii) Communication type 3 (CBnCKP and CBnDAP bits = 10)



Notes 1. The INTCBnT interrupt is set when the data written to the CBnTX register is transferred to the data shift register in the continuous transmission or continuous transmission/reception mode. In the single transmission or single transmission/reception mode, the INTCBnT interrupt request signal is not generated, but the INTCBnR interrupt request signal is generated upon end of communication.

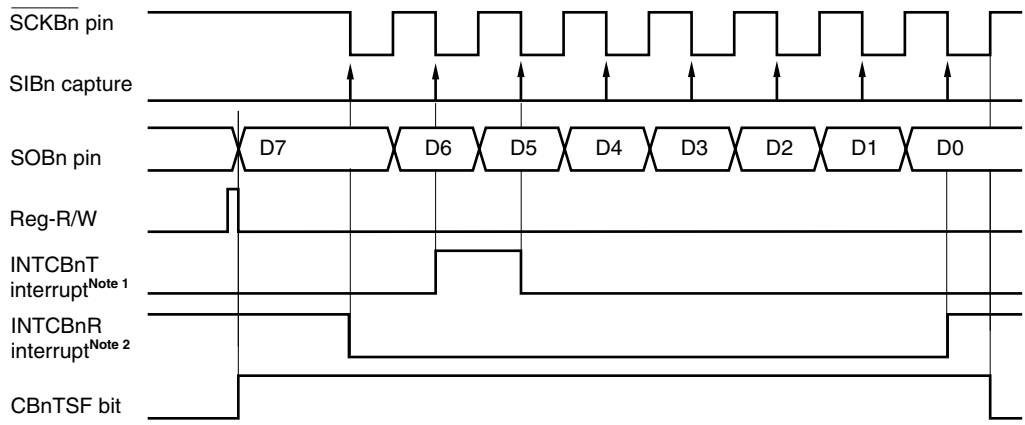
<R> **2.** The INTCBnR interrupt occurs if reception is correctly ended and receive data is ready in the CBnRX register while reception is enabled. In the single mode, the INTCBnR interrupt request signal is generated even in the transmission mode, upon end of communication.

<R> **Caution In single transfer mode, writing to the CBnTX register with the CBnTSF bit set to 1 is ignored. This has no influence on the operation during transfer.**

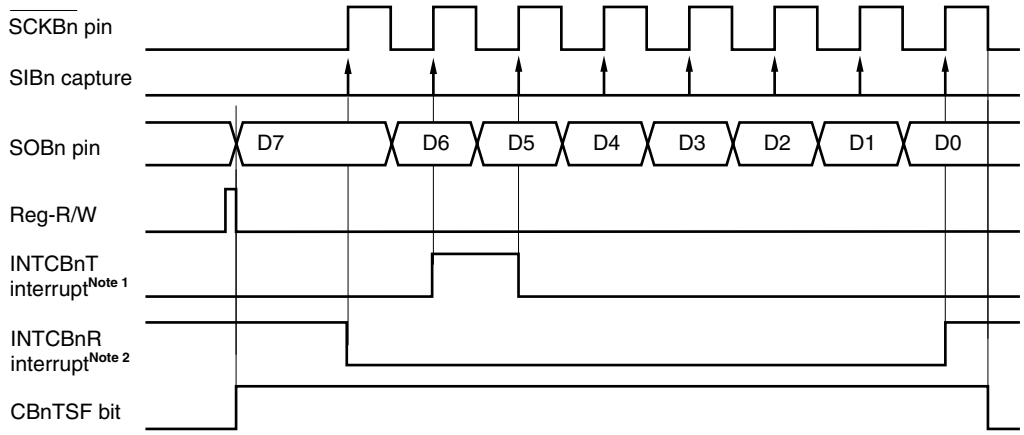
For example, if the next data is written to the CBnTX register when DMA is started by generating the INTCBnR signal (μPD703253Y only), the written data is not transferred because the CBnTSF bit is set to 1.

Use the continuous transfer mode, not the single transfer mode, for such applications.

(iii) Communication type 2 (CBnCKP and CBnDAP bits = 01)



(iv) Communication type 4 (CBnCKP and CBnDAP bits = 11)



Notes 1. The INTCBnT interrupt is set when the data written to the CBnTX register is transferred to the data shift register in the continuous transmission or continuous transmission/reception modes. In the single transmission or single transmission/reception modes, the INTCBnT interrupt request signal is not generated, but the INTCBnR interrupt request signal is generated upon end of communication.

<R> **2.** The INTCBnR interrupt occurs if reception is correctly ended and receive data is ready in the CBnRX register while reception is enabled. In the single mode, the INTCBnR interrupt request signal is generated even in the transmission mode, upon end of communication.

<R> **Caution In single transfer mode, writing to the CBnTX register with the CBnTSF bit set to 1 is ignored. This has no influence on the operation during transfer.**

For example, if the next data is written to the CBnTX register when DMA is started by generating the INTCBnR signal (μPD703253Y only), the written data is not transferred because the CBnTSF bit is set to 1.

Use the continuous transfer mode, not the single transfer mode, for such applications.

14.7 Output Pins

(1) $\overline{\text{SCKBn}}$ pin

When CSIBn operation is disabled (CBnCTL0.CBnPWR bit = 0), the $\overline{\text{SCKBn}}$ pin output status is as follows.

CBnCKP	CBnCKS2	CBnCKS1	CBnCKS0	$\overline{\text{SCKBn}}$ Pin Output
0	1	1	1	High impedance
	Other than above			Fixed to high level
1	1	1	1	High impedance
	Other than above			Fixed to low level

Remarks 1. The output level of the $\overline{\text{SCKBn}}$ pin changes if any of the CBnCTL1.CBnCKP and CBnCKS2 to CBnCKS0 bits is rewritten.

2. n = 0, 1, and 3

(2) SOBn pin

When CSIBn operation is disabled (CBnPWR bit = 0), the SOBn pin output status is as follows.

CBnTXE	CBnDAP	CBnDIR	SOBn Pin Output
0	×	×	Fixed to low level
1	0	×	SOBn latch value (low level)
	1	0	CBnTX0 value (MSB)
		1	CBnTX0 value (LSB)

Remarks 1. The SOBn pin output changes when any one of the CBnCTL0.CBnTXE, CBnCTL0.CBnDIR bits, and CBnCTL1.CBnDAP bit is rewritten.

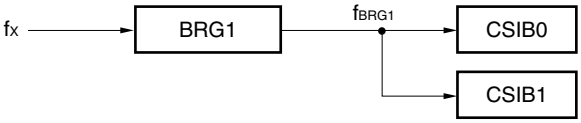
2. ×: Don't care

3. n = 0, 1, and 3

14.8 Baud Rate Generator

CSIB0 and CSIB1 are connected to the BRG1 baud rate generator as shown in the following block diagram.

Caution The output from the baud rate generator cannot be used as the transfer clock for CSIB3.



(1) BRG1 prescaler mode register (PRSM1)

The PRSM1 register controls generation of the baud rate signal for CSIB.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: PRSM1 FFFFF320H

	7	6	5	<4>	3	2	1	0
PRSM1	0	0	0	BGCE1	0	0	BGCS11	BGCS10

BGCE1	Baud rate output
0	Disabled
1	Enabled

BGCS11	BGCS10	Input clock selection (f_{BGCS1})	Setting value (k)
0	0	f_{XX}	0
0	1	$f_{XX}/2$	1
1	0	$f_{XX}/4$	2
1	1	$f_{XX}/8$	3

Cautions 1. Do not rewrite the PRSM1 register during operation.

2. Set the PRSM1 register before setting the BGCE1 bit to 1.

(2) BRG1 prescaler compare register (PRSCM1)

The PRSCM1 register is an 8-bit compare register.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: PRSCM1 FFFFF321H

	7	6	5	4	3	2	1	0
PRSCM1	PRSCM17	PRSCM16	PRSCM15	PRSCM14	PRSCM13	PRSCM12	PRSCM11	PRSCM10

Cautions 1. Do not rewrite the PRSCM1 register during operation.

2. Set the PRSCM1 register before setting the PRSM1.BGCE1 bit to 1.

14.8.1 Baud rate generation

The transmission/reception clock is generated by dividing the main clock. The baud rate generated from the main clock is obtained by the following equation.

$$f_{BRG1} = \frac{f_{XX}}{2^{k+1} \times N}$$

<R> **Caution Set f_{BRG1} to 8 MHz or lower.**

Remark f_{BRG1}: BRG1 count clock

f_{XX}: Main clock frequency

k: PRSM1 register setting value = 0 to 3

N: PRSCM1 register setting value = 1 to 256

However, N = 256 only when PRSCM1 register is set to 00H.

14.9 Cautions

<R> (1) When transferring transmit data and receive data using DMA transfer (μPD703253Y only), error processing cannot be performed even if an overrun error occurs during serial transfer. Check that the no overrun error has occurred by reading the CBnSTR.CBnOVE bit after DMA transfer has been completed.

(2) In regards to registers that are forbidden from being rewritten during operations (CBnCTL0.CBnPWR bit is 1), if rewriting has been carried out by mistake during operations, set the CBnCTL0.CBnPWR bit to 0 once, then initialize CSIBn.

Registers to which rewriting during operation are prohibited are shown below.

- CBnCTL0 register: CBnTXE, CBnRXE, CBnDIR, CBnTMS bits
- CBnCTL1 register: CBnCKP, CBnDAP, CBnCKS2 to CBnCKS0 bits
- CBnCTL2 register: CBnCL3 to CBnCL0 bits

(3) In communication type 2 or 4 (CBnCTL1.CBnDAP bit = 1), the CBnSTR.CBnTSF bit is cleared half a $\overline{\text{SCKBn}}$ clock after occurrence of a reception complete interrupt (INTCBnR).

In the single transfer mode, writing the next transmit data is ignored during communication (CBnTSF bit = 1), and the next communication is not started. Also if reception-only communication (CBnCTL0.CBnTXE bit = 0, CBnCTL0.CBnRXE bit = 1) is set, the next communication is not started even if the receive data is read during communication (CBnTSF bit = 1).

Therefore, when using the single transfer mode with communication type 2 or 4 (CBnDAP bit = 1), pay particular attention to the following.

- To start the next transmission, confirm that CBnTSF bit = 0 and then write the transmit data to the CBnTX register.
- To perform the next reception continuously when reception-only communication (CBnTXE bit = 0, CBnRXE bit = 1) is set, confirm that CBnTSF bit = 0 and then read the CBnRX register.

Or, use the continuous transfer mode instead of the single transfer mode. Use of the continuous transfer mode <R> is recommended especially for using DMA (μPD703253Y only).

Remark n = 0, 1, 3

To use the I²C bus function, set the P38/SDA00, P39/SCL00, P40/SDA01, and P41/SCL01 pins as the serial transmit/receive data I/O pins (SDA00 and SDA01) and serial clock I/O pins (SCL00 and SCL01), and set them to N-ch open-drain output.

15.1 Mode Switching of I²C Bus and Other Serial Interfaces

15.1.1 CSIB0 and I²C01 mode switching

In the V850ES/SG1, CSIB0 and I²C01 are alternate functions of the same pin and therefore cannot be used simultaneously. Set I²C01 in advance, using the PMC4 and PFC4 registers, before use.

Caution The transmit/receive operation of CSIB0 and I²C01 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.

Figure 15-1. CSIB0 and I²C01 Mode Switch Settings

After reset: 00H R/W Address: FFFFF448H

	7	6	5	4	3	2	1	0
PMC4	0	0	0	0	0	PMC42	PMC41	PMC40

After reset: 00H R/W Address: FFFFF468H

	7	6	5	4	3	2	1	0
PFC4	0	0	0	0	0	0	PFC41	PFC40

PMC4n	PFC4n	Operation mode
0	×	Port I/O mode
1	0	CSIB0 mode
1	1	I²C01 mode

Remarks 1. n = 0, 1

2. × = don't care

15.2 Features

I²C00 and I²C01 have the following two modes.

- Operation stopped mode
- I²C (Inter IC) bus mode (multimasters supported)

(1) Operation stopped mode

In this mode, serial transfers are not performed, thus enabling a reduction in power consumption.

(2) I²C bus mode (multimaster support)

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock pin (SCL0n) and a serial data bus pin (SDA0n).

This mode complies with the I²C bus format and the master device can generate “start condition”, “address”, “transfer direction specification”, “data”, and “stop condition” data to the slave device via the serial data bus. The slave device automatically detects the received statuses and data by hardware. This function can simplify the part of an application program that controls the I²C bus.

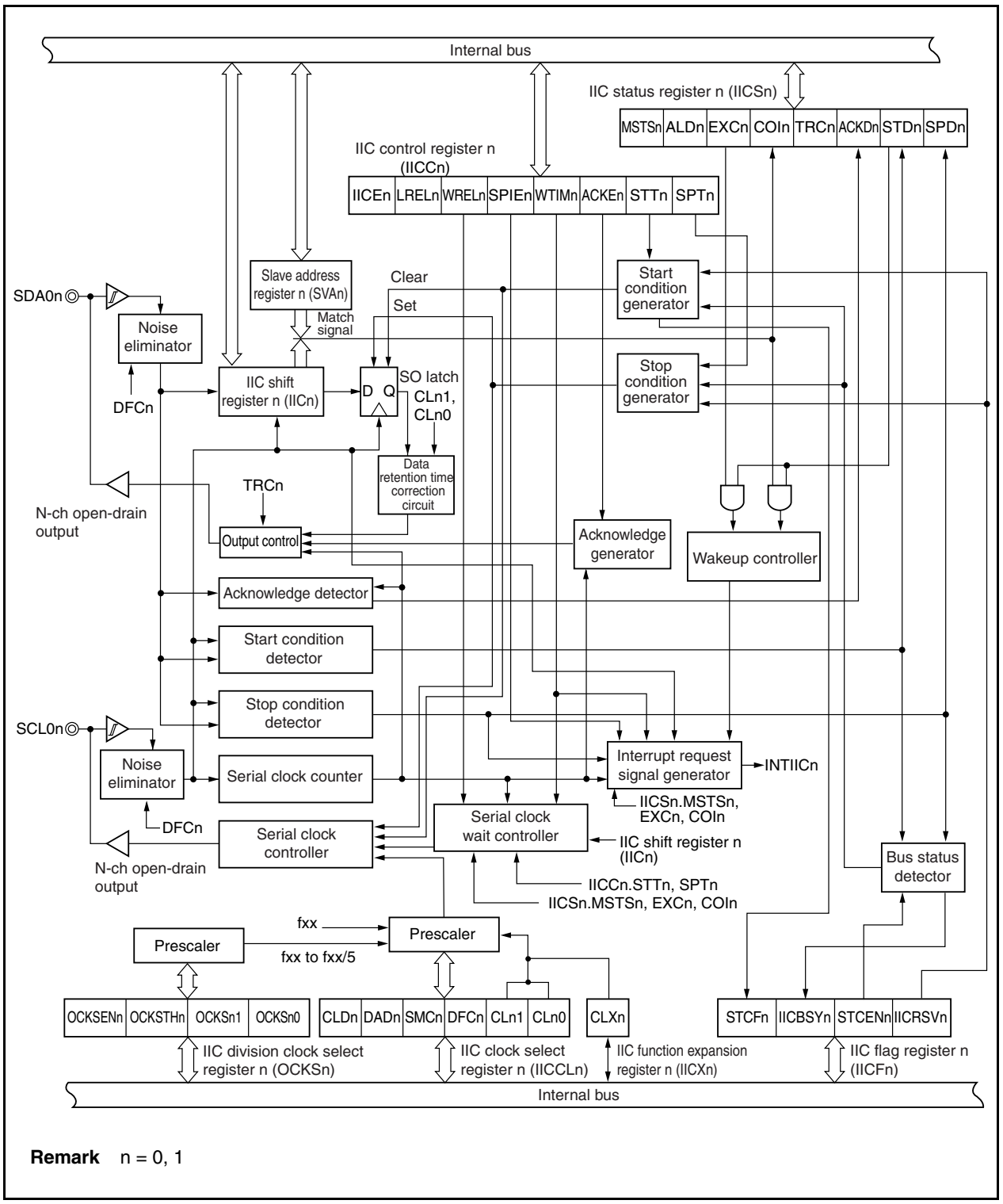
Since SCL0n and SDA0n pins are used for N-ch open-drain outputs, I²C0n requires pull-up resistors for the serial clock line and the serial data bus line.

Remark n = 0, 1

15.3 Configuration

The block diagram of the I²C0n is shown below.

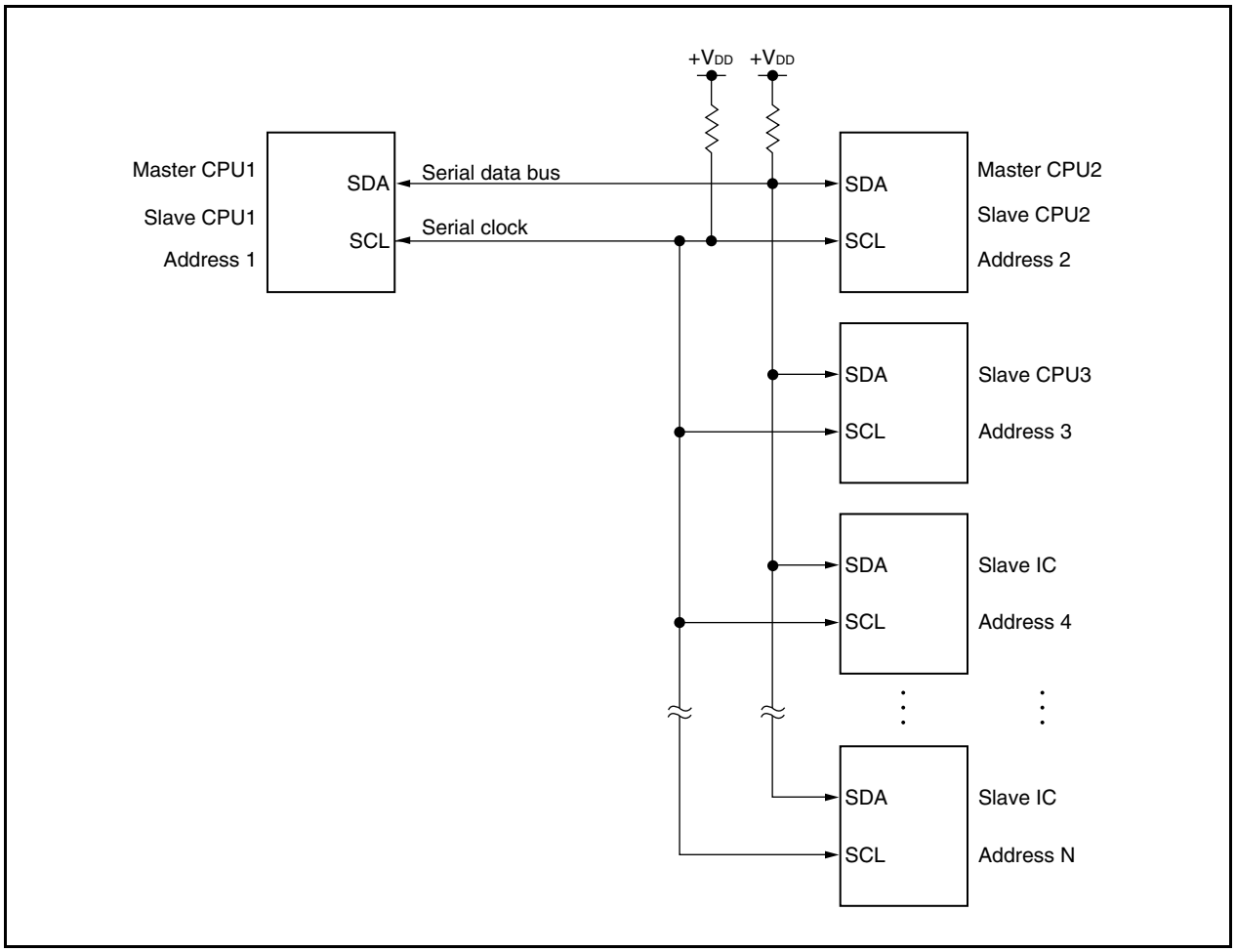
Figure 15-2. Block Diagram of I²C0n



Remark n = 0, 1

A serial bus configuration example is shown below.

Figure 15-3. Serial Bus Configuration Example Using I²C Bus



I²C0n includes the following hardware (n = 0, 1).

Table 15-1. Configuration of I²C0n

Item	Configuration
Registers	IIC shift register n (IICn) Slave address register n (SVAn)
Control registers	IIC control register n (IICCn) IIC status register n (IICSn) IIC flag register n (IICF0n) IIC clock select register n (IICCLn) IIC function expansion register n (IICXn) IIC division clock select registers 0, 1 (OCKS0, OCKS1)

(1) IIC shift register n (IICn)

The IICn register converts 8-bit serial data into 8-bit parallel data and vice versa, and can be used for both transmission and reception (n = 0, 1).

Write and read operations to the IICn register are used to control the actual transmit and receive operations.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

(2) Slave address register n (SVAn)

The SVAn register sets local addresses when in slave mode (n = 0, 1).

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

(3) SO latch

The SO latch is used to retain the output level of the SDA0n pin (n = 0, 1).

(4) Wakeup controller

This circuit generates an interrupt request signal (INTIICn) when the address received by this register matches the address value set to the SVAn register or when an extension code is received (n = 0, 1).

(5) Prescaler

This selects the sampling clock to be used.

(6) Serial clock counter

This counter counts the serial clocks that are output and the serial clocks that are input during transmit/receive operations and is used to verify that 8-bit data was transmitted or received.

(7) Interrupt request signal generator

This circuit controls the generation of interrupt request signals (INTIICn).

An I²C interrupt is generated following either of two triggers.

- Falling edge of eighth or ninth clock of the serial clock (set by IICCn.WTIMn bit)
- Interrupt occurrence due to stop condition detection (set by IICCn.SPIEn bit)

Remark n = 0, 1

(8) Serial clock controller

In master mode, this circuit generates the clock output via the SCL0n pin from the sampling clock (n = 0, 1).

(9) Serial clock wait controller

This circuit controls the wait timing.

(10) $\overline{\text{ACK}}$ generator, stop condition detector, start condition detector, and $\overline{\text{ACK}}$ detector

These circuits are used to generate and detect various statuses.

(11) Data hold time correction circuit

This circuit generates the hold time for data corresponding to the falling edge of the SCL0n pin.

(12) Start condition generator

A start condition is generated when the IICCn.STTn bit is set.

However, in the communication reservation disabled status (IICFn.IICRSVn bit = 1), this request is ignored and the IICFn.STCFn bit is set to 1 if the bus is not released (IICFn.IICBSYn bit = 1).

(13) Stop condition generator

A stop condition is generated when the IICCn.SPTn bit is set (1).

(14) Bus status detector

Whether the bus is released or not is ascertained by detecting a start condition and stop condition.

However, the bus status cannot be detected immediately after operation, so set the bus status detector to the initial status by using the IICFn.STCENn bit.

15.4 Registers

I²C00 and I²C01 are controlled by the following registers.

- IIC control registers 0, 1 (IICC0, IICC1)
- IIC status registers 0, 1 (IICS0, IICS1)
- IIC flag registers 0, 1 (IICF0, IICF1)
- IIC clock select registers 0, 1 (IICCL0, IICCL1)
- IIC function expansion registers 0, 1 (IICX0, IICX1)
- IIC division clock select registers 0, 1 (OCKS0, OCKS1)

The following registers are also used.

- IIC shift registers 0, 1 (IIC0, IIC1)
- Slave address registers 0, 1 (SVA0, SVA1)

Remark For the alternate-function pin settings, see **Table 4-15 Using Port Pin as Alternate-Function Pin**.

(1) IIC control registers 0, 1 (IICC0, IICC1)

The IICCn register enables/stops I²C0n operations, sets the wait timing, and sets other I²C operations (n = 0, 1). This register can be read or written in 8-bit or 1-bit units. However, set the SPIEn, WTIMn, and ACKEn bits when the IICEn bit is 0 or during the wait period. When setting the IICEn bit from “0” to “1”, these bits can also be set at the same time.

Reset sets this register to 00H.

(1/4)

After reset: 00H R/W Address: IICC0 FFFFFD82H, IICC1 FFFFFD92H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICCn	IICEn	LRELn	WRELn	SPIEn	WTIMn	ACKEn	STTn	SPTn

(n = 0, 1)

IICEn	Specification of I²Cn operation enable/disable
0	Operation stopped. IICSn register reset[Note 1]. Internal operation stopped.
1	Operation enabled.
Be sure to set this bit to 1 when the SCL0n and SDA0n lines are high level.	
Condition for clearing (IICEn bit = 0)	Condition for setting (IICEn bit = 1)
• Cleared by instruction • After reset	• Set by instruction

LRELn[Note 2]	Exit from communications
0	Normal operation
1	This exits from the current communication operation and sets standby mode. This setting is automatically cleared after being executed. Its uses include cases in which a locally irrelevant extension code has been received. The SCL0n and SDA0n lines are set to high impedance. The STTn and SPTn bits and the MSTSn, EXCn, COIn, TRCn, ACKDn, and STDn bits of the IICSn register are cleared.
The standby mode following exit from communications remains in effect until the following communication entry conditions are met. • After a stop condition is detected, restart is in master mode. • An address match occurs or an extension code is received after the start condition.	
Condition for clearing (LRELn bit = 0)	Condition for setting (LRELn bit = 1)
• Automatically cleared after execution • After reset	• Set by instruction

WRELn[Note 2]	Wait state cancellation control
0	Wait state not canceled
1	Wait state canceled. This setting is automatically cleared after wait state is canceled.
Condition for clearing (WRELn bit = 0)	Condition for setting (WRELn bit = 1)
• Automatically cleared after execution • After reset	• Set by instruction

Notes 1. The IICSn register, IICFn.STCFn and IICFn.IICBSYn bits, and IICCLn.CLDn and IICCLn.DADn bits are reset.

2. This flag's signal is invalid when the IICEn bit = 0.

Caution If the I²Cn operation is enabled (IICEn bit = 1) when the SCL0n line is high level and the SDA0n line is low level, the start condition is detected immediately. To avoid this, after enabling the I²Cn operation, immediately set the LRELn bit to 1 with a bit manipulation instruction.

Remark The LRELn and WRELn bits are 0 when read after the data has been set.

(2/4)

SPIEn[Note]	Enable/disable generation of interrupt request when stop condition is detected
0	Disabled
1	Enabled

Condition for clearing (SPIEn bit = 0)	Condition for setting (SPIEn bit = 1)
• Cleared by instruction • After reset	• Set by instruction

WTIMn[Note]	Control of wait state and interrupt request generation
0	Interrupt request is generated at the eighth clock's falling edge. Master mode: After output of eight clocks, clock output is set to low level and the wait state is set. Slave mode: After input of eight clocks, the clock is set to low level and the wait state is set for the master device.
1	Interrupt request is generated at the ninth clock's falling edge. Master mode: After output of nine clocks, clock output is set to low level and the wait state is set. Slave mode: After input of nine clocks, the clock is set to low level and the wait state is set for the master device.
During address transfer, an interrupt occurs at the falling edge of the ninth clock regardless of this bit setting. This bit setting becomes valid when the address transfer is completed. In master mode, a wait state is inserted at the falling edge of the ninth clock during address transfer. For a slave device that has received a local address, a wait state is inserted at the falling edge of the ninth clock after $\overline{ACK}$ is generated. When the slave device has received an extension code, however, a wait state is inserted at the falling edge of the eighth clock.	

Condition for clearing (WTIMn bit = 0)	Condition for setting (WTIMn bit = 1)
• Cleared by instruction • After reset	• Set by instruction

ACKEn[Note]	Acknowledgment control
0	Acknowledgment disabled.
1	Acknowledgment enabled. During the ninth clock period, the SDA0n line is set to low level.
The ACKEn bit setting is invalid for address reception. In this case, $\overline{ACK}$ is generated when the addresses match. However, the ACKEn bit setting is valid for reception of the extension code.	

<R>

Condition for clearing (ACKEn bit = 0)	Condition for setting (ACKEn bit = 1)
• Cleared by instruction • After reset	• Set by instruction

Note This flag's signal is invalid when the IICEn bit = 0.

Remark n = 0, 1

(3/4)

STTn	Start condition trigger
0	Start condition is not generated.
1	When bus is released (in STOP mode): A start condition is generated (for starting as master). The SDA0n line is changed from high level to low level while the SCLn line is high level and then the start condition is generated. Next, after the rated amount of time has elapsed, the SCL0n line is changed to low level (wait state). During communication with a third party: • If the communication reservation function is enabled (IICFn.IICRSVn bit = 0) This trigger functions as a start condition reserve flag. When set to 1, it releases the bus and then automatically generates a start condition. • If the communication reservation function is disabled (IICRSVn = 1) The IICFn.STCFn bit is set to 1 and information set (1) to the STTn bit is cleared. This trigger does not generate a start condition. In the wait state (when master device): A restart condition is generated after the wait state is released.

Cautions concerning set timing

For master reception: Cannot be set to 1 during transfer. Can be set to 1 only when the ACKEn bit has been set to 0 and the slave has been notified of final reception.

For master transmission: A start condition cannot be generated normally during the $\overline{\text{ACK}}$ period. Set to 1 during the wait period that follows output of the ninth clock.

For slave: Even when the communication reservation function is disabled (IICRSVn bit = 1), the communication reservation status is entered.

- Setting to 1 at the same time as the SPTn bit is prohibited.
- When the STTn bit is set to 1, setting the STTn bit to 1 again is disabled until the setting is cleared to 0.

Condition for clearing (STTn bit = 0)	Condition for setting (STTn bit = 1)
• When the STTn bit is set to 1 in the communication reservation disabled status • Cleared by loss in arbitration • Cleared by start condition generation in the master device • When the LRELn bit = 1 (communication save) • When the IICEn bit = 0 (operation stop) • After reset	• Set by instruction

Remarks 1. The STTn bit is 0 if it is read immediately after data setting.

2. n = 0, 1

(4/4)

SPTn	Stop condition trigger
0	Stop condition is not generated.
1	Stop condition is generated (termination of master device's transfer). After the SDA0n line goes to low level, either set the SCL0n line to high level or wait until the SCL0n pin goes to high level. Next, after the rated amount of time has elapsed, the SDA0n line is changed from low level to high level and a stop condition is generated.

Cautions concerning set timing

For master reception: Cannot be set to 1 during transfer.
Can be set to 1 only when the ACKEn bit has been set to 0 and during the wait period after the slave has been notified of final reception.

For master transmission: A stop condition cannot be generated normally during the $\overline{\text{ACK}}$ reception period. Set to 1 during the wait period that follows output of the ninth clock.

- Cannot be set to 1 at the same time as the STTn bit.
- The SPTn bit can be set to 1 only when in master mode[Note].
- When the WTIMn bit has been set to 0, if the SPTn bit is set to 1 during the wait period that follows output of eight clocks, note that a stop condition will be generated during the high-level period of the ninth clock.
  The WTIMn bit should be changed from 0 to 1 during the wait period following output of eight clocks, and the SPTn bit should be set to 1 during the wait period that follows output of the ninth clock.
- When the SPTn bit is set to 1, setting the SPTn bit to 1 again is disabled until the setting is cleared to 0.

Condition for clearing (SPTn bit = 0)	Condition for setting (SPTn bit = 1)
• Cleared by loss in arbitration • Automatically cleared after stop condition is detected • When the LRELn bit = 1 (communication save) • When the IICEn bit = 0 (operation stop) • After reset	• Set by instruction

Note Set the SPTn bit to 1 only in master mode. However, when the IICRSVn bit is 0, the SPTn bit must be set to 1 and a stop condition generated before the first stop condition is detected following the switch to the operation enabled status. For details, see **15.15 Cautions**.

Caution When the TRCn bit = 1, the WRELn bit is set to 1 during the ninth clock and the wait state is canceled, after which the TRCn bit is cleared to 0 and the SDA0n line is set to high impedance.

Remarks 1. The SPTn bit is 0 if it is read immediately after data setting.

2. n = 0, 1

(2) IIC status registers 0, 1 (IICS0, IICS1)

The IICSn register indicates the status of I²C0n (n = 0, 1).

This register is read-only, in 8-bit or 1-bit units. However, the IICSn register can only be read when the IICCn.STTn bit is 1 or during the wait period.

Reset sets this register to 00H.

Caution Accessing the IICSn register is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

(1/3)

After reset: 00H R Address: IICS0 FFFFFD86H, IICS1 FFFFFD96H

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICSn	MSTSn	ALDn	EXCn	COIn	TRCn	ACKDn	STDn	SPDn

(n = 0, 1)

MSTSn	Master device status
0	Slave device status or communication standby status
1	Master device communication status

Condition for clearing (MSTSn bit = 0)	Condition for setting (MSTSn bit = 1)
• When a stop condition is detected • When the ALDn bit = 1 (arbitration loss) • Cleared by LRELn bit = 1 (communication save) • When the IICEn bit changes from 1 to 0 (operation stop) • After reset	• When a start condition is generated

ALDn	Arbitration loss detection
0	This status means either that there was no arbitration or that the arbitration result was a “win”.
1	This status indicates the arbitration result was a “loss”. The MSTSn bit is cleared to 0.

Condition for clearing (ALDn bit = 0)	Condition for setting (ALDn bit = 1)
• Automatically cleared after the IICSn register is read[Note] • When the IICEn bit changes from 1 to 0 (operation stop) • After reset	• When the arbitration result is a “loss”.

EXCn	Detection of extension code reception
0	Extension code was not received.
1	Extension code was received.

Condition for clearing (EXCn bit = 0)	Condition for setting (EXCn bit = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by LRELn bit = 1 (communication save) • When the IICEn bit changes from 1 to 0 (operation stop) • After reset	• When the higher four bits of the received address data are either “0000” or “1111” (set at the rising edge of the eighth clock).

Note This bit is also cleared when a bit manipulation instruction is executed for another bit in the IICSn register.

(2/3)

COIn	Matching address detection
0	Addresses do not match.
1	Addresses match.

Condition for clearing (COIn bit = 0)	Condition for setting (COIn bit = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by LRELn bit = 1 (communication save) • When the IICEn bit changes from 1 to 0 (operation stop) • After reset	• When the received address matches the local address (SVAn register) (set at the rising edge of the eighth clock).

TRCn	Transmit/receive status detection
0	Receive status (other than transmit status). The SDA0n line is set to high impedance.
1	Transmit status. The value in the SO latch is enabled for output to the SDA0n line (valid starting at the falling edge of the first byte's ninth clock).

Condition for clearing (TRCn bit = 0)	Condition for setting (TRCn bit = 1)
• When a stop condition is detected • Cleared by LRELn bit = 1 (communication save) • When the IICEn bit changes from 1 to 0 (operation stop) • Cleared by IICCn.WRELn bit = 1[Note] • When the ALDn bit changes from 0 to 1 (arbitration loss) • After reset Master • When "1" is output to the first byte's LSB (transfer direction specification bit) Slave • When a start condition is detected When not used for communication	Master • When a start condition is generated • When "0" is output to the first byte's LSB (transfer direction specification bit) Slave • When "1" is input by the first byte's LSB (transfer direction specification bit)

ACKDn	$\overline{\text{ACK}}$ detection
0	$\overline{\text{ACK}}$ was not detected.
1	$\overline{\text{ACK}}$ was detected.

Condition for clearing (ACKDn bit = 0)	Condition for setting (ACKD bit = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock • Cleared by LRELn bit = 1 (communication save) • When the IICEn bit changes from 1 to 0 (operation stop) • After reset	• After the SDA0n bit is set to low level at the rising edge of the SCL0n pin's ninth clock

Note The TRCn bit is cleared to 0 and SDA0n line becomes high impedance when the WRELn bit is set to 1 and the wait state is canceled to 0 at the ninth clock by TRCn bit = 1.

Remark n = 0, 1

(3/3)

STDn	Start condition detection
0	Start condition was not detected.
1	Start condition was detected. This indicates that the address transfer period is in effect

Condition for clearing (STDn bit = 0)	Condition for setting (STDn bit = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock following address transfer • Cleared by LRELn bit = 1 (communication save) • When the IICEn bit changes from 1 to 0 (operation stop) • After reset	• When a start condition is detected

SPDn	Stop condition detection
0	Stop condition was not detected.
1	Stop condition was detected. The master device's communication is terminated and the bus is released.

Condition for clearing (SPDn bit = 0)	Condition for setting (SPDn bit = 1)
• At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition • When the IICEn bit changes from 1 to 0 (operation stop) • After reset	• When a stop condition is detected

Remark n = 0, 1

(3) IIC flag registers 0, 1 (IICF0, IICF1)

The IICFn register sets the I²C0n operation mode and indicates the I²C bus status.

This register can be read or written in 8-bit or 1-bit units. However, the STCFn and IICBSYn bits are read-only.

IICRSVn enables/disables the communication reservation function (see **15.14 Communication Reservation**).

The initial value of the IICBSYn bit is set by using the STCENn bit (see **15.15 Cautions**).

The IICRSVn and STCENn bits can be written only when operation of I²C0n is disabled (IICCn.IICEn bit = 0).

After operation is enabled, IICFn can be read (n = 0, 1).

Reset sets this register to 00H.

After reset: 00H R/W[Note] Address: IICF0 FFFFFD8AH, IICF1 FFFFFD9AH

	<7>	<6>	5	4	3	2	<1>	<0>
IICFn	STCFn	IICBSYn	0	0	0	0	STCENn	IICRSVn

(n = 0, 1)

STCFn	STTn bit clear
0	Start condition issued
1	Start condition cannot be issued, STTn bit cleared

Condition for clearing (STCFn bit = 0)	Condition for setting (STCFn bit = 1)
• Cleared by IICCn.STTn bit = 1 • When the IICCn.IICEn bit = 0 • After reset	• When start condition is not issued and STTn flag is cleared to 0 during communication reservation is disabled (IICRSVn bit = 1).

IICBSYn	I²C0n bus status
0	Bus released status (default communication status when STCENn bit = 1)
1	Bus communication status (default communication status when STCENn bit = 0)

Condition for clearing (IICBSYn bit = 0)	Condition for setting (IICBSYn bit = 1)
• When stop condition is detected • When the IICEn bit = 0 • After reset	• When start condition is detected • By setting the IICEn bit when the STCENn bit = 0

STCENn	Initial start enable trigger
0	Start conditions cannot be generated until a stop condition is detected following operation enable (IICEn bit = 1).
1	Start conditions can be generated even if a stop condition is not detected following operation enable (IICEn bit = 1).

Condition for clearing (STCENn bit = 0)	Condition for setting (STCENn bit = 1)
• When start condition is detected • After reset	• Setting by instruction

IICRSVn	Communication reservation function disable bit
0	Communication reservation enabled
1	Communication reservation disabled

Condition for clearing (IICRSVn bit = 0)	Condition for setting (IICRSVn bit = 1)
• Clearing by instruction • After reset	• Setting by instruction

Note Bits 6 and 7 are read-only bits.

Cautions 1. Write the STCENn bit only when operation is stopped (IICEn bit = 0).

2. When the STCENn bit = 1, the bus released status (IICBSYn bit = 0) is recognized regardless of the actual bus status immediately after the I²Cn bus operation is enabled. Therefore, to issue the first start condition (STTn bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

3. Write the IICRSVn bit only when operation is stopped (IICEn bit = 0).

(4) IIC clock select registers 0, 1 (IICCL0, IICCL1)

The IICCLn register sets the transfer clock for I²C0n.

This register can be read or written in 8-bit or 1-bit units. However, the CLDn and DADn bits are read-only.

Set the IICCLn register when the IICCn.IICEn bit = 0.

The SMCn, CLn1, and CLn0 bits are set by the combination of the IICXn.CLXn bit and the OCKSTHn, OCKSn1, and OCKSn0 bits of the OCKSn register (see **15.4 (6) I²C0n transfer clock setting method**) (n = 0, 1).

Reset sets this register to 00H.

After reset: 00H R/W[Note] Address: IICCL0 FFFFFD84H, IICCL1 FFFFFD94H

	7	6	<5>	<4>	3	2	1	0
IICCLn	0	0	CLDn	DADn	SMCn	DFCn	CLn1	CLn0

(n = 0, 1)

CLDn	Detection of SCL0n pin level (valid only when IICCn.IICEn bit = 1)
0	The SCL0n pin was detected at low level.
1	The SCL0n pin was detected at high level.

Condition for clearing (CLDn bit = 0)	Condition for setting (CLDn bit = 1)
• When the SCL0n pin is at low level • When the IICEn bit = 0 (operation stop) • After reset	• When the SCL0n pin is at high level

DADn	Detection of SDA0n pin level (valid only when IICEn bit = 1)
0	The SDA0n pin was detected at low level.
1	The SDA0n pin was detected at high level.

Condition for clearing (DADn bit = 0)	Condition for setting (DAD0n bit = 1)
• When the SDA0n pin is at low level • When the IICEn bit = 0 (operation stop) • After reset	• When the SDA0n pin is at high level

SMCn	Operation mode switching
0	Operation in standard mode.
1	Operation in high-speed mode.

DFCn	Digital filter operation control
0	Digital filter off.
1	Digital filter on.
The digital filter can be used only in high-speed mode. In high-speed mode, the transfer clock does not vary regardless of the DFCn bit setting (on/off). The digital filter is used to eliminate noise in high-speed mode.	

Note Bits 4 and 5 are read-only bits.

Caution Be sure to clear bits 7 and 6 to "0".

Remark When the IICCn.IICEn bit = 0, 0 is read when reading the CLDn and DADn bits.

(5) IIC function expansion registers 0, 1 (IICX0, IICX1)

The IICXn register sets I²C0n function expansion (valid only in the high-speed mode).

This register can be read or written in 8-bit or 1-bit units.

Setting of the CLXn bit is performed in combination with the SMCn, CLn1, and CLn0 bits of the IICCLn register and the OCKSTHn, OCKSn1, and OCKSn0 bits of the OCKSn register (see **15.4 (6) I²C0n transfer clock setting method**) (n = 0, 1).

Set the IICXn register when the IICCn.IICEn bit = 0.

Reset sets this register to 00H.

After reset: 00H R/W Address: IICX0 FFFFFD85H, IICX1 FFFFFD95H

	7	6	5	4	3	2	1	<0>
IICXn	0	0	0	0	0	0	0	CLXn

(n = 0, 1)

(6) I²C0n transfer clock setting method

The I²C0n transfer clock frequency (f_{SCL}) is calculated using the following expression (n = 0, 1).

$$f_{SCL} = 1/(m \times T + t_R + t_F)$$

m = 12, 18, 24, 36, 44, 48, 54, 60, 66, 72, 86, 88, 96, 132, 172, 176, 198, 220, 258, 344 (see **Table 15-2 Clock Settings**).

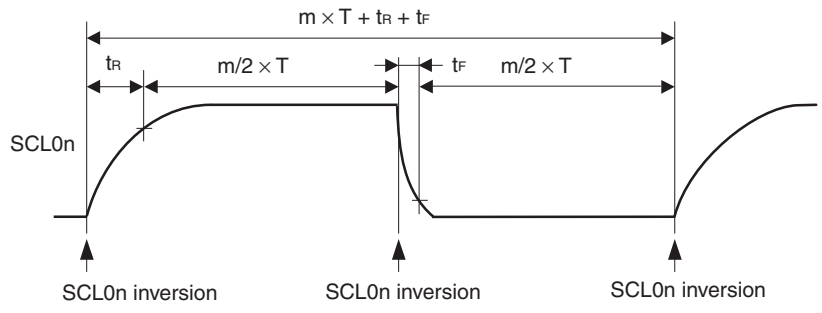
T: $1/f_{XX}$

t_R: SCL0n pin rise time

t_F: SCL0n pin fall time

For example, the I²C0n transfer clock frequency (f_{SCL}) when f_{XX} = 19.2 MHz, m = 198, t_R = 200 ns, and t_F = 50 ns is calculated using following expression.

$$f_{SCL} = 1/(198 \times 52 \text{ ns} + 200 \text{ ns} + 50 \text{ ns}) \cong 94.7 \text{ kHz}$$



The clock to be selected can be set by the combination of the SMCn, CLn1, and CLn0 bits of the IICCLn register, the CLXn bit of the IICXn register, and the OCKSTHn, OCKSn1, and OCKSn0 bits of the OCKSn register (n = 0, 1).

Table 15-2. Clock Settings

IICXn	IICCLn			Selection Clock	Transfer Clock	Settable Main Clock Frequency (f_{XX}) Range	Operating Mode
Bit 0	Bit 3	Bit 1	Bit 0				
CLXn	SMCn	CLn1	CLn0				
0	0	0	0	f_{XX} (when OCKSn = 18H set)	$f_{XX}/44$	2.00 MHz ≤ f_{XX} ≤ 4.19 MHz	Standard mode (SMCn bit = 0)
				$f_{XX}/2$ (when OCKSn = 10H set)	$f_{XX}/88$	4.00 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/3$ (when OCKSn = 11H set)	$f_{XX}/132$	6.00 MHz ≤ f_{XX} ≤ 12.57 MHz	
				$f_{XX}/4$ (when OCKSn = 12H set)	$f_{XX}/176$	8.00 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/5$ (when OCKSn = 13H set)	$f_{XX}/220$	10.00 MHz ≤ f_{XX} ≤ 20.00 MHz	
0	0	0	1	f_{XX} (when OCKSn = 18H set)	$f_{XX}/86$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/2$ (when OCKSn = 10H set)	$f_{XX}/172$	8.38 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/3$ (when OCKSn = 11H set)	$f_{XX}/258$	12.57 MHz ≤ f_{XX} ≤ 20.00 MHz	
				$f_{XX}/4$ (when OCKSn = 12H set)	$f_{XX}/344$	16.76 MHz ≤ f_{XX} ≤ 20.00 MHz	
0	0	1	0	f_{XX} Note	$f_{XX}/86$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	
0	0	1	1	f_{XX} (when OCKSn = 18H set)	$f_{XX}/66$	6.40 MHz	
				$f_{XX}/2$ (when OCKSn = 10H set)	$f_{XX}/132$	12.80 MHz	
				$f_{XX}/3$ (when OCKSn = 11H set)	$f_{XX}/198$	19.20 MHz	
0	1	0	×	f_{XX} (when OCKSn = 18H set)	$f_{XX}/24$	4.19 MHz ≤ f_{XX} ≤ 8.38 MHz	High-speed mode (SMCn bit = 1)
				$f_{XX}/2$ (when OCKSn = 10H set)	$f_{XX}/48$	8.00 MHz ≤ f_{XX} ≤ 16.76 MHz	
				$f_{XX}/3$ (when OCKSn = 11H set)	$f_{XX}/72$	12.00 MHz ≤ f_{XX} ≤ 20.00 MHz	
				$f_{XX}/4$ (when OCKSn = 12H set)	$f_{XX}/96$	16.00 MHz ≤ f_{XX} ≤ 20.00 MHz	
0	1	1	0	f_{XX} Note	$f_{XX}/24$	4.00 MHz ≤ f_{XX} ≤ 8.38 MHz	
0	1	1	1	f_{XX} (when OCKSn = 18H set)	$f_{XX}/18$	6.40 MHz	
				$f_{XX}/2$ (when OCKSn = 10H set)	$f_{XX}/36$	12.80 MHz	
				$f_{XX}/3$ (when OCKSn = 11H set)	$f_{XX}/54$	19.20 MHz	
1	1	0	×	f_{XX} (when OCKSn = 18H set)	$f_{XX}/12$	4.00 MHz ≤ f_{XX} ≤ 4.19 MHz	
				$f_{XX}/2$ (when OCKSn = 10H set)	$f_{XX}/24$	8.00 MHz ≤ f_{XX} ≤ 8.38 MHz	
				$f_{XX}/3$ (when OCKSn = 11H set)	$f_{XX}/36$	12.00 MHz ≤ f_{XX} ≤ 12.57 MHz	
				$f_{XX}/4$ (when OCKSn = 12H set)	$f_{XX}/48$	16.00 MHz ≤ f_{XX} ≤ 16.67 MHz	
				$f_{XX}/5$ (when OCKSn = 13H set)	$f_{XX}/60$	20.00 MHz	
1	1	1	0	f_{XX} Note	$f_{XX}/12$	4.00 MHz ≤ f_{XX} ≤ 4.19 MHz	
Other than above				Setting prohibited	–	–	–

Note Since the selection clock is f_{XX} regardless of the value set to the OCKSn register, clear the OCKSn register to 00H (I²C division clock stopped status).

Remarks 1. n = 0, 1

2. ×: don't care

(7) IIC division clock select registers 0, 1 (OCKS0, OCKS1)

The OCKSn register controls the I²C0n division clock (n = 0, 1).

This register controls the I²C00 division clock via the OCKS0 register and the I²C01 division clock via the OCKS1 register.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: OCKS0 FFFFF340H, OCKS1 FFFFF344H

	7	6	5	4	3	2	1	0
OCKSn	0	0	0	OCKSENn	OCKSTHn	0	OCKSn1	OCKSn0

(n = 0, 1)

OCKSENn	Operation setting of I²C division clock
0	Disable I²C division clock operation
1	Enable I²C division clock operation

OCKSTHn	OCKSn1	OCKSn0	Selection of I²C division clock
0	0	0	$f_{XX}/2$
0	0	1	$f_{XX}/3$
0	1	0	$f_{XX}/4$
0	1	1	$f_{XX}/5$
1	0	0	f_{XX}
Other than above			Setting prohibited

(8) IIC shift registers 0, 1 (IIC0, IIC1)

The IICn register is used for serial transmission/reception (shift operations) synchronized with the serial clock. This register can be read or written in 8-bit units, but data should not be written to the IICn register during a data transfer.

Access (read/write) the IICn register only during the wait period. Accessing this register in communication states other than the wait period is prohibited. However, for the master device, the IICn register can be written once only after the transmission trigger bit (IICCn.STTn bit) has been set to 1.

A wait state is released by writing the IICn register during the wait period, and data transfer is started (n = 0, 1).

Reset sets this register to 00H.

After reset: 00H R/W Address: IIC0 FFFFFD80H, IIC1 FFFFFD90H

	7	6	5	4	3	2	1	0
IICn								

(n = 0, 1)

(9) Slave address registers 0, 1 (SVA0, SVA1)

The SVAn register holds the I²C bus's slave address.

This register can be read or written in 8-bit units, but bit 0 should be fixed to 0. However, rewriting this register is prohibited when the IICSn.STDn bit = 1 (start condition detection).

Reset sets this register to 00H.

After reset: 00H R/W Address: SVA0 FFFFFD83H, SVA1 FFFFFD93H

	7	6	5	4	3	2	1	0
SVAn								0

(n = 0, 1)

15.5 I²C Bus Mode Functions

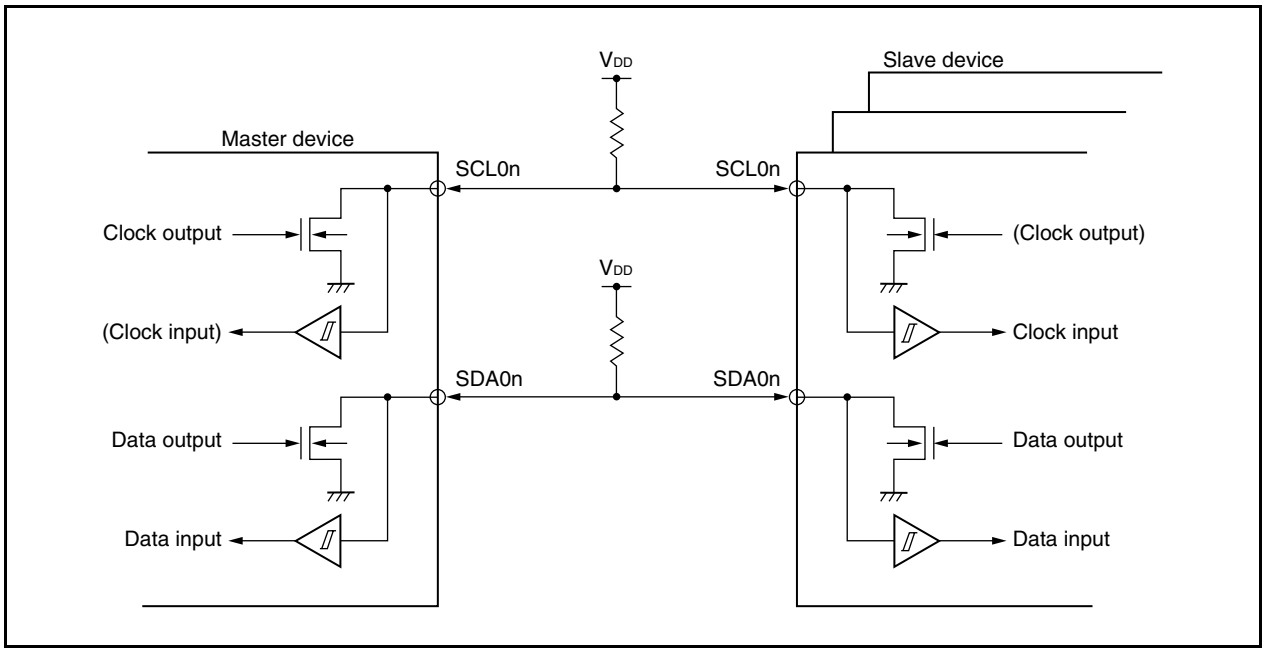
15.5.1 Pin configuration

The serial clock pin (SCL0n) and serial data bus pin (SDA0n) are configured as follows (n = 0, 1).

SCL0n ..... This pin is used for serial clock input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.
SDA0n ..... This pin is used for serial data input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Since outputs from the serial clock line and the serial data bus line are N-ch open-drain outputs, an external pull-up resistor is required.

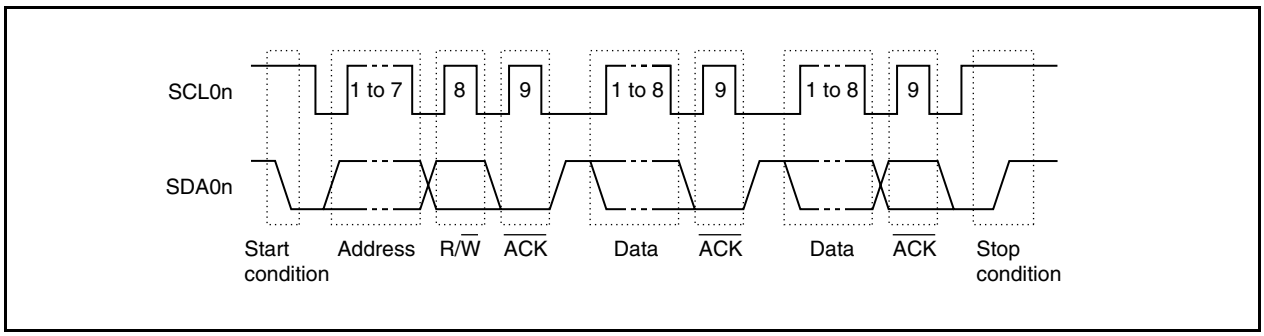
Figure 15-4. Pin Configuration Diagram



15.6 I²C Bus Definitions and Control Methods

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus. The transfer timing for the "start condition", "address", "transfer direction specification", "data", and "stop condition" generated on the I²C bus's serial data bus is shown below.

Figure 15-5 I²C Bus Serial Data Transfer Timing



The master device generates the start condition, slave address, and stop condition.

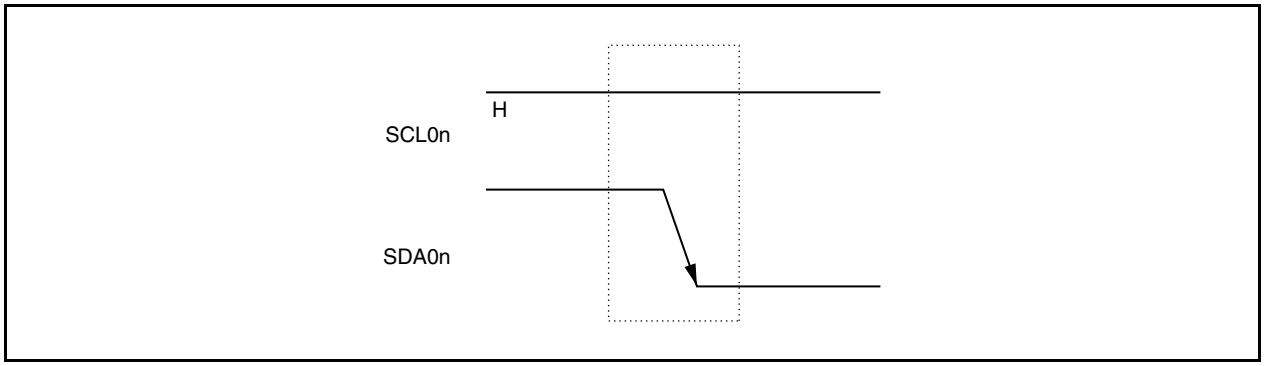
$\overline{ACK}$ can be generated by either the master or slave device (normally, it is generated by the device that receives 8-bit data).

The serial clock (SCL0n) is continuously output by the master device. However, in the slave device, the SCL0n pin's low-level period can be extended and a wait state can be inserted (n = 0, 1).

15.6.1 Start condition

A start condition is met when the SCL0n pin is high level and the SDA0n pin changes from high level to low level. The start condition for the SCL0n and SDA0n pins is generated that the master device to the slave device when starting a serial transfer. The slave device can defect the start condition (n = 0, 1).

Figure 15-6. Start Condition



A start condition is generated when the IICCn.STTn bit is set (1) after a stop condition has been detected (IICSn.SPDn bit = 1). When a start condition is detected, the IICSn.STDn bit is set (1) (n = 0, 1).

Caution When the IICCn.IICEn bit of the V850ES/SG1 is set to 1 while communications with other devices are in progress, the start condition may be detected depending on the status of the communication line. Be sure to set the IICCn.IICEn bit to 1 when the SCL0n and SDA0n lines are high level.

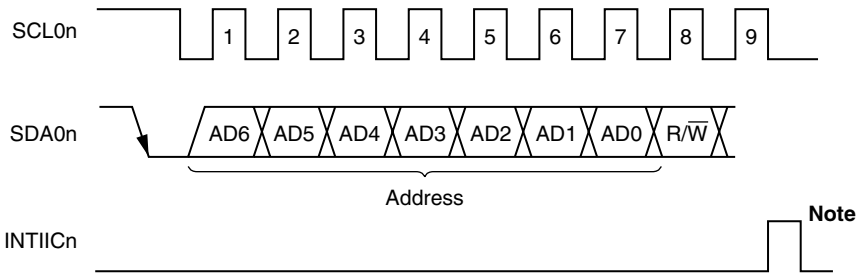
15.6.2 Addresses

The 7 bits of data that follow the start condition are defined as an address.

An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via the bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave devices include hardware that detects the start condition and checks whether or not the 7-bit address data matches the data values stored in the SVAn register. If the address data matches the values of the SVAn register, the slave device is selected and communicates with the master device until the master device generates a start condition or stop condition (n = 0, 1).

Figure 15-7. Address



Note The interrupt request signal (INTIICn) is generated if a local address or extension code is received during slave device operation.

Remark n = 0, 1

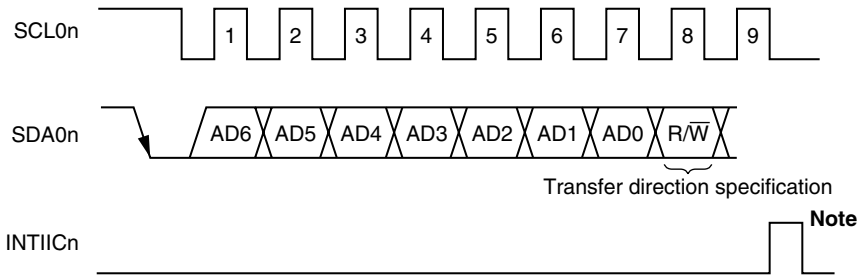
The slave address and the eighth bit, which specifies the transfer direction as described in **15.6.3 Transfer direction specification** below, are written together to IIC shift register n (IICn) and then output. Received addresses are written to the IICn register (n = 0, 1).

The slave address is assigned to the higher 7 bits of the IICn register.

15.6.3 Transfer direction specification

In addition to the 7-bit address data, the master device sends 1 bit that specifies the transfer direction. When this transfer direction specification bit has a value of 0, it indicates that the master device is transmitting data to a slave device. When the transfer direction specification bit has a value of 1, it indicates that the master device is receiving data from a slave device.

Figure 15-8. Transfer Direction Specification



Note The INTIICn signal is generated if a local address or extension code is received during slave device operation.

Remark n = 0, 1

15.6.4 $\overline{\text{ACK}}$

$\overline{\text{ACK}}$ is used to confirm the serial data status of the transmitting and receiving devices.

The receiving device returns $\overline{\text{ACK}}$ for every 8 bits of data it receives.

The transmitting device normally receives $\overline{\text{ACK}}$ after transmitting 8 bits of data. When $\overline{\text{ACK}}$ is returned from the receiving device, the reception is judged as normal and processing continues. The detection of $\overline{\text{ACK}}$ is confirmed with the IICSn.ACKDn bit.

When the master device is the receiving device, after receiving the final data, it does not return $\overline{\text{ACK}}$ and generates the stop condition. When the slave device is the receiving device and does not return $\overline{\text{ACK}}$, the master device generates either a stop condition or a restart condition, and then stops the current transmission. Failure to return $\overline{\text{ACK}}$ may be caused by the following factors.

(a) Reception was not performed normally.
(b) The final data was received.
(c) The receiving device (slave) does not exist for the specified address.

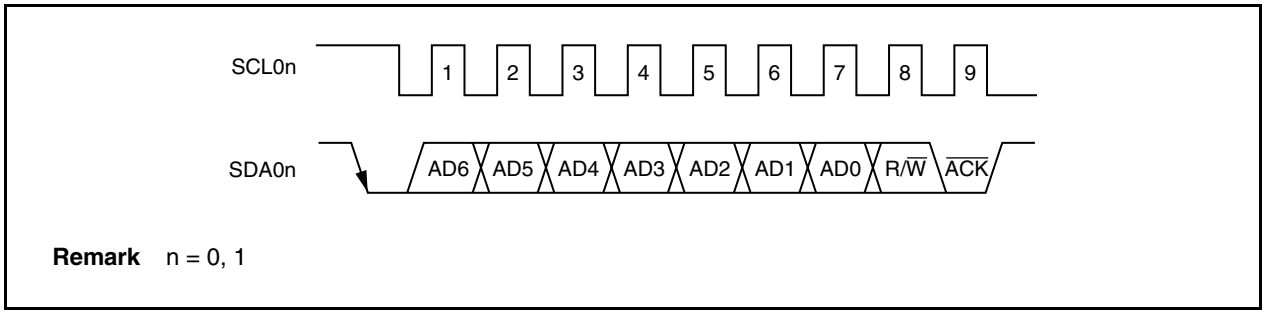
When the receiving device sets the SDA0n line to low level during the ninth clock, $\overline{\text{ACK}}$ is generated (normal reception).

When the IICCn.ACKEn bit is set to 1, automatic $\overline{\text{ACK}}$ generation is enabled. Transmission of the eighth bit following the 7 address data bits causes the IICSn.TRCn bit to be set. Normally, set the ACKEn bit to 1 for reception (TRCn bit = 0).

When the slave device is receiving (when TRCn bit = 0), if the slave device cannot receive data, clear the ACKEn bit to 0 to indicate to the master that no more data can be received.

Similarly, when the master device is receiving (when TRCn bit = 0) and the subsequent data is not needed, clear the ACKEn bit to 0 to prevent $\overline{\text{ACK}}$ from being generated. This notifies the slave device (transmitting device) of the end of the data transmission (transmission stopped).

Figure 15-9. $\overline{\text{ACK}}$



Remark n = 0, 1

When the local address is received, $\overline{\text{ACK}}$ is automatically generated regardless of the value of the ACKEn bit. No $\overline{\text{ACK}}$ is generated if the received address is not a local address (NACK).

When receiving the extension code, set the ACKEn bit to 1 in advance to generate $\overline{\text{ACK}}$.

The $\overline{\text{ACK}}$ generation method during data reception is based on the wait timing setting, as described by the following.

- When 8-clock wait is selected (IICCn.WTIMn bit = 0):
  $\overline{\text{ACK}}$ is generated at the falling edge of the SCL0n pin's eighth clock if the ACKEn bit is set to 1 before the wait state cancellation.
- When 9-clock wait is selected (IICCn.WTIMn bit = 1):
  $\overline{\text{ACK}}$ is generated if the ACKEn bit is set to 1 in advance.

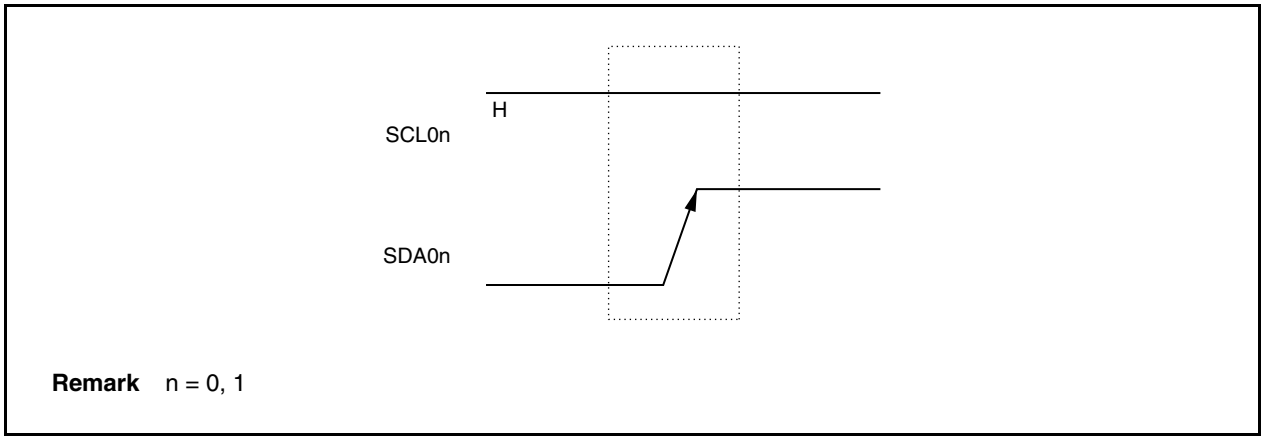
Remark n = 0, 1

15.6.5 Stop condition

When the SCL0n pin is high level, changing the SDA0n pin from low level to high level generates a stop condition (n = 0, 1).

A stop condition is generated when the master device outputs to the slave device when serial transfer has been completed. When used as the slave device, the start condition can be detected.

Figure 15-10. Stop Condition



Remark n = 0, 1

A stop condition is generated when the IICCn.SPTn bit is set to 1. When the stop condition is detected, the IICSn.SPDn bit is set to 1 and the interrupt request signal (INTIICn) is generated when the IICCn.SPIEn bit is set to 1 (n = 0, 1).

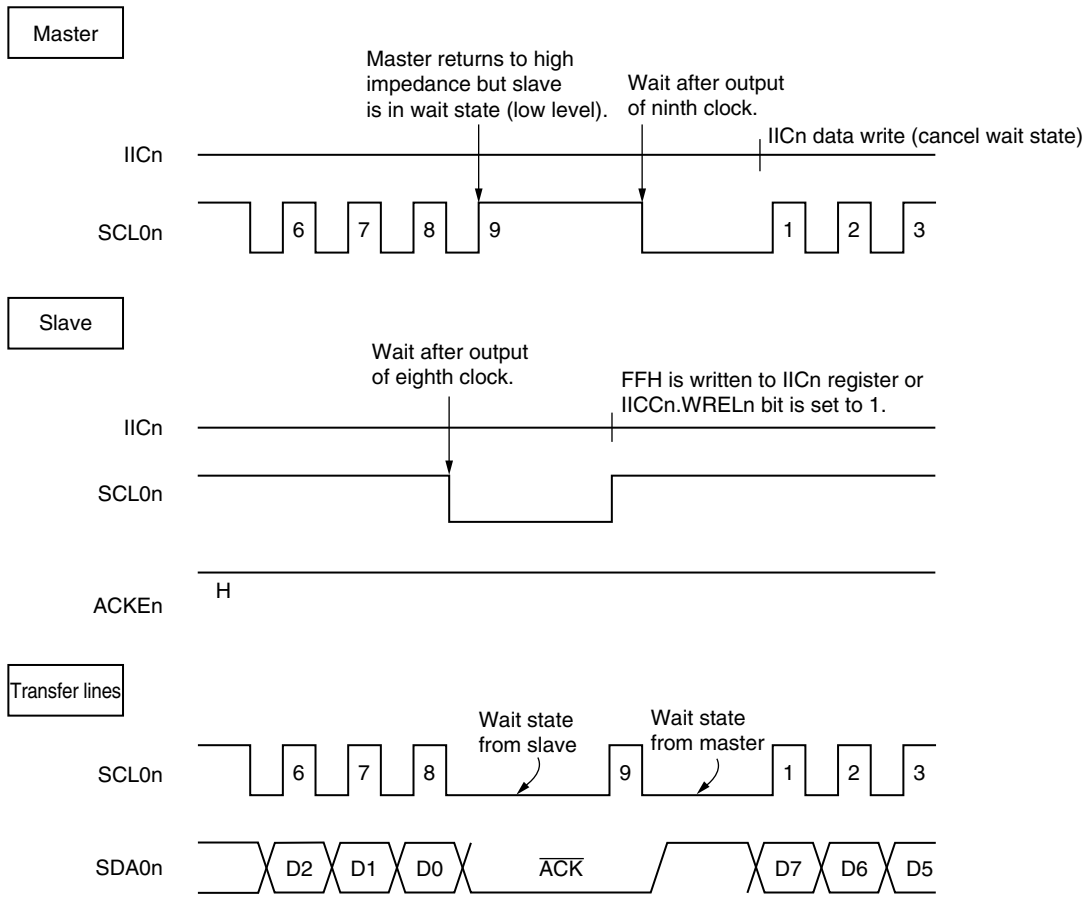
15.6.6 Wait state

A wait state is used to notify the communication partner that a device (master or slave) is preparing to transmit or receive data (i.e., is in a wait state).

Setting the SCL0n pin to low level notifies the communication partner of the wait state. When the wait state has been canceled for both the master and slave devices, the next data transfer can begin (n = 0, 1).

Figure 15-11. Wait State (1/2)

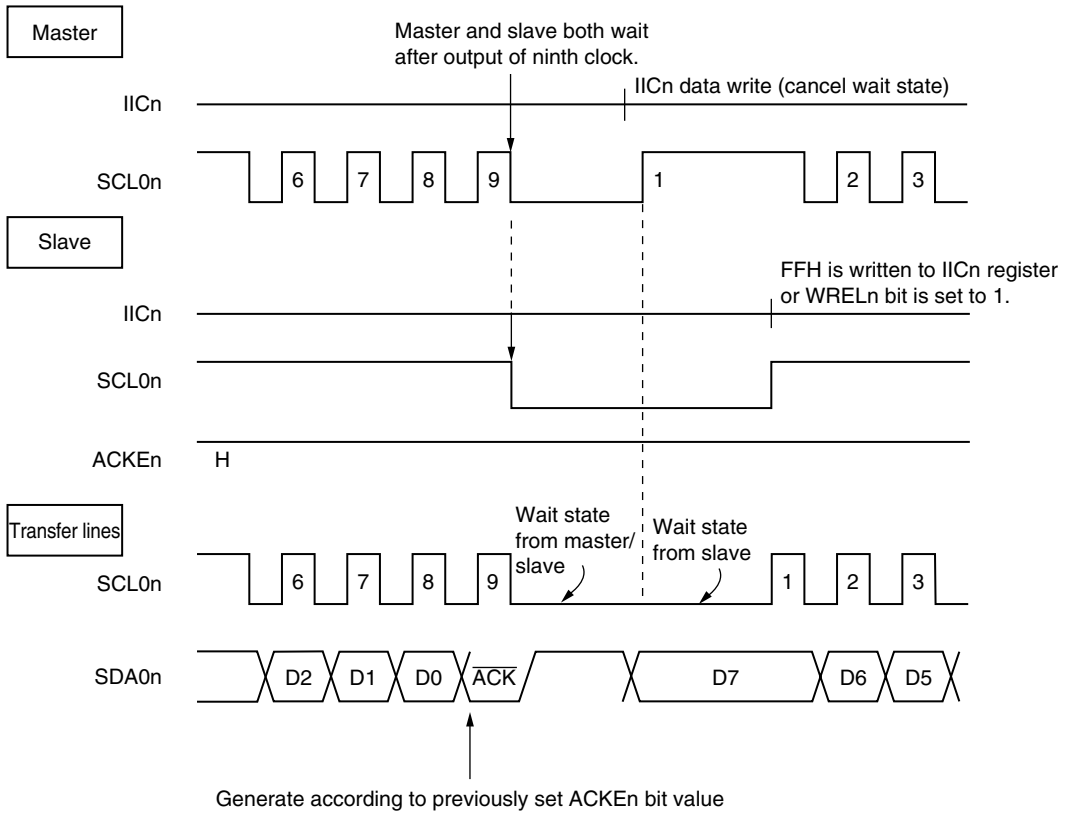
(a) When master device has a nine-clock wait and slave device has an eight-clock wait (master: transmission, slave: reception, and IICCn.ACKEn bit = 1)



Remark n = 0, 1

Figure 15-11. Wait State (2/2)

(b) When master and slave devices both have a nine-clock wait (master: transmission, slave: reception, and ACKEn bit = 1)



Remark n = 0, 1

<R> A wait state is automatically generated after generation of the start condition. A wait state is also automatically generated depending on the setting of the IICCn.WTIMn bit (n = 0, 1).

Normally, when the IICCn.WRELn bit is set to 1 or when FFH is written to the IICn register on the receiving side, the wait state is canceled and the transmitting side writes data to the IICn register to cancel the wait state.

The master device can also cancel the wait state via either of the following methods.

- By setting the IICCn.STTn bit to 1
- By setting the IICCn.SPTn bit to 1

15.6.7 Wait state cancellation method

In the case of I²C0n, wait state can be canceled normally in the following ways (n = 0, 1).

- By writing data to the IICn register
- By setting the IICCn.WRELn bit to 1 (wait state cancellation)
- <R> By setting the IICCn.STTn bit to 1 (start condition generation)[Note]
- <R> By setting the IICCn.SPTn bit to 1 (stop condition generation)[Note]

<R> **Note** Master only

If any of these wait state cancellation actions is performed, I²C0n will cancel wait state and restart communication.

When canceling wait state and sending data (including address), write data to the IICn register.

To receive data after canceling wait state, or to complete data transmission, set the WRELn bit to 1.

To generate a restart condition after canceling wait state, set the STTn bit to 1.

To generate a stop condition after canceling wait state, set the SPTn bit to 1.

Execute cancellation only once for each wait state.

For example, if data is written to the IICn register following wait state cancellation by setting the WRELn bit to 1, conflict between the SDAn line change timing and IICn register write timing may result in the data output to the SDAn line may be incorrect.

Even in other operations, if communication is stopped halfway, clearing the IICCn.IICEn bit to 0 will stop communication, enabling wait state to be cancelled.

If the I²C bus dead-locks due to noise, etc., setting the IICCn.LRELn bit to 1 causes the communication operation to be exited, enabling wait state to be cancelled.

15.7 I²C Interrupt Request Signals (INTIICn)

The following shows the value of the IICSn register at the INTIICn interrupt request signal generation timing and at the INTIICn signal timing.

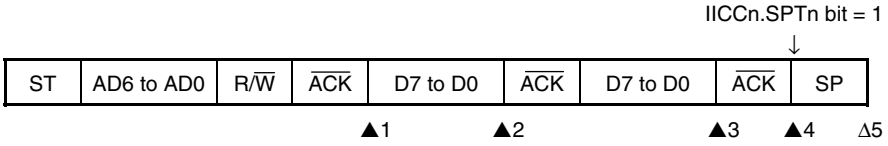
Remarks 1. ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

2. n = 0, 1

15.7.1 Master device operation

(1) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

<1> When IICCn.WTIMn bit = 0



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B

<R> ▲3: IICSn register = 1000X000B (WTIMn bit = 1[Note])

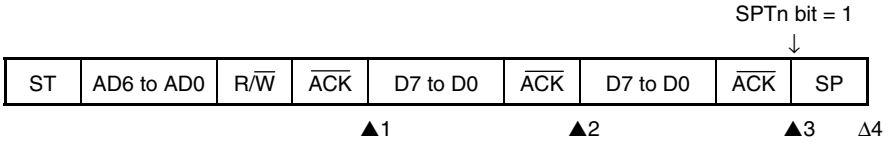
▲4: IICSn register = 1000XX00B

Δ 5: IICSn register = 00000001B

<R> **Note** Set the WTIMn bit (1) and change the timing of generating the interrupt request signal (INTIICn) to generate the stop condition.

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0, 1

<2> When WTIMn bit = 1



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X100B

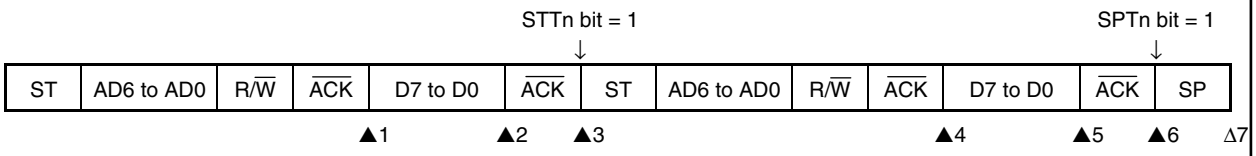
▲3: IICSn register = 1000XX00B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0, 1

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

<1> When WTIMn bit = 0



▲1: IICSn register = 1000X110B

<R> ▲2: IICSn register = 1000X000B (WTIMn bit = 1[Note 1])

<R> ▲3: IICSn register = 1000XX00B (WTIMn bit = 0[Note 2])

<R> ▲4: IICSn register = 1000X110B

<R> ▲5: IICSn register = 1000X000B (WTIMn bit = 1[Note 3])

▲6: IICSn register = 1000XX00B

Δ 7: IICSn register = 00000001B

<R> **Notes 1.** Set the WTIMn bit (1) and change the timing of generating the interrupt request signal (INTIICn) to generate the start condition.

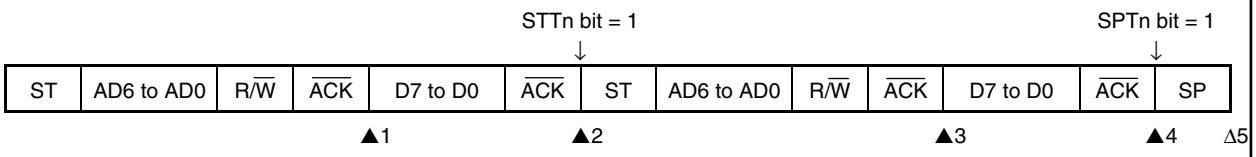
<R> **2.** Clear the WTIMn bit (0) to restore the original setting.

<R> **3.** Set the WTIMn bit (1) and change the timing of generating the interrupt request signal (INTIICn) to generate the stop condition.

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000XX00B

▲3: IICSn register = 1000X110B

▲4: IICSn register = 1000XX00B

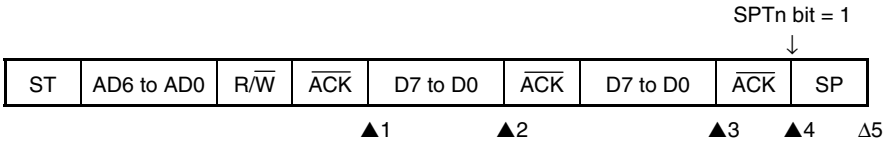
Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0, 1

(3) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

<1> When WTIMn bit = 0



▲1: IICSn register = 1010X110B

▲2: IICSn register = 1010X000B

<R> ▲3: IICSn register = 1010X000B (WTIMn bit = 1[Note])

▲4: IICSn register = 1010XX00B

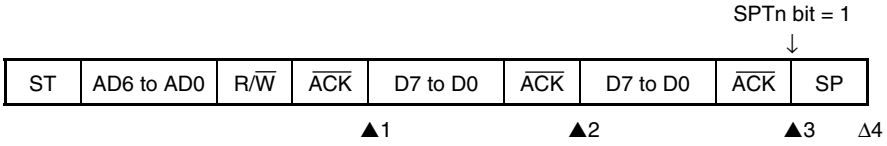
Δ 5: IICSn register = 00000001B

<R> **Note** Set the WTIMn bit (1) and change the timing of generating the interrupt request signal (INTIICn) to generate the stop condition.

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1



▲1: IICSn register = 1010X110B

▲2: IICSn register = 1010X100B

▲3: IICSn register = 1010XX00B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0, 1

15.7.2 Slave device operation (when receiving slave address (address match))

(1) Start ~ Address ~ Data ~ Data ~ Stop

<1> When IICCn.WTIMn bit = 0

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2		▲3	Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X000B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when IICCn.SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2		▲3	Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X100B

▲3: IICSn register = 0001XX00B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, address match)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2					▲3	▲4	Δ5

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001X000B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1 (after restart, address match)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1			▲2				▲3	▲4	Δ5

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001XX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, address mismatch (extension code reception))

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2				▲3		▲4		Δ5

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0010X010B

▲4: IICSn register = 0010X000B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1 (after restart, address mismatch (extension code reception))

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2				▲3	▲4		▲5	Δ6

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

▲3: IICSn register = 0010X010B

▲4: IICSn register = 0010X110B

▲5: IICSn register = 0010XX00B

Δ 6: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0, 1

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, address mismatch (= not extension code))

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2					▲3		Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001X000B

<R> ▲3: IICSn register = 00000110B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1 (after restart, address mismatch (= not extension code))

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1			▲2				▲3		Δ4

▲1: IICSn register = 0001X110B

▲2: IICSn register = 0001XX00B

<R> ▲3: IICSn register = 00000110B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

15.7.3 Slave device operation (when receiving extension code)

<R> The device always participates in communication when it receives the extension code.

(1) Start ~ Code ~ Data ~ Data ~ Stop

<1> When IICCn.WTIMn bit = 0

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X000B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when IICCn.SPIEn bit = 1
X: don't care
2. n = 0, 1

<2> When WTIMn bit = 1

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 ▲4 Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010X100B

▲4: IICSn register = 0010XX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0, 1

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, address match)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2					▲3	▲4	Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0001X110B

▲4: IICSn register = 0001X000B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1 (after restart, address match)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1	▲2		▲3				▲4	▲5	Δ6

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

▲4: IICSn register = 0001X110B

▲5: IICSn register = 0001XX00B

Δ 6: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, extension code reception)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2				▲3		▲4	Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X010B

▲4: IICSn register = 0010X000B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1 (after restart, extension code reception)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1	▲2		▲3			▲4	▲5		▲6 Δ7

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

▲4: IICSn register = 0010X010B

▲5: IICSn register = 0010X110B

▲6: IICSn register = 0010XX00B

Δ 7: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When WTIMn bit = 0 (after restart, address mismatch (= not extension code))

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X000B

<R> ▲3: IICSn register = 00000110B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1 (after restart, address mismatch (= not extension code))

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 ▲4 Δ5

▲1: IICSn register = 0010X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010XX00B

<R> ▲4: IICSn register = 00000110B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

15.7.4 Operation without communication

(1) Start ~ Code ~ Data ~ Data ~ Stop

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

Δ1

Δ 1: IICSn register = 00000001B

Remarks 1. Δ: Generated only when SPIEn bit = 1

2. n = 0, 1

15.7.5 Arbitration loss operation (operation as slave after arbitration loss)

<R> When the device is used as the master in a multi-master system, read the IICSn.MSTSn bit to check the arbitration result each time the INTIICn interrupt has been generated.

(1) When arbitration loss occurs during transmission of slave address data

<1> When IICCn.WTIMn bit = 0

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0101X110B

▲2: IICSn register = 0001X000B

▲3: IICSn register = 0001X000B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when IICCn.SPIEn bit = 1
X: don't care
2. n = 0, 1

<2> When WTIMn bit = 1

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 ▲3 Δ4

▲1: IICSn register = 0101X110B

▲2: IICSn register = 0001X100B

▲3: IICSn register = 0001XX00B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. n = 0, 1

(2) When arbitration loss occurs during transmission of extension code

<1> When WTIMn bit = 0

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1		▲2		▲3	Δ4

▲1: IICSn register = 0110X010B

▲2: IICSn register = 0010X000B

▲3: IICSn register = 0010X000B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1	▲2		▲3	▲4	Δ5

▲1: IICSn register = 0110X010B

▲2: IICSn register = 0010X110B

▲3: IICSn register = 0010X100B

▲4: IICSn register = 0010XX00B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

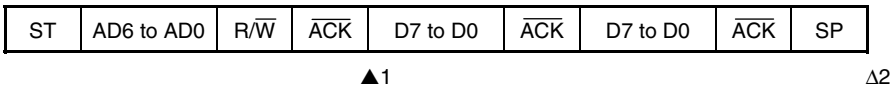
X: don't care

2. n = 0, 1

15.7.6 Operation when arbitration loss occurs (no communication after arbitration loss)

<R> When the device is used as the master in a multi-master system, read the IICSn.MSTSn bit to check the arbitration result each time the INTIICn interrupt has been generated.

(1) When arbitration loss occurs during transmission of slave address data



▲1 Δ2

▲1: IICSn register = 01000110B

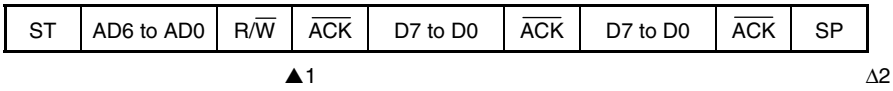
Δ 2: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when IICCn.SPIEn bit = 1

2. n = 0, 1

(2) When arbitration loss occurs during transmission of extension code



▲1 Δ2

▲1: IICSn register = 0110X010B

IICCn.LRELn bit is set to 1 by software

Δ 2: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

(3) When arbitration loss occurs during data transfer

<1> When IICCn.WTIMn bit = 0

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 Δ3

▲1: IICSn register = 10001110B

▲2: IICSn register = 01000000B

Δ 3: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

2. n = 0, 1

<2> When WTIMn bit = 1

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP

▲1 ▲2 Δ3

▲1: IICSn register = 10001110B

▲2: IICSn register = 01000100B

Δ 3: IICSn register = 00000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

2. n = 0, 1

(4) When arbitration loss occurs due to restart condition during data transfer

<1> Not extension code (Example: Address mismatch)

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to Dn	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1					▲2			Δ3

▲1: IICSn register = 1000X110B

▲2: IICSn register = 01000110B

Δ 3: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. Dn = D6 to D0
n = 0, 1

<2> Extension code

ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to Dn	ST	AD6 to AD0	R/$\overline{W}$	$\overline{ACK}$	D7 to D0	$\overline{ACK}$	SP
			▲1					▲2			Δ3

▲1: IICSn register = 1000X110B

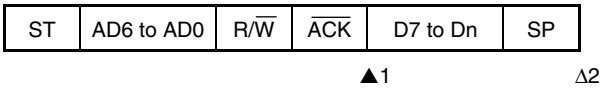
▲2: IICSn register = 0110X010B

IICCn.LRELn bit is set to 1 by software

Δ 3: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care
2. Dn = D6 to D0
n = 0, 1

(5) When arbitration loss occurs due to stop condition during data transfer



▲1 Δ2

▲1: IICSn register = 1000X110B

Δ 2: IICSn register = 01000001B

Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

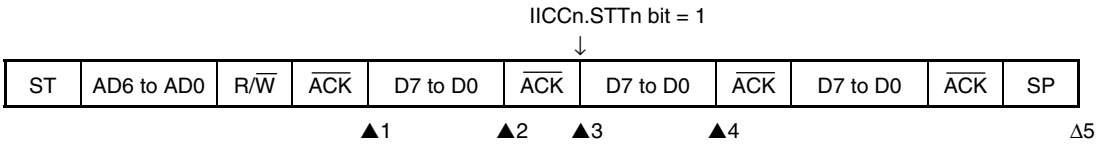
X: don't care

2. Dn = D6 to D0

n = 0, 1

(6) When arbitration loss occurs due to low level of SDA0n pin when attempting to generate a restart condition

<1> When WTIMn bit = 0



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B (WTIMn bit = 1)

<R> ▲3: IICSn register = 1000X100B (WTIMn bit = 0)

▲4: IICSn register = 01000000B

Δ 5: IICSn register = 00000001B

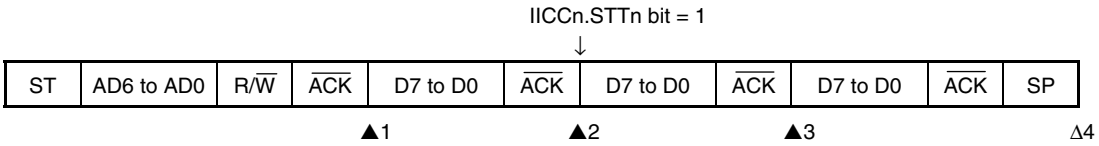
Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1



▲1: IICSn register = 1000X110B

<R> ▲2: IICSn register = 1000X100B

▲3: IICSn register = 01000100B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated

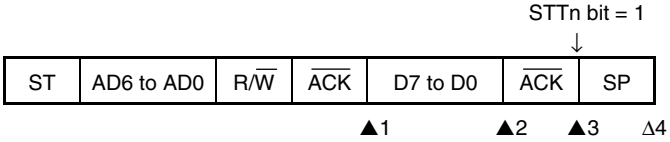
Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

(7) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

<1> When WTIMn bit = 0



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B (WTIMn bit = 1)

▲3: IICSn register = 1000XX00B

Δ 4: IICSn register = 01000001B

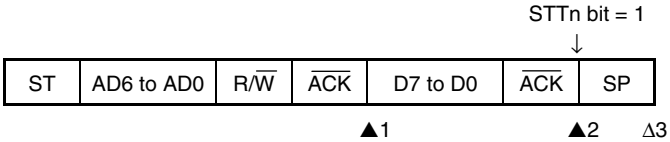
Remarks 1. ▲: Always generated

Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000XX00B

Δ 3: IICSn register = 01000001B

Remarks 1. ▲: Always generated

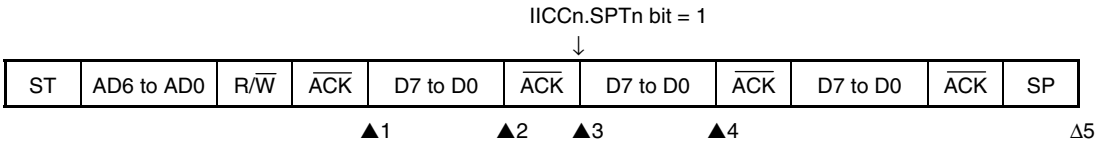
Δ: Generated only when SPIEn bit = 1

X: don't care

2. n = 0, 1

(8) When arbitration loss occurs due to low level of SDA0n pin when attempting to generate a stop condition

<1> When WTIMn bit = 0



▲1: IICSn register = 1000X110B

▲2: IICSn register = 1000X000B (WTIMn bit = 1)

<R> ▲3: IICSn register = 1000X100B (WTIMn bit = 0)

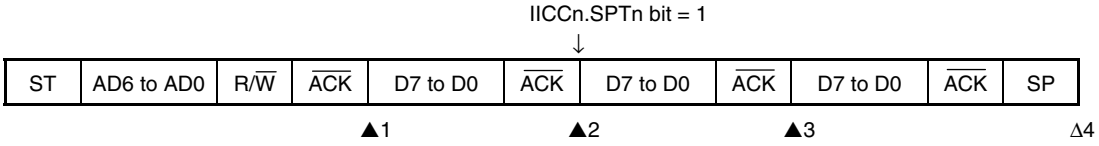
<R> ▲4: IICSn register = 01000100B

Δ 5: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0, 1

<2> When WTIMn bit = 1



▲1: IICSn register = 1000X110B

<R> ▲2: IICSn register = 1000X100B

<R> ▲3: IICSn register = 01000100B

Δ 4: IICSn register = 00000001B

Remarks 1. ▲: Always generated
Δ: Generated only when SPIEn bit = 1
X: don't care

2. n = 0, 1

15.8 Interrupt Request Signal (INTIICn) Generation Timing and Wait Control

The setting of the IICCn.WTIMn bit determines the timing by which the INTIICn register is generated and the corresponding wait control, as shown below (n = 0, 1).

Table 15-3. INTIICn Generation Timing and Wait Control

WTIMn Bit	During Slave Device Operation			During Master Device Operation		
	Address	Data Reception	Data Transmission	Address	Data Reception	Data Transmission
0	9 Notes 1, 2	8 Note 2	8 Note 2	9	8	8
1	9 Notes 1, 2	9 Note 2	9 Note 2	9	9	9

Notes 1. The slave device's INTIICn signal and wait period occur at the falling edge of the ninth clock only when there is a match with the address set to the SVAn register.
At this point, the $\overline{ACK}$ is generated regardless of the value set to the IICCn.ACKEn bit. For a slave device that has received an extension code, the INTIICn signal occurs at the falling edge of the eighth clock.
When the address does not match after restart, the INTIICn signal is generated at the falling edge of the ninth clock, but no wait occurs.

2. If the received address does not match the contents of the SVAn register and an extension code is not received, neither the INTIICn signal nor a wait occurs.

Remarks 1. The numbers in the table indicate the number of the serial clock's clock signals. Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

2. n = 0, 1

(1) During address transmission/reception

- Slave device operation: Interrupt and wait timing are determined regardless of the WTIMn bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIMn bit.

(2) During data reception

- Master/slave device operation: Interrupt and wait timing is determined according to the WTIMn bit.

(3) During data transmission

- Master/slave device operation: Interrupt and wait timing is determined according to the WTIMn bit.

(4) Wait cancellation method

The four wait cancellation methods are as follows.

- By writing data to the IICn register
- By setting the IICCn.WRELn bit to 1 (wait cancellation)
- By setting the IICCn.STTn bit to 1 (start condition generation)[Note]
- By setting the IICCn.SPTn bit to 1 (stop condition generation)[Note]

Note Master only

When an 8-clock wait has been selected (WTIMn bit = 0), whether or not the $\overline{ACK}$ has been generated must be determined prior to wait cancellation.

Remark n = 0, 1

(5) Stop condition detection

The INTIICn signal is generated when a stop condition is detected.

Remark n = 0, 1

15.9 Address Match Detection Method

In I²C bus mode, the master device can select a particular slave device by transmitting the corresponding slave address.

Address match detection is performed automatically by hardware. The INTIICn signal occurs when a local address has been set to the SVAn register and when the address set to the SVAn register matches the slave address sent by the master device, or when an extension code has been received (n = 0, 1).

15.10 Error Detection

In I²C bus mode, the status of the serial data bus pin (SDA0n) during data transmission is captured by the IICn register of the transmitting device, so the data of the IICn register prior to transmission can be compared with the transmitted IICn data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match (n = 0, 1).

15.11 Extension Code

(1) When the higher 4 bits of the receive address are either 0000 or 1111, the extension code flag (IICSn.EXCn bit) is set for extension code reception and an interrupt request signal (INTIICn) is issued at the falling edge of the eighth clock (n = 0, 1).
The local address stored in the SVAn register is not affected.

(2) If 11110xx0 is set to the SVAn register by a 10-bit address transfer and 11110xx0 is transferred from the master device, the results are as follows. Note that the INTIICn signal occurs at the falling edge of the eighth clock (n = 0, 1)

- Higher four bits of data match: EXCn bit = 1
- Seven bits of data match: IICSn.COIn bit = 1

(3) Since the processing after the interrupt request signal occurs differs according to the data that follows the extension code, such processing is performed by software.

<R> The device participates in communication when it receives the extension code while it is operating as a slave, even if the address does not match.
For example, when operation as a slave is not desired after the extension code is received, set the IICCn.LRELn bit to 1 and the CPU will enter the next communication wait state.

Table 15-4. Extension Code Bit Definitions

Slave Address	R/$\overline{W}$ Bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	X	CBUS address
0000 010	X	Address that is reserved for different bus format
1111 0xx	X	10-bit slave address specification

15.12 Arbitration

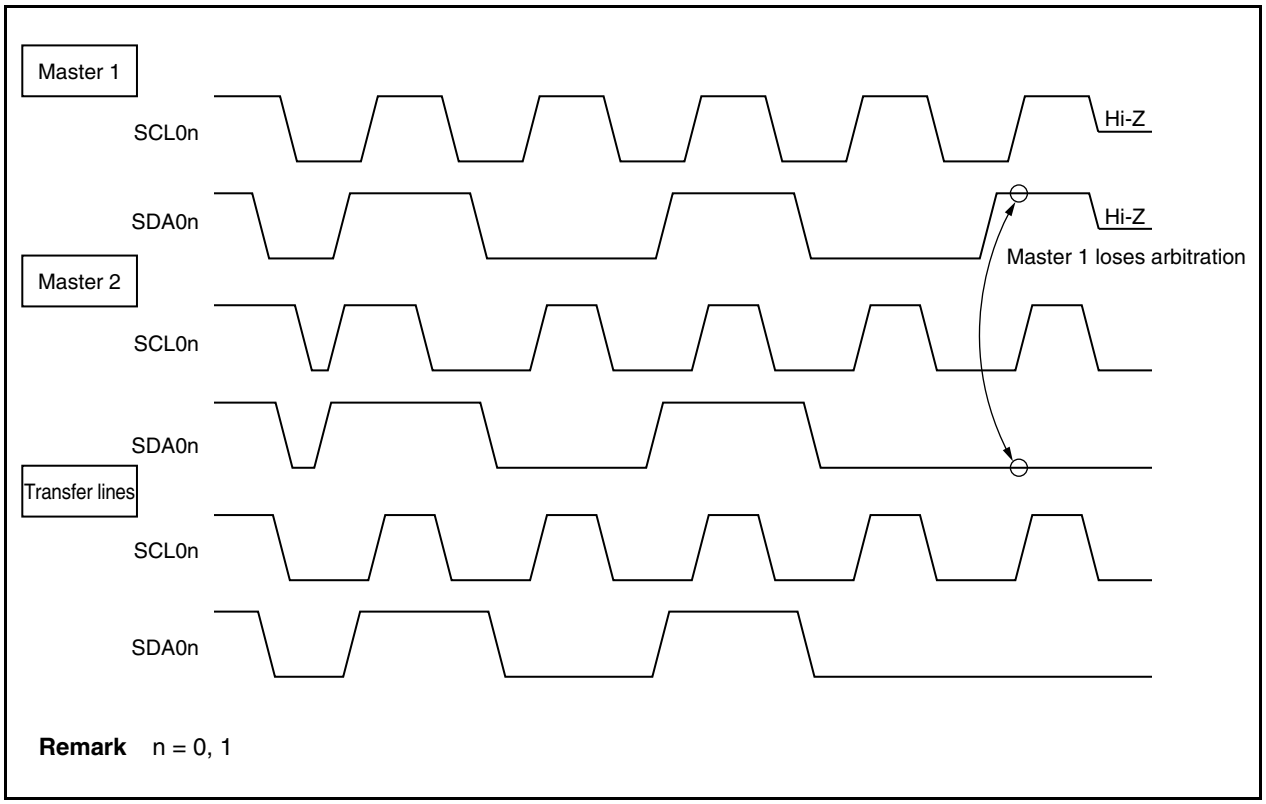
When several master devices simultaneously generate a start condition (when the IICCn.STTn bit is set to 1 before the IICSn.STDn bit is set to 1), communication between the master devices is performed while the number of clocks is adjusted until the data differs. This kind of operation is called arbitration (n = 0, 1).

When one of the master devices loses in arbitration, an arbitration loss flag (IICSn.ALDn bit) is set to 1 via the timing by which the arbitration loss occurred, and the SCL0n and SDA0n lines are both set to high impedance, which releases the bus (n = 0, 1).

Arbitration loss is detected based on the timing of the next interrupt request signal (INTIICn) (the eighth or ninth clock, when a stop condition is detected, etc.) and the setting of the ALDn bit to 1, which is made by software (n = 0, 1).

For details of interrupt request timing, see **15.7 I²C Interrupt Request Signals (INTIICn)**.

Figure 15-12. Arbitration Timing Example



Remark n = 0, 1

Table 15-5. Status During Arbitration and Interrupt Request Signal Generation Timing

Status During Arbitration	Interrupt Request Generation Timing
Transmitting address transmission	At falling edge of eighth or ninth clock following byte transfer[Note 1]
Read/write data after address transmission	
Transmitting extension code	
Read/write data after extension code transmission	
Transmitting data	
$\overline{\text{ACK}}$ transfer period after data reception	
When restart condition is detected during data transfer	
When stop condition is detected during data transfer	When stop condition is generated (when IICCn.SPIEn bit = 1)[Note 2]
When SDA0n pin is low level while attempting to generate restart condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When stop condition is detected while attempting to generate restart condition	When stop condition is generated (when IICCn.SPIEn bit = 1)[Note 2]
When DSA0n pin is low level while attempting to generate stop condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When SCL0n pin is low level while attempting to generate restart condition	

Notes 1. When the IICCn.WTIMn bit = 1, an INTIICn signal occurs at the falling edge of the ninth clock. When the WTIMn bit = 0 and the extension code's slave address is received, an INTIICn signal occurs at the falling edge of the eighth clock (n = 0, 1).

2. When there is a possibility that arbitration will occur, set the SPIEn bit to 1 for master device operation (n = 0, 1).

15.13 Wakeup Function

The I²C bus slave function is a function that generates an interrupt request signal (INTIICn) when a local address and extension code have been received.

This function makes processing more efficient by preventing unnecessary the INTIICn signal from occurring when addresses do not match.

When a start condition is detected, wakeup standby mode is set. This wakeup standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which has generated a start condition) to a slave device.

However, when a stop condition is detected, the IICCn.SPIEn bit is set regardless of the wakeup function, and this determines whether INTIICn signal is enabled or disabled (n = 0, 1).

15.14 Communication Reservation

15.14.1 When communication reservation function is enabled (IICFn.IICRSVn bit = 0)

To start master device communications when not currently using the bus, a communication reservation can be made to enable transmission of a start condition when the bus is released. There are two modes in which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ is not returned and the bus was released when the IICCn.LRELn bit was set to 1) (n = 0, 1).

If the IICCn.STTn bit is set to 1 while the bus is not used, a start condition is automatically generated and a wait status is set after the bus is released (after a stop condition is detected).

<R> The device automatically starts communication as the master when an address is written to the IICn register after the IICCn.SPIEn bit has been set (1) and release of the bus has been detected (i.e., the stop condition has been detected) by generation of an interrupt request (INTIICn). Data written to the IICn register is invalid before the stop condition is detected.

When STTn has been set to 1, the operation mode (as start condition or as communication reservation) is determined according to the bus status (n = 0, 1).

If the bus has been released ..... A start condition is generated
If the bus has not been released (standby mode) ..... Communication reservation

To detect which operation mode has been determined for the STTn bit, set the STTn bit to 1, wait for the wait period, then check the IICSn.MSTSn bit (n = 0, 1).

The wait periods, which should be set via software, are listed in Table 15-6. These wait periods can be set by the SMCn, CLn1, and CLn0 bits of the IICCLn register and the IICXn.CLXn bit (n = 0, 1).

Table 15-6. Wait Periods

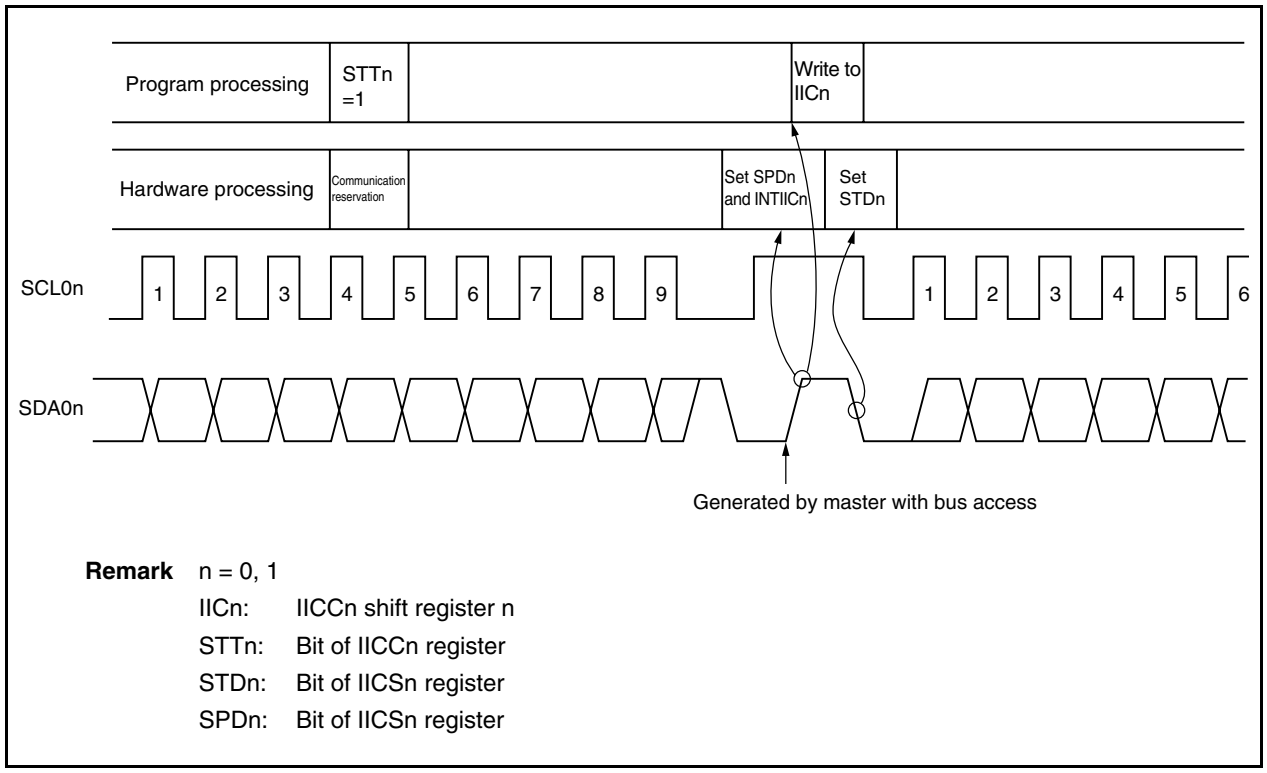
Clock Selection	CLXn	SMCn	CLn1	CLn0	Wait Period
fxx (when OCKSn = 18H set)	0	0	0	0	26 clocks
fxx/2 (when OCKSn = 10H set)	0	0	0	0	52 clocks
fxx/3 (when OCKSn = 11H set)	0	0	0	0	78 clocks
fxx/4 (when OCKSn = 12H set)	0	0	0	0	104 clocks
fxx/5 (when OCKSn = 13H set)	0	0	0	0	130 clocks
fxx (when OCKSn = 18H set)	0	0	0	1	47 clocks
fxx/2 (when OCKSn = 10H set)	0	0	0	1	94 clocks
fxx/3 (when OCKSn = 11H set)	0	0	0	1	141 clocks
fxx/4 (when OCKSn = 12H set)	0	0	0	1	188 clocks
fxx	0	0	1	0	47 clocks
fxx (when OCKSn = 18H set)	0	1	0	×	16 clocks
fxx/2 (when OCKSn = 10H set)	0	1	0	×	32 clocks
fxx/3 (when OCKSn = 11H set)	0	1	0	×	48 clocks
fxx/4 (when OCKSn = 12H set)	0	1	0	×	64 clocks
fxx	0	1	1	0	16 clocks
fxx (when OCKSn = 18H set)	1	1	0	×	10 clocks
fxx/2 (when OCKSn = 10H set)	1	1	0	×	20 clocks
fxx/3 (when OCKSn = 11H set)	1	1	0	×	30 clocks
fxx/4 (when OCKSn = 12H set)	1	1	0	×	40 clocks
fxx	1	1	1	0	10 clocks

Remarks 1. n = 0, 1

2. × = don't care

The communication reservation timing is shown below.

Figure 15-13. Communication Reservation Timing

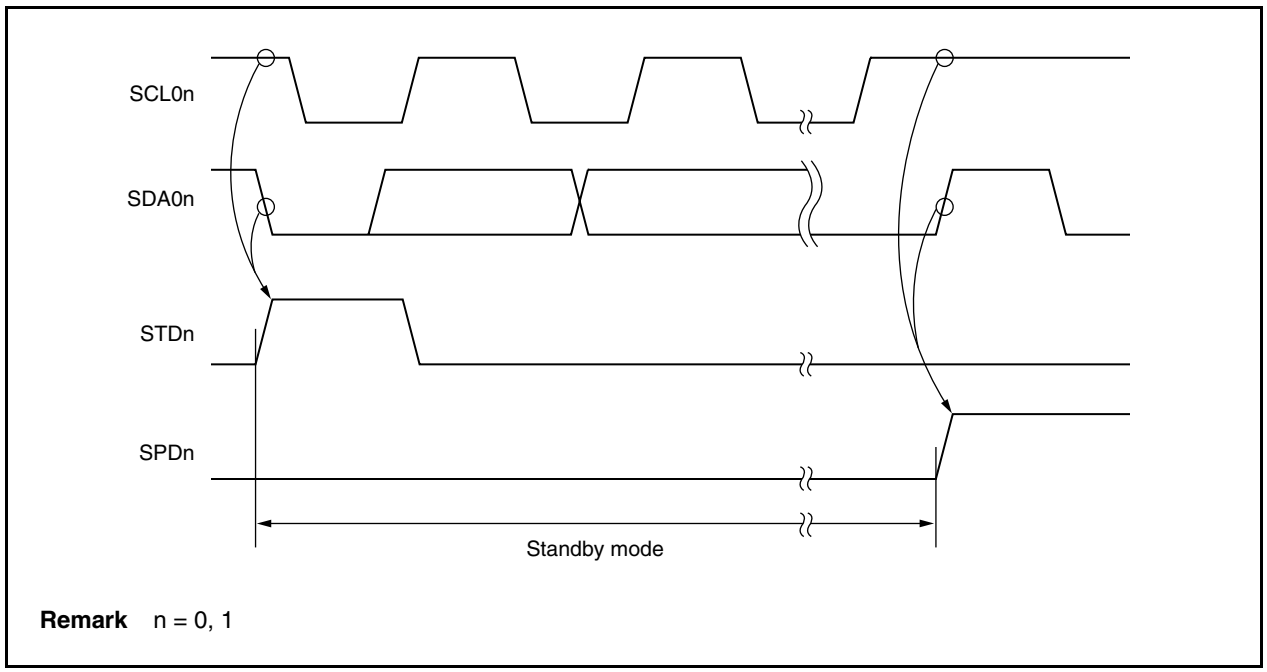


Remark n = 0, 1

IICn: IICCn shift register n
STTn: Bit of IICCn register
STDn: Bit of IICSn register
SPDn: Bit of IICSn register

Communication reservations are accepted via the following timing. After the IICSn.STDn bit is set to 1, a communication reservation can be made by setting the IICCn.STTn bit to 1 before a stop condition is detected (n = 0, 1).

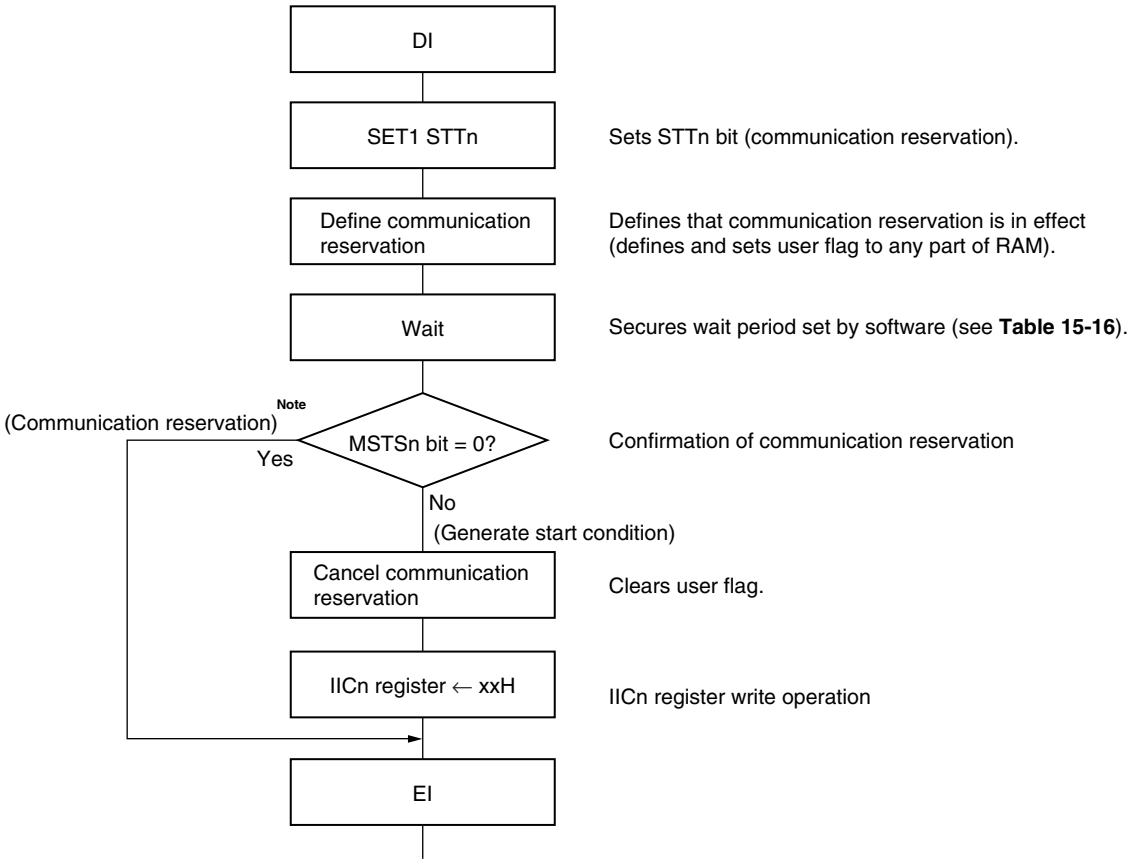
Figure 15-14. Timing for Accepting Communication Reservations



Remark n = 0, 1

The communication reservation flowchart is illustrated below.

Figure 15-15. Communication Reservation Flowchart



Note The communication reservation operation executes a write to the IICn register when a stop condition interrupt request occurs.

Remark n = 0, 1

15.14.2 When communication reservation function is disabled (IICFn.IICRSVn bit = 1)

When the IICCn.STTn bit is set when the bus is not used in a communication during bus communication, this request is rejected and a start condition is not generated. There are two modes in which the bus is not used

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ is not returned and the bus was released when the IICCn.LRELn bit was set to 1) (n = 0, 1).

To confirm whether the start condition was generated or request was rejected, check the IICFn.STCFn flag. The time shown in Table 15-7 is required until the STCFn flag is set after setting the STTn bit to 1. Therefore, secure the time by software.

<R>

Table 15-7. Wait Periods

Selection Clock	OCKSENn	OCKSTHn	OCKSn1	OCKSn0	CLn1	CLn0	Wait Period
fxx/2 (when setting OCKSn = 10H)	1	0	0	0	0	×	6 clocks
fxx/3 (when setting OCKSn = 11H)	1	0	0	1	0	×	9 clocks
fxx/4 (when setting OCKSn = 12H)	1	0	1	0	0	×	12 clocks
fxx/5 (when setting OCKSn = 13H)	1	0	1	1	0	×	15 clocks
fxx (when setting OCKSn = 18H)	1	1	0	0	0	×	3 clocks
fxx	0	0	0	0	1	0	3 clocks
fxx/2 (when setting OCKSn = 10H)	1	0	0	0	1	1	6 clocks
fxx/3 (when setting OCKSn = 11H)	1	0	0	1	1	1	9 clocks
fxx/4 (when setting OCKSn = 12H)	1	0	1	0	1	1	12 clocks
fxx/5 (when setting OCKSn = 13H)	1	0	1	1	1	1	15 clocks
fxx (when setting OCKSn = 18H)	1	1	0	0	1	1	3 clocks

Remarks 1. ×: don't care

2. n = 0, 1

3. Clock = fxx (main clock frequency)

15.15 Cautions

(1) When IICFn.STCENn bit = 0

Immediately after the I²C0n operation is enabled, the bus communication status (IICFn.IICBSYn bit = 1) is recognized regardless of the actual bus status. To execute master communication in the status where a stop condition has not been detected, generate a stop condition and then release the bus before starting the master communication.

Use the following sequence for generating a stop condition.

<1> Set the IICCLn register.
<2> Set the IICCn.IICEn bit.
<3> Set the IICCn.SPTn bit.

(2) When IICFn.STCENn bit = 1

Immediately after I²C0n operation is enabled, the bus released status (IICBSYn bit = 0) is recognized regardless of the actual bus status. To generate the first start condition (IICCn.STTn bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

(3) When the IICCn.IICEn bit of the V850ES/SG1 is set to 1 while communications with other devices are in progress, the start condition may be detected depending on the status of the communication line. Be sure to set the IICCn.IICEn bit to 1 when the SCL0n and SDA0n lines are high level.

(4) Determine the operation clock frequency by the IICCLn, IICXn, and OCKSn registers before enabling the operation (IICCn.IICEn bit = 1). To change the operation clock frequency, clear the IICCn.IICEn bit to 0 once.

(5) After the IICCn.STTn and IICCn.SPTn bits have been set to 1, they must not be re-set without being cleared to 0 first.

(6) If transmission has been reserved, set the IICCN.SPIEn bit to 1 so that an interrupt request is generated by the detection of a stop condition. After an interrupt request has been generated, the wait status will be released by writing communication data to I²Cn, then transferring will begin. If an interrupt is not generated by the detection of a stop condition, transmission will halt in the wait status because an interrupt request was not generated. However, it is not necessary to set the SPIEn bit to 1 for the software to detect the IICSn.MSTSn bit.

Remark n = 0, 1

<R>

15.16 Communication Operations

The following shows three operation procedures with the flowchart.

(1) Master operation in single master system

The flowchart when using the V850ES/SG1 as the master in a single master system is shown below.

This flowchart is broadly divided into the initial settings and communication processing. Execute the initial settings at startup. If communication with the slave is required, prepare the communication and then execute communication processing.

(2) Master operation in multimaster system

In the I²C0n bus multimaster system, whether the bus is released or used cannot be judged by the I²C bus specifications when the bus takes part in a communication. Here, when data and clock are at a high level for a certain period (1 frame), the V850ES/SG1 takes part in a communication with bus released state.

This flowchart is broadly divided into the initial settings, communication waiting, and communication processing. The processing when the V850ES/SG1 looses in arbitration and is specified as the slave is omitted here, and only the processing as the master is shown. Execute the initial settings at startup to take part in a communication. Then, wait for the communication request as the master or wait for the specification as the slave. The actual communication is performed in the communication processing, and it supports the transmission/reception with the slave and the arbitration with other masters.

(3) Slave operation

An example of when the V850ES/SG1 is used as the slave of the I²C0n bus is shown below.

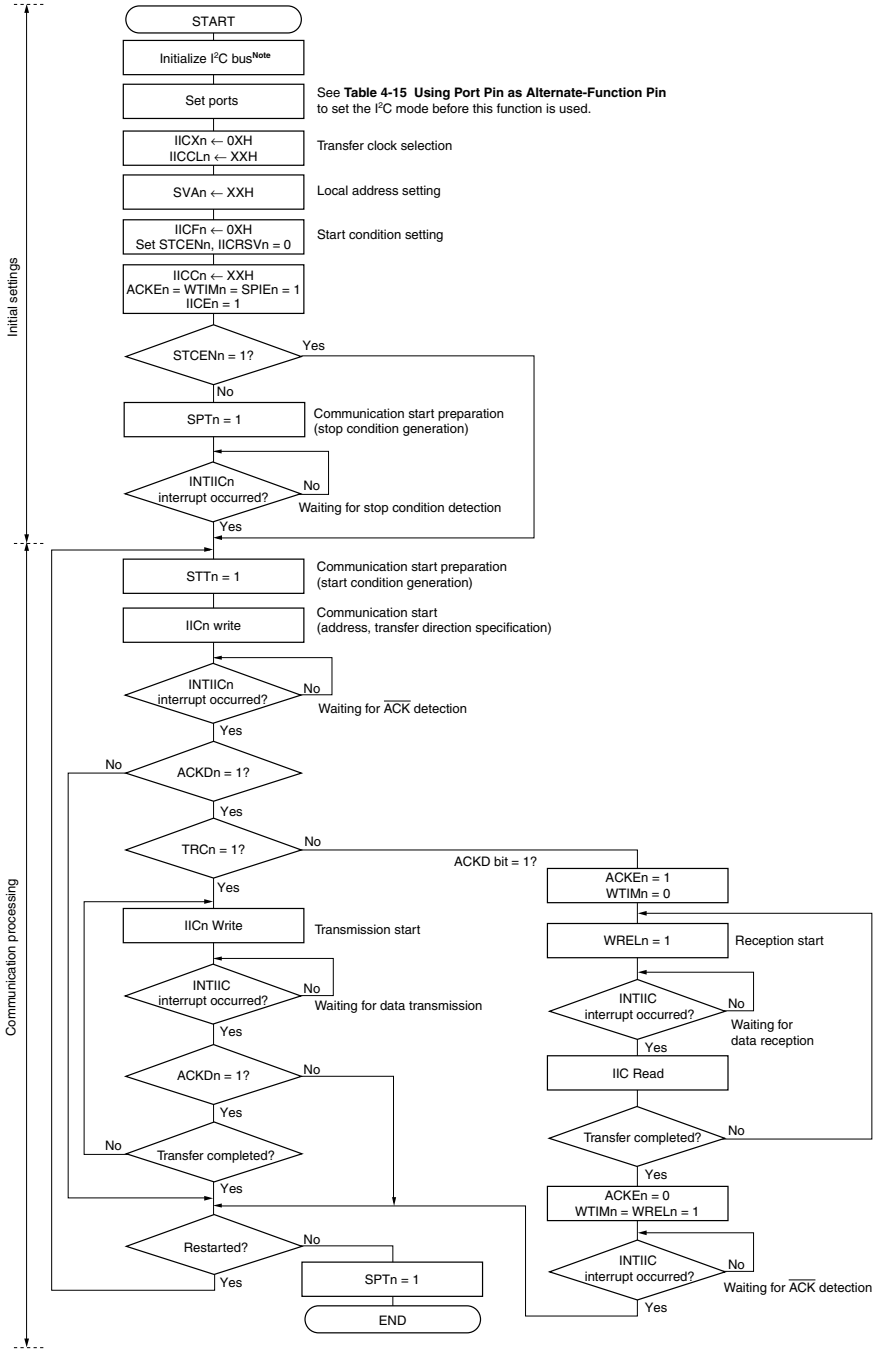
When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIICn interrupt occurrence (communication waiting). When the INTIICn interrupt occurs, the communication status is judged and its result is passed as a flag over to the main processing.

By checking the flags, necessary communication processing is performed.

Remark n = 0, 1

<R> 15.16.1 Master operation in single master system

Figure 15-16. Master Operation in Single Master System



Note Release the I²C0n bus (SCL0n, SDA0n pins = high level) in conformity with the specifications of the product in communication.

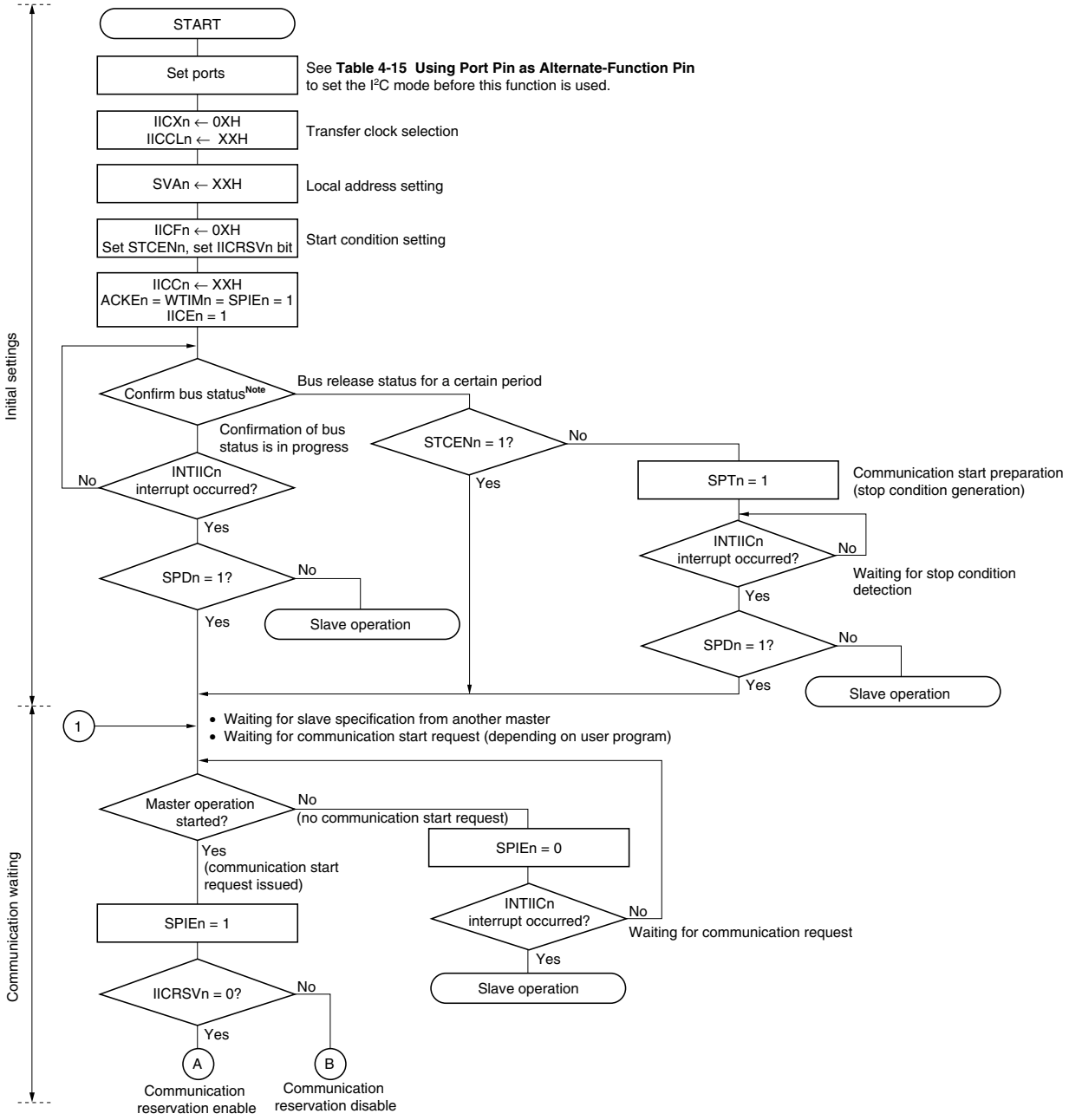
For example, when the EEPROM™ outputs a low level to the SDA0n pin, set the SCL0n pin to the output port and output clock pulses from that output port until when the SDA0n pin is constantly high level.

Remarks 1. For the transmission and reception formats, conform to the specifications of the product in communication.

2. n = 0, 1

<R> **15.16.2 Master operation in multimaster system**

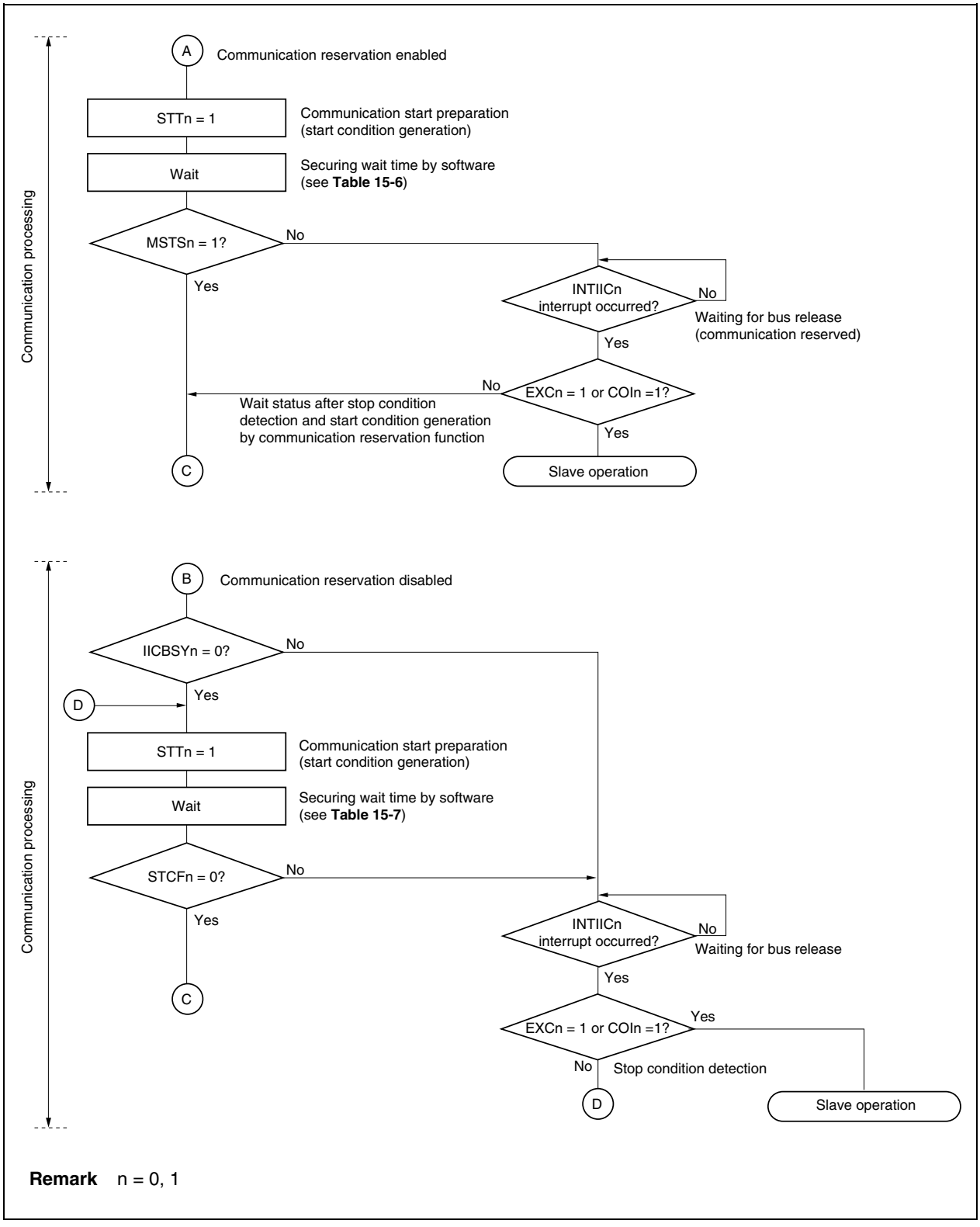
Figure 15-17. Master Operation in Multimaster System (1/3)



Note Confirm that the bus release status (IICCLn.CLDn bit = 1, IICCLn.DADn bit = 1) has been maintained for a certain period (1 frame, for example). When the SDA0n pin is constantly low level, determine whether to release the I²C0n bus (SCL0n, SDA0n pins = high level) by referring to the specifications of the product in communication.

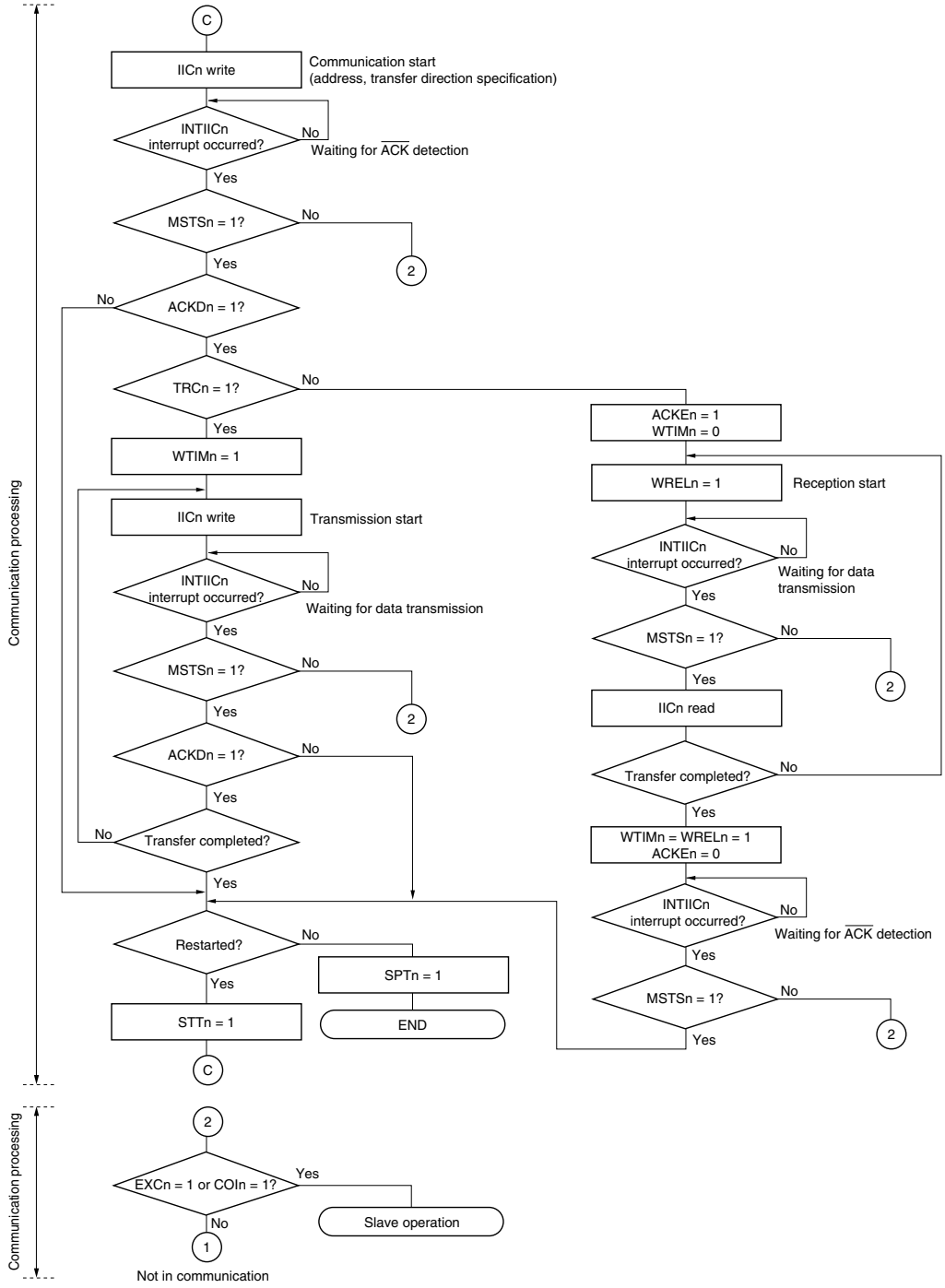
Remark n = 0, 1

Figure 15-17. Master Operation in Multimaster System (2/3)



Remark n = 0, 1

Figure 15-17. Master Operation in Multimaster System (3/3)



Remarks 1. Conform the transmission and reception formats to the specifications of the product in communication.

2. When using the V850ES/SG1 as the master in the multimaster system, read the IICSn.MSTSn bit for each INTIICn interrupt occurrence to confirm the arbitration result.

3. When using the V850ES/SG1 as the slave in the multimaster system, confirm the status using the IICSn and IICFn registers for each INTIICn interrupt occurrence to determine the next processing.

4. n = 0, 1

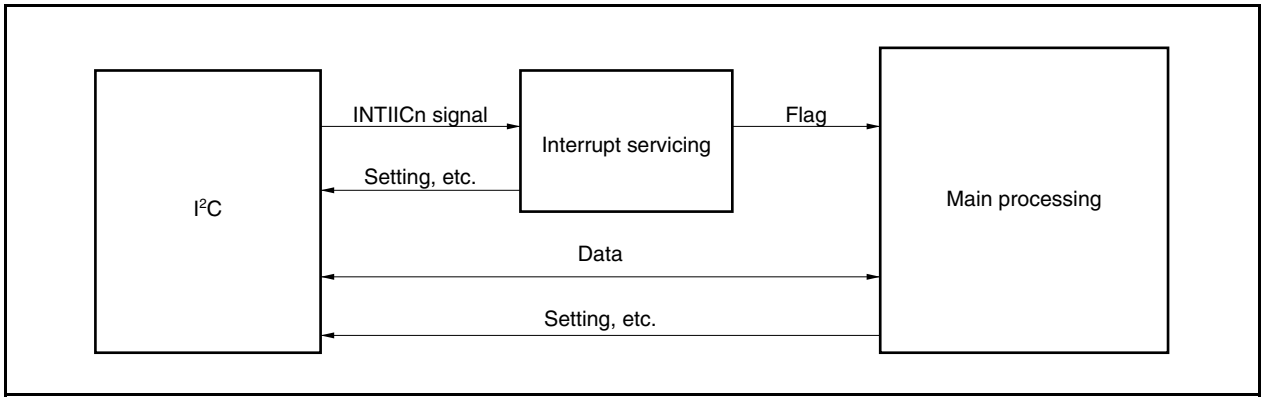
15.16.3 Slave operation

The following shows the processing procedure of the slave operation.

Basically, the operation of the slave device is event-driven. Therefore, processing by an INTIICn interrupt (processing requiring a significant change of the operation status, such as stop condition detection during communication) is necessary.

The following description assumes that data communication does not support extension codes. Also, it is assumed that the INTIICn interrupt servicing performs only status change processing and that the actual data communication is performed during the main processing.

Figure 15-18. Software Outline During Slave Operation



Therefore, the following three flags are prepared so that the data transfer processing can be performed by transmitting these flags to the main processing instead of INTIICn signal.

(1) Communication mode flag

This flag indicates the following communication statuses.

Clear mode: Data communication not in progress

Communication mode: Data communication in progress (valid address detection stop condition detection, $\overline{\text{ACK}}$ from master not detected, address mismatch)

(2) Ready flag

This flag indicates that data communication is enabled. This is the same status as an INTIICn interrupt during normal data transfer. This flag is set in the interrupt processing block and cleared in the main processing block. The ready flag for the first data for transmission is not set in the interrupt processing block, so the first data is transmitted without clear processing (the address match is regarded as a request for the next data).

(3) Communication direction flag

This flag indicates the direction of communication and is the same as the value of IICSn.TRCn bit.

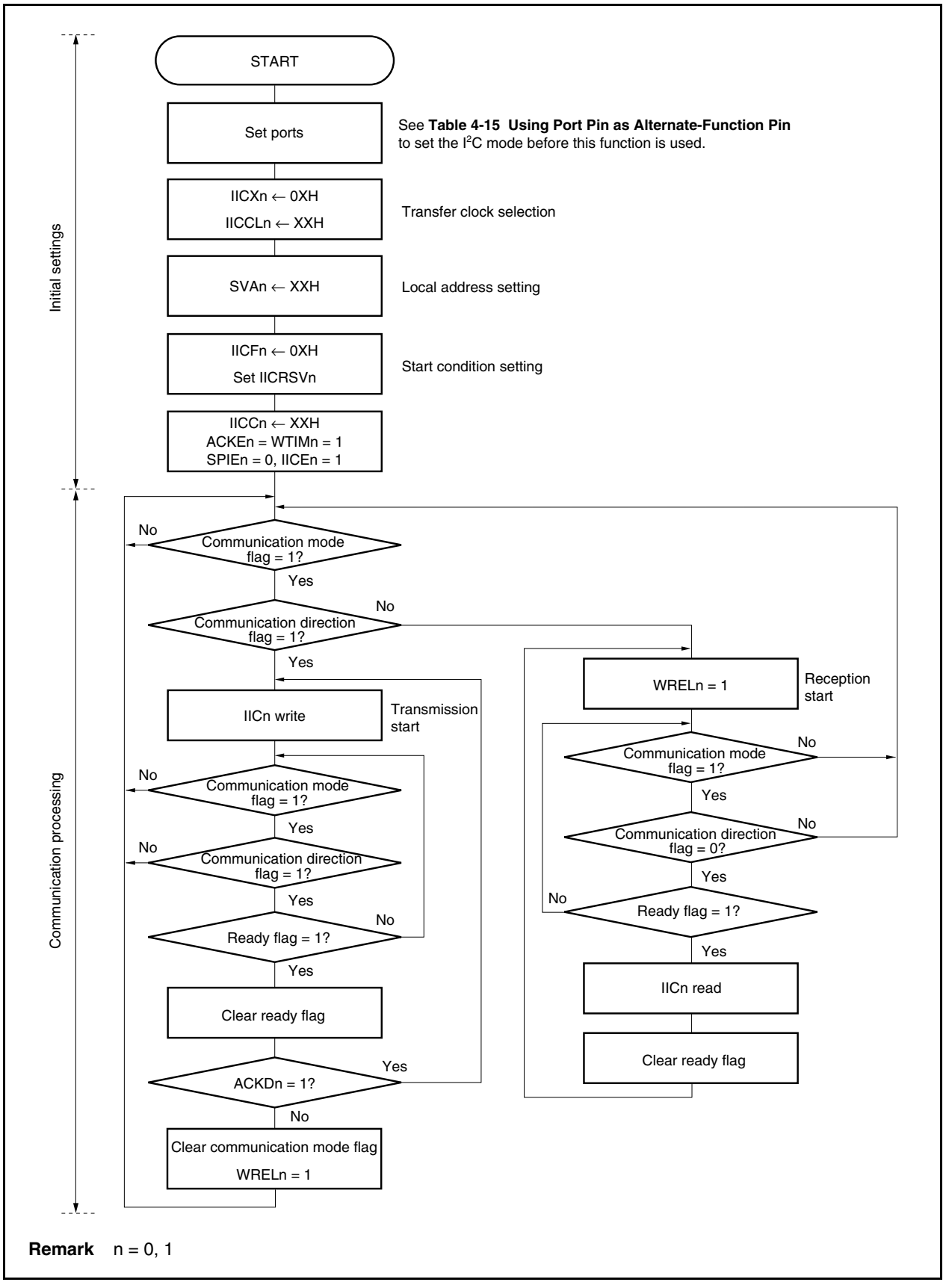
The following shows the operation of the main processing block during slave operation.

Start I²C0n and wait for the communication enabled status. When communication is enabled, perform transfer using the communication mode flag and ready flag (the processing of the stop condition and start condition is performed by interrupts, conditions are confirmed by flags).

For transmission, repeat the transmission operation until the master device stops returning $\overline{\text{ACK}}$. When the master device stops returning $\overline{\text{ACK}}$, transfer is complete.

For reception, receive the required number of data and do not return $\overline{\text{ACK}}$ for the next data immediately after transfer is complete. After that, the master device generates the stop condition or restart condition. This causes exit from communications.

<R> **Figure 15-19. Slave Operation Flowchart (1)**



Remark n = 0, 1

The following shows an example of the processing of the slave device by an INTIICn interrupt (it is assumed that no extension codes are used here). During an INTIICn interrupt, the status is confirmed and the following steps are executed.

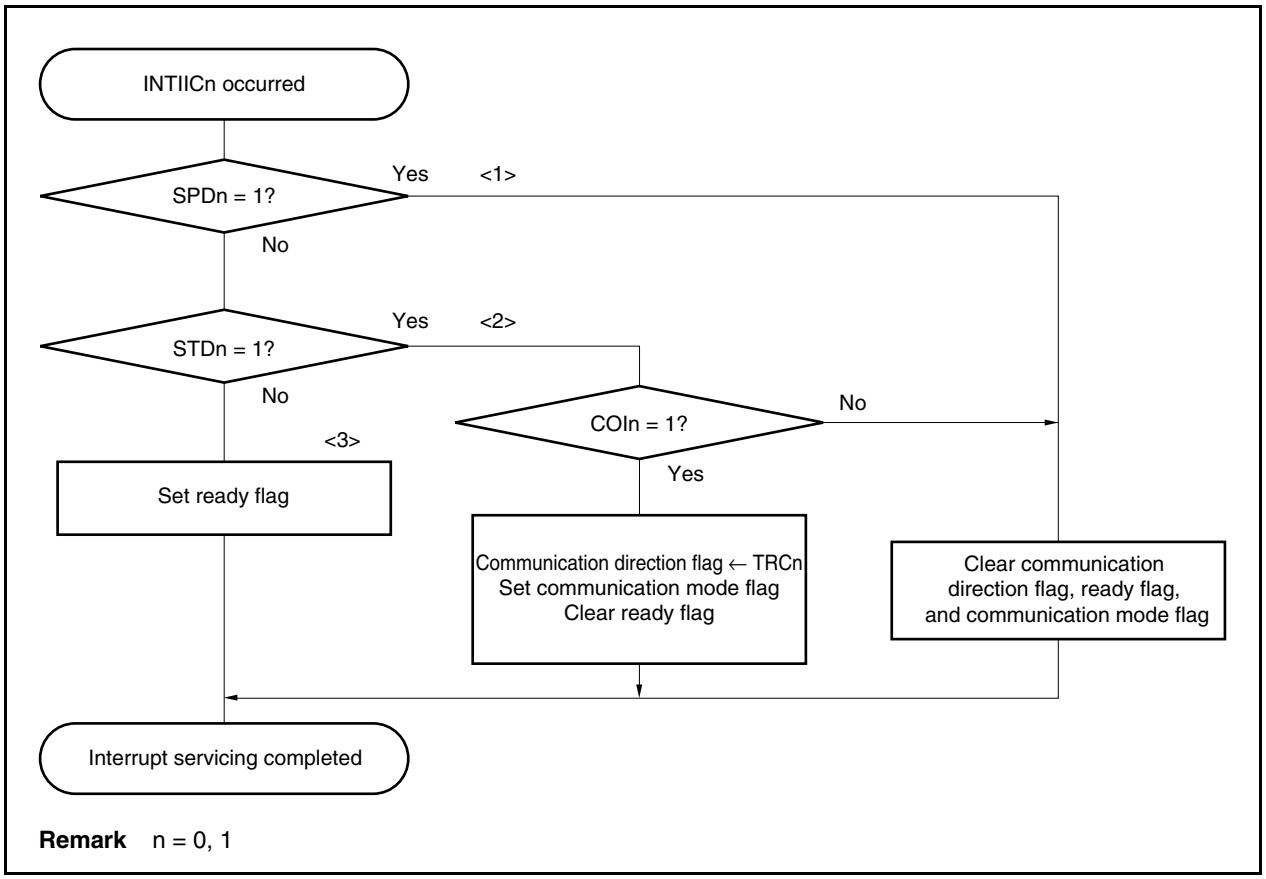
<1> When a stop condition is detected, communication is terminated.

<2> When a start condition is detected, the address is confirmed. If the address does not match, communication is terminated. If the address matches, the communication mode is set and wait is released, and operation returns from the interrupt (the ready flag is cleared).

<3> For data transmission/reception, when the ready flag is set, operation returns from the interrupt while the I²C0n bus remains in the wait status.

Remark <1> to <3> in the above correspond to <1> to <3> in **Figure 15-20 Slave Operation Flowchart (2)**.

<R> **Figure 15-20. Slave Operation Flowchart (2)**



Remark n = 0, 1

15.17 Timing of Data Communication

When using I²C bus mode, the master device generates an address via the serial bus to select one of several slave devices as its communication partner.

After outputting the slave address, the master device transmits the IICSn.TRCn bit, which specifies the data transfer direction, and then starts serial communication with the slave device.

The shift operation of the IICn register is synchronized with the falling edge of the serial clock pin (SCL0n). The transmit data is transferred to the SO latch and is output (MSB first) via the SDA0n pin.

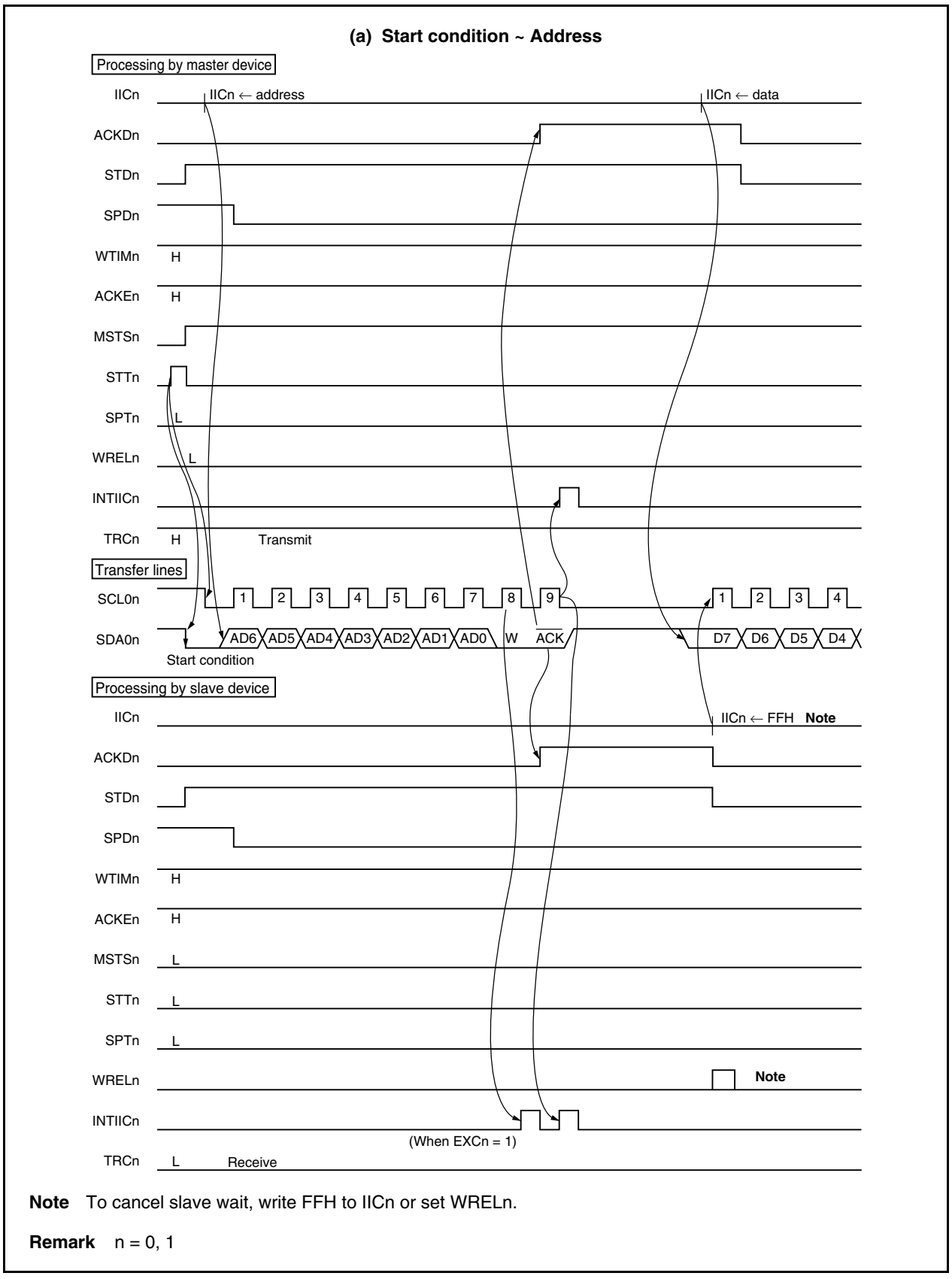
Data input via the SDA0n pin is captured by the IICn register at the rising edge of the SCL0n pin.

The data communication timing is shown below.

Remark n = 0, 1

<R>

Figure 15-21. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)



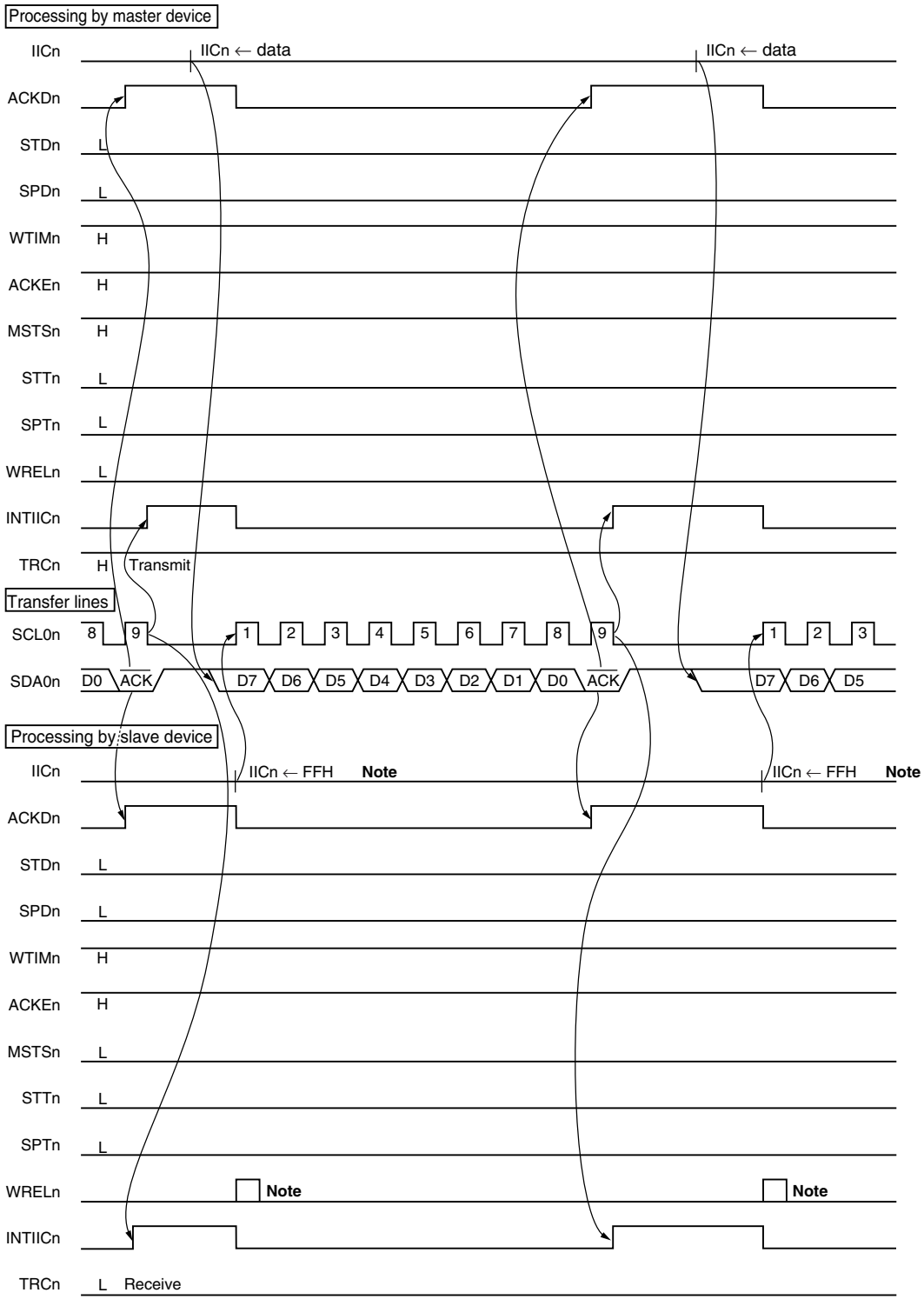
Note To cancel slave wait, write FFH to IICn or set WRELn.

Remark n = 0, 1

<R>

Figure 15-21. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)

(b) Data



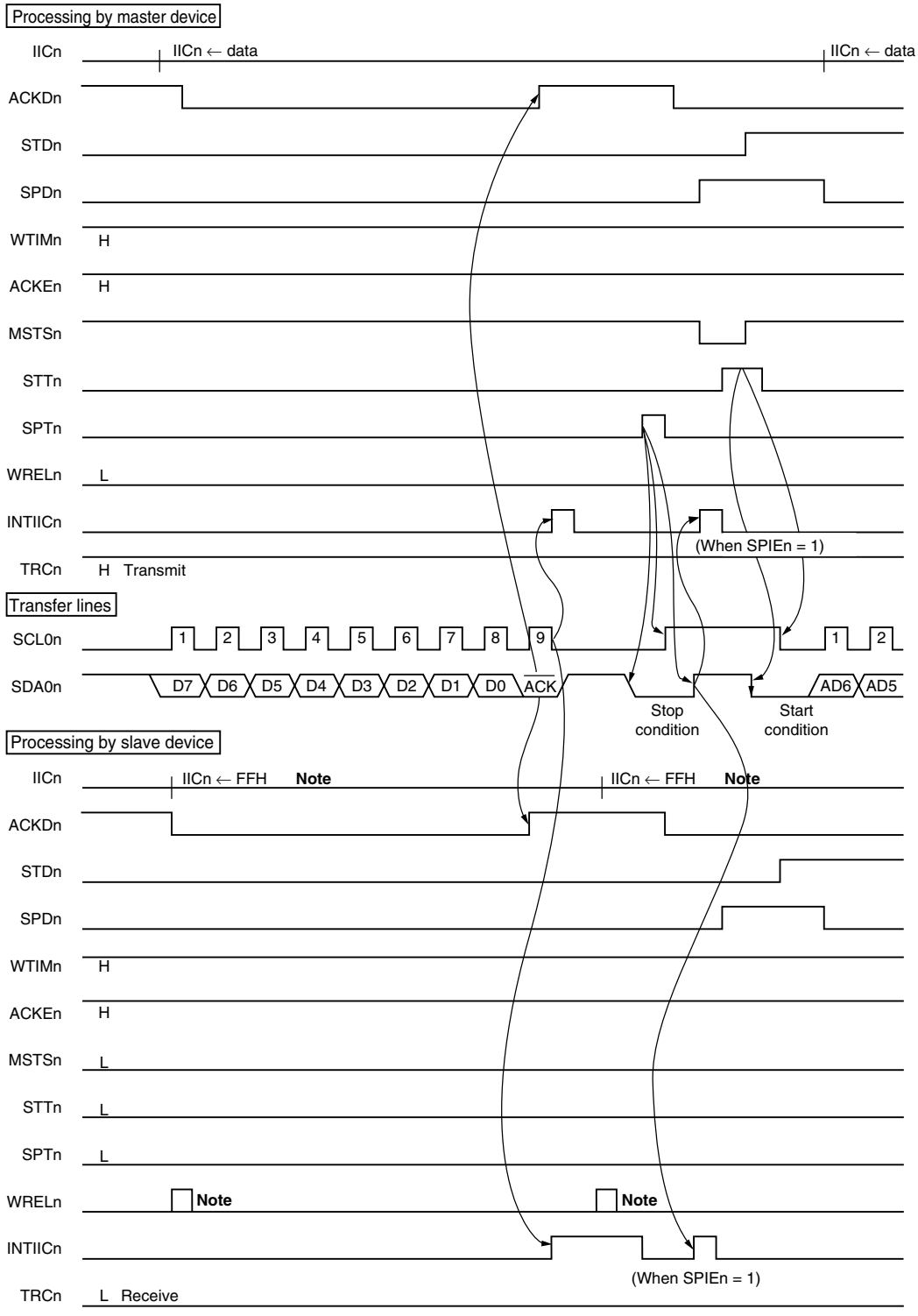
Note To cancel slave wait, write FFH to IICn or set WRELn.

Remark n = 0, 1

<R>

Figure 15-21. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)

(c) Stop condition



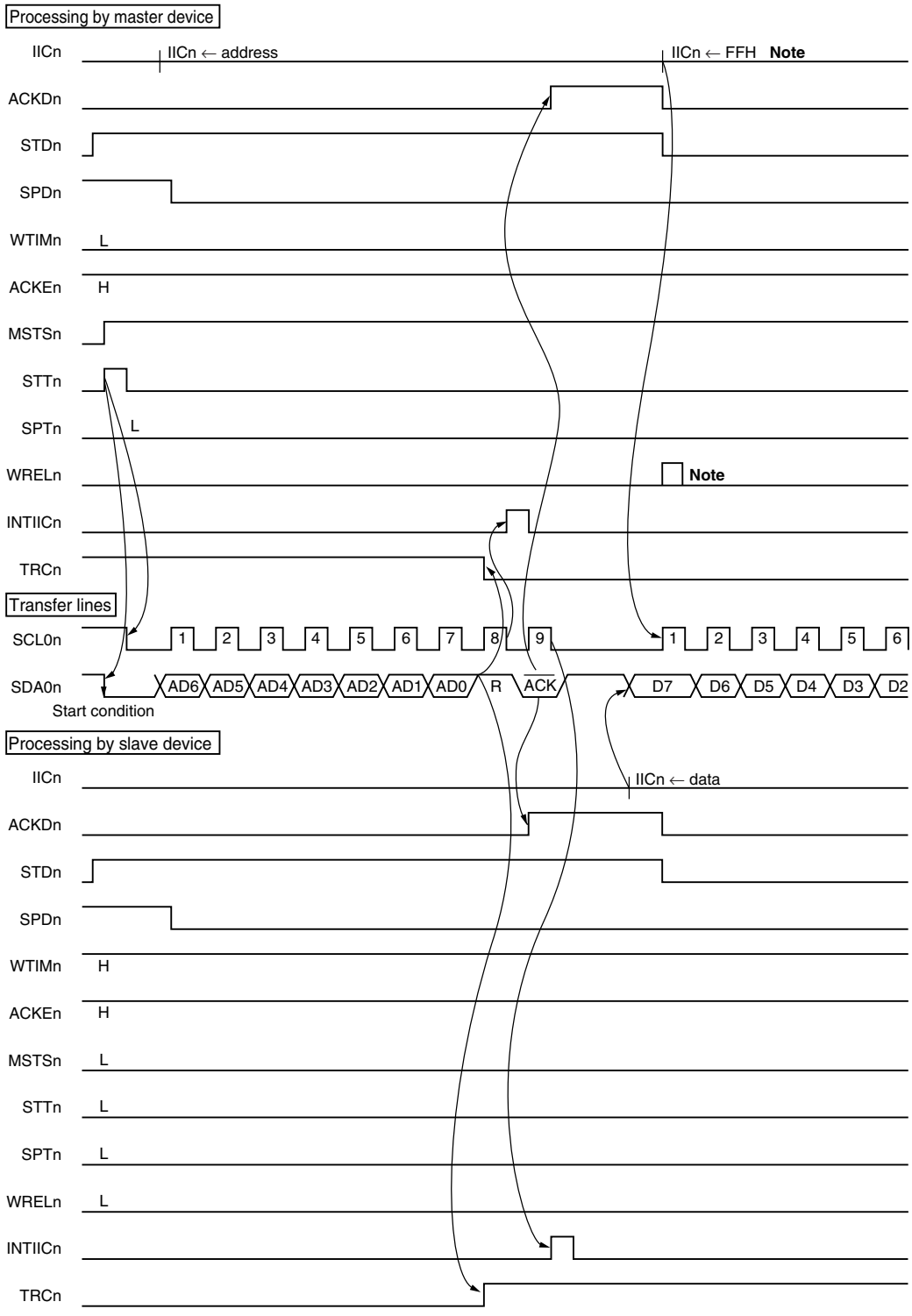
Note To cancel slave wait, write FFH to IICn or set WRELn.

Remark n = 0, 1

<R>

Figure 15-22. Example of Slave to Master Communication
(When 8-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (1/3)

(a) Start condition ~ Address



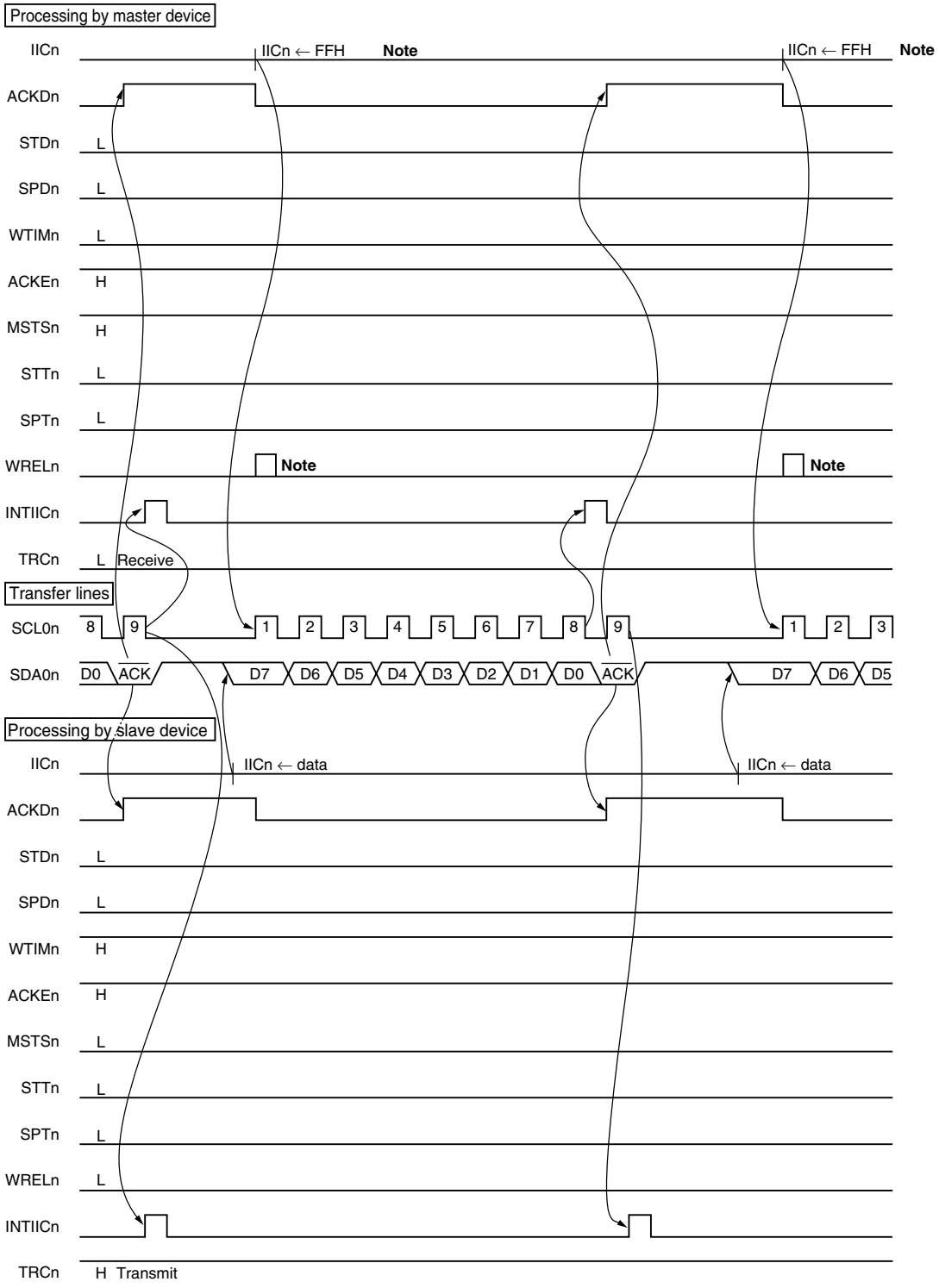
Note To cancel master wait, write FFH to IICn or set WRELn.

Remark n = 0, 1

<R>

Figure 15-22. Example of Slave to Master Communication
(When 8-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (2/3)

(b) Data



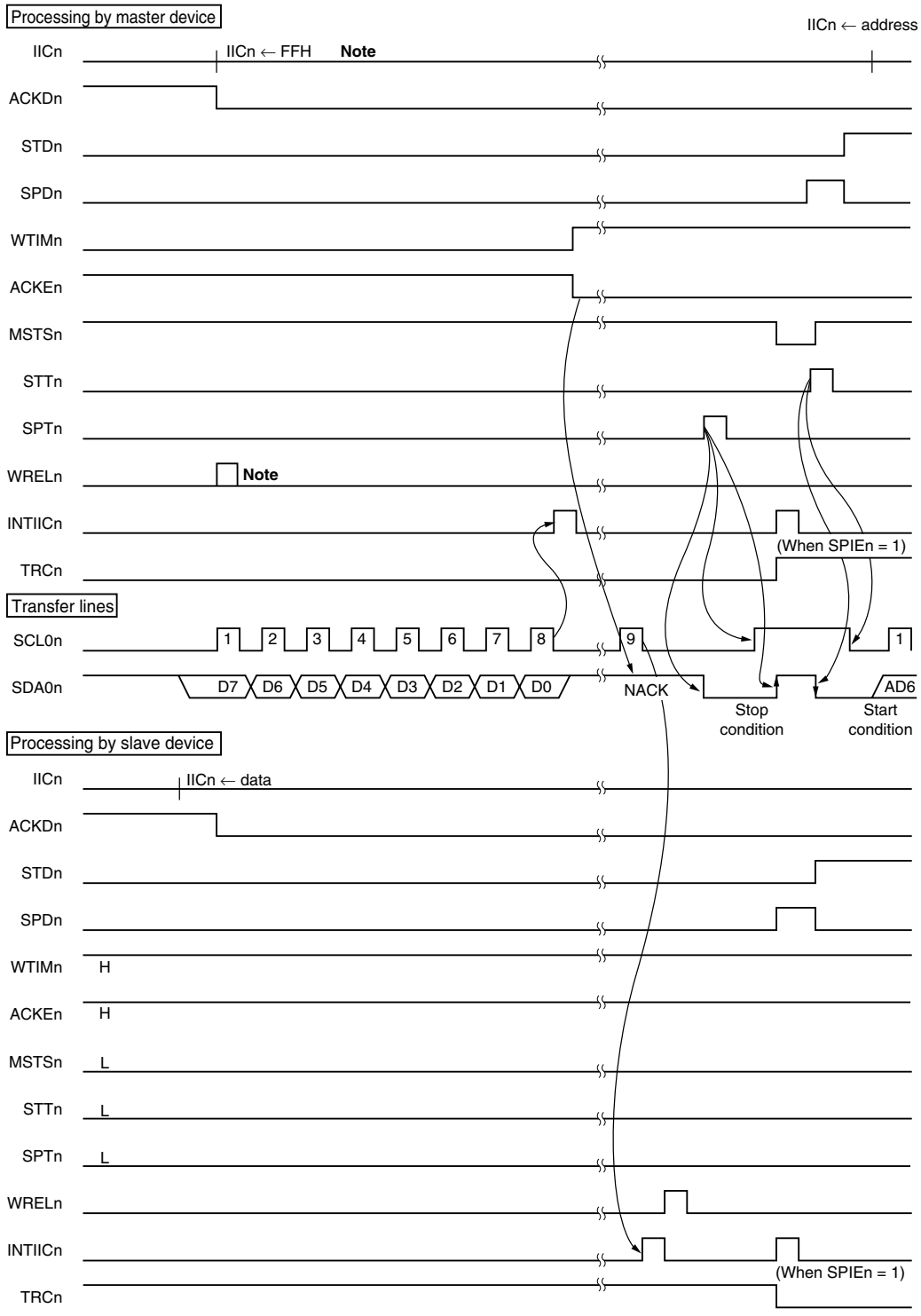
Note To cancel master wait, write FFH to IICn or set WRELn.

Remark n = 0, 1

<R>

Figure 15-22. Example of Slave to Master Communication
(When 8-Clock Wait for Master and 9-Clock Wait for Slave Are Selected) (3/3)

(c) Stop condition



Note To cancel master wait, write FFH to IICn or set WRELn.

Remark n = 0, 1

CHAPTER 16 IEBus CONTROLLER

IEBus (Inter Equipment Bus) is a small-scale digital data transfer system that transfers data between units.

The IEBus controller is provided only in the μPD703252Y.

To implement IEBus with the μPD703252Y, an external IEBus driver and receiver are necessary because they are not provided.

The internal IEBus controller of the μPD703252Y is of negative logic.

16.1 Functions

16.1.1 Communication protocol of IEBus

The communication protocol of the IEBus is as follows.

(1) Multi-task mode

All the units connected to the IEBus can transfer data to the other units.

(2) Broadcasting communication function

Communication between one unit and multiple units can be performed as follows.

- Group-unit broadcast communication: Broadcast communication to group units
- All-unit broadcast communication: Broadcast communication to all units.

(3) Effective transfer rate

The effective transfer rate is in mode 1 or mode 2 (the μPD703252Y do not support mode 0 for the effective transfer rate).

- Mode 1: Approx. 17 kbps
- Mode 2: Approx. 26 kbps

Caution Different modes must not be mixed on one IEBus.

(4) Communication mode

Data transfer is executed in half-duplex asynchronous communication mode.

(5) Access control: CSMA/CD (Carrier Sense Multiple Access with Collision Detection)

The priority of the IEBus is as follows:

<1> Broadcast communication takes precedence over individual communication (communication from one unit to another).

<2> The lower master address takes precedence.

(6) Communication scale

The communication scale of IEBus is as follows.

- Number of units: 50 MAX.
- Cable length: 150 m MAX. (when twisted pair cable is used)

Caution The communication scale in an actual system differs depending on the characteristics of the cables, etc., constituting the IEBus driver/receiver and IEBus.

16.1.2 Determination of bus mastership (arbitration)

An operation to occupy the bus is performed when a unit connected to the IEBus controls the other units. This operation is called arbitration.

When two or more units simultaneously start transmission, arbitration is used to grant one of the units the permission to occupy the bus.

Because only one unit is granted the bus mastership as a result of arbitration, the priority conditions of the bus are predetermined as follows.

Caution The bus mastership is released if communication is aborted.

(1) Priority by communication type

Broadcast communication (communication from one unit to multiple units) takes precedence over normal communication (communication from one unit to another).

(2) Priority by master address

If the communication type is the same, communication with the lower master address takes precedence.

A master address consists of 12 bits, with unit 000H having the highest priority and unit FFFH having the lowest priority.

16.1.3 Communication mode

The IEBus has three communication modes each having a different transfer rate. The μPD703252Y supports communication modes 1 and 2. The transfer rate and the maximum number of transfer bytes in one communication frame in communication modes 1 and 2 are as shown in Table 16-1.

Table 16-1. Transfer Rate and Maximum Number of Transfer Bytes in Each Communication Mode

Communication Mode	Maximum Number of Transfer Bytes (Bytes/Frame)	Effective Transfer Rate (kbps)[Note]
1	32	Approx. 17
2	128	Approx. 26

Note The effective transfer rate when the maximum number of transfer bytes is transmitted.

Select the communication mode for each unit connected to the IEBus before starting communication. If the communication mode of the master unit and that of the partner unit (slave unit) are not the same, communication is not correctly executed.

16.1.4 Communication address

With the IEBus, each unit is assigned a specific 12-bit address. This communication address consists of the following identification numbers.

- Higher 4 bits: Group number (number to identify the group to which each unit belongs)
- Lower 8 bits: Unit number (number to identify each unit in a group)

16.1.5 Broadcast communication

Normally, transmission or reception is performed between the master unit and its partner slave unit on a one-to-one basis. During broadcast communication, however, two or more slave units exist and the master unit executes transmission to these slave units. Because two or more slave units exist, the NACK signal is returned by the communicating slave unit as an acknowledge bit.

Whether broadcast communication or normal communication is to be executed is selected by the broadcast bit (for this bit, see **16.1.6 (2) Broadcast bit**).

Broadcast communication is classified into two types: group-unit broadcast communication and all-unit broadcast communication. Group-unit broadcast and all-unit broadcast are identified by the value of the slave address (for the slave address, see **16.1.6 (4) Slave address field**).

(1) Group-unit broadcast communication

Broadcast communication is performed to the units in a group identified by the group number indicated by the higher 4 bits of the communication address.

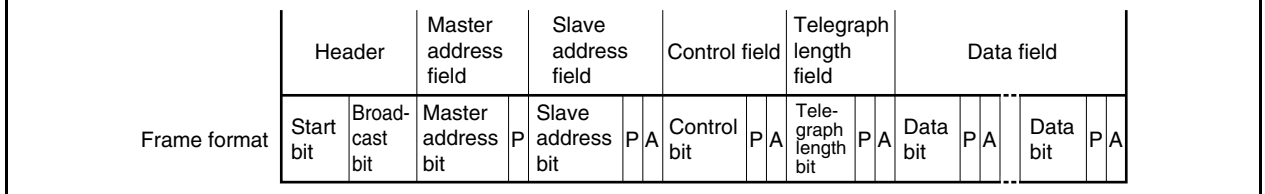
(2) All-unit broadcast communication

Broadcast communication is performed to all the units, regardless of the value of the group number.

16.1.6 Transfer format of IEBus

Figure 16-1 shows the transfer signal format of the IEBus.

Figure 16-1. IEBus Transfer Signal Format



Remarks 1. P: Parity bit

A: Acknowledge ($\overline{\text{ACK}}$/NACK) bit

2. The master unit ignores the acknowledge bit during broadcast communication.

(1) Start bit

The start bit is a signal that informs the other units of the start of data transfer. The unit that is to start data transfer outputs a high-level signal (start bit) from the $\overline{\text{IETX}}$ pin for a specific time, and then starts outputting the broadcast bit.

If another unit has already output its start bit when one unit is to output the start bit, this unit does not output the start bit but waits for completion of output of the start bit by the other unit. When the output of the start bit by the other unit is complete, the unit starts outputting the broadcast bit in synchronization with the completion of the start bit output by the other unit.

The units other than the one that has started communication detect this start bit, and enter the reception status.

(2) Broadcast bit

This bit indicates whether the master selects one slave (individual communication) or multiple slaves (broadcast communication) as the other party of communication.

When the broadcast bit is 0, it indicates broadcast communication; when it is 1, individual communication is indicated. Broadcast communication is classified into two types: group-unit communication and all-unit communication. These communication types are identified by the value of the slave address (for the slave address, see **16.1.6 (4) Slave address field**).

Because two or more slave units exist as a partner slave unit of communication in the case of broadcast communication, the NACK signal is returned as an acknowledge bit in each field subsequent to the master address field.

If two or more units start transmitting a communication frame at the same time, broadcast communication takes precedence over individual communication, and wins in arbitration.

If one unit occupies the bus as the master, the value set to the broadcast request flag (BCR.ALLRQ bit) is output.

(3) Master address field

The master address field is output by the master to inform a slave of the master's address.

The configuration of the master address field is as shown in Figure 16-2.

If two or more units start transmitting the broadcast bit at the same time, the master address field makes a judgment of arbitration.

The master address field compares the data it outputs with the data on the bus each time it has output one bit. If the master address output by the master address field is found to be different from the data on the bus as a result of comparison, it is assumed that the master has lost in arbitration. As a result, the master stops transmission and enters the reception status.

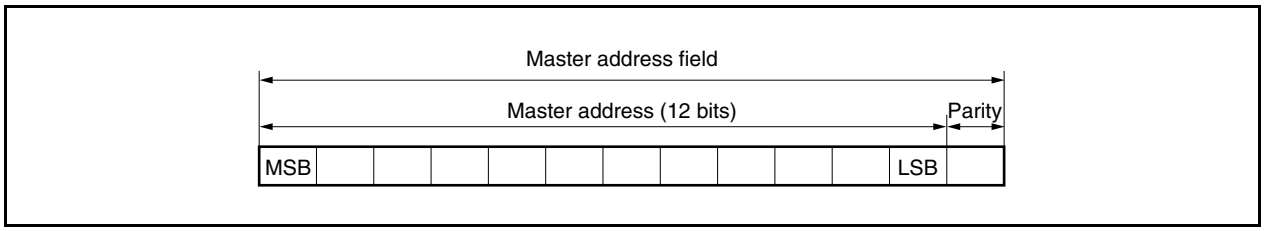
Because the IEBus is configured of wired AND, the unit having the minimum master address of the units participating in arbitration (arbitration masters) wins in arbitration.

After a 12-bit master address has been output, only one unit remains in the transmission status as one master unit.

Next, this master unit outputs a parity bit, determines the master address of other unit, and starts outputting a slave address field.

If one unit occupies the bus as the master, the address set by the UAR register is output.

Figure 16-2. Master Address Field



(4) Slave address field

The master outputs the address of the unit with which it is to communicate.

Figure 16-3 shows the configuration of the slave address field.

A parity bit is output after a 12-bit slave address has been transmitted in order to prevent a wrong slave address from being received by mistake. Next, the master unit detects an $\overline{ACK}$ signal from the slave unit to confirm that the slave unit exists on the bus. When the master has detected the $\overline{ACK}$ signal, it starts outputting the control field. During broadcast communication, however, the master does not confirm the acknowledge bit but starts outputting the control field.

The slave unit outputs the $\overline{ACK}$ signal if its slave address matches and if the slave detects that the parities of both the master address and slave address are even. The slave unit judges that the master address or slave address has not been correctly received and outputs the NACK signal if the parities are odd. At this time, the master unit is in the standby (monitor) status, and communication ends.

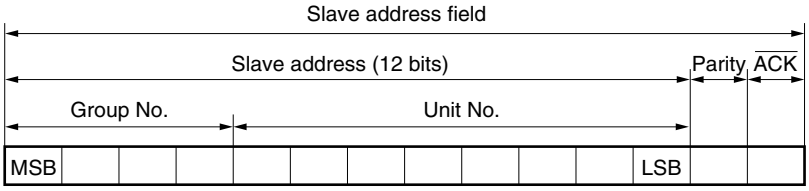
During broadcast communication, the slave address is used to identify group-unit broadcast or all-unit broadcast, as follows:.

If slave address is FFFH: All-unit broadcast communication
If slave address is other than FFFH: Group-unit broadcast communication

Remark The group No. during group-unit broadcasting communication is the value of the higher 4 bits of the slave address.

If one unit occupies the bus as the master, the address set by the SAR register is output.

Figure 16-3. Slave Address Field



(5) Control field

The master outputs the operation it requires the slave to perform, by using this field.

The configuration of the control field is as shown in Figure 16-4.

If the parity following the control bit is even and if the slave unit can execute the function required by the master unit, the slave unit outputs an $\overline{ACK}$ signal and starts outputting the telegraph length field. If the slave unit cannot execute the function required by the master unit even if the parity is even, or if the parity is odd, the slave unit outputs the NACK signal, and returns to the standby (monitor) status.

The master unit starts outputting the telegraph field after detecting the $\overline{ACK}$ signal.

If the master can detect the NACK signal, the master unit enters the standby status, and communication ends. During broadcast communication, however, the master unit does not confirm the acknowledge bit, and starts outputting the telegraph length field.

The contents of the control bits are shown below.

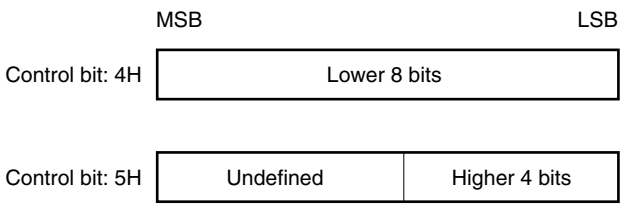
Table 16-2. Contents of Control Bits

Bit 3[Note 1]	Bit 2	Bit 1	Bit 0	Function
0	0	0	0	Read slave status
0	0	0	1	Undefined
0	0	1	0	Undefined
0	0	1	1	Read data and lock[Note 2]
0	1	0	0	Read lock address (lower 8 bits)[Note 3]
0	1	0	1	Read lock address (higher 4 bits)[Note 3]
0	1	1	0	Read slave status and unlock[Note 2]
0	1	1	1	Read data
1	0	0	0	Undefined
1	0	0	1	Undefined
1	0	1	0	Write command and lock[Note 2]
1	0	1	1	Write data and lock[Note 2]
1	1	0	0	Undefined
1	1	0	1	Undefined
1	1	1	0	Write command
1	1	1	1	Write data

Notes 1. The telegraph length bit of the telegraph length field and data transfer direction of the data field change as follows depending on the value of bit 3 (MSB).
If bit 3 is '1': Transfer from master unit to slave unit
If bit 3 is '0': Transfer from slave unit to master unit

2. This is a control bit that specifies locking or unlocking (see **16.1.7 (4) Locking and unlocking**).

3. The lock address is transferred in 1-byte (8-bit) units and is configured as follows:



If the control bit received from the master unit is not as shown in Table 16-3, the unit locked by the master unit rejects acknowledging the control bit, and outputs the NACK signal.

Table 16-3. Control Field for Locked Slave Unit

Bit 3	Bit 2	Bit 1	Bit 0	Function
0	0	0	0	Read slave status
0	1	0	0	Read lock address (lower 8 bits)
0	0	0	1	Read lock address (higher 4 bits)

Moreover, units for which lock is not set by the master unit reject acknowledgment and output a NACK signal when the control data shown in Table 16-4 is acknowledged.

Table 16-4. Control Field for Unlocked Slave Unit

Bit 3	Bit 2	Bit 1	Bit 0	Function
0	1	0	0	Lock address read (lower 8 bits)
0	1	0	1	Lock address read (higher 4 bits)

If one unit occupies the bus as the master, the value set to the CDR register is output.

Figure 16-4. Control Field

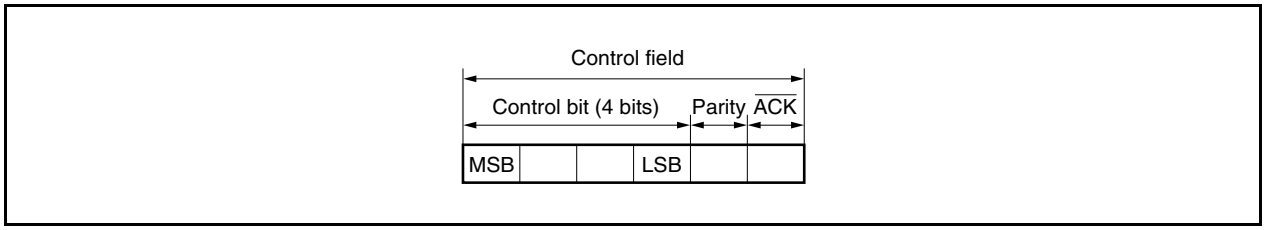


Table 16-5. Acknowledge Signal Output Condition of Control Field

(a) If received control data is AH, BH, EH, or FH

Communication Type (USR.ALLTRANS bit) Individual Communication = 0 Broadcast Communication = 1	Communication Target (USR.SLVRQ bit) Slave Specification = 1 No Specification = 0	Lock Status (USR.LOCK bit) Lock = 1 Unlock = 0	Master Unit Identification (Match with PAR register) Lock Request Unit = 1 Other = 0	Slave Transmission Enable (BCR.ENSLVTX bit)	Slave Reception Enable (BCR.ENSLVRX bit)	Received Control Data			
						AH	BH	EH	FH
0	1	0	don't care	don't care	1	○			
		1	1						
Other than above						×			

(b) If received control data is 0H, 3H, 4H, 5H, 6H, or 7H

Communication Type (USR.ALLTRANS bit) Individual Communication = 0 Broadcast Communication = 1	Communication Target (USR.SLVRQ bit) Slave Specification = 1 No Specification = 0	Lock Status (USR.LOCK bit) Lock = 1 Unlock = 0	Master Unit Identification (Match with PAR register) Lock Request Unit = 1 Other = 0	Slave Transmission Enable (BCR.ENSLVTX bit)	Slave Reception Enable (BCR.ENSLVRX bit)	Received Control Data					
						0H	3H	4H	5H	6H	7H
0	1	0	don't care	0	don't care	○	×	×	×	○	×
				1		○	○	×	×	○	○
		1	0	don't care		○	×	○	○	×	×
			1	0		○	×	○	○	○	×
				1		○	○	○	○	○	○
Other than above						×					

Caution If the received control data is other than the data shown in Table 16-5, × (NACK signal is returned) is unconditionally assumed.

Remark ○: $\overline{\text{ACK}}$ signal is returned.
×: NACK signal is returned.

(6) Telegraph length field

This field is output by the transmission side to inform the reception side of the number of bytes of the transmit data.

The configuration of the telegraph length field is as shown in Figure 16-5.

Table 16-6 shows the relationship between the telegraph length bit and the number of transmit data.

Figure 16-5. Telegraph Length Field

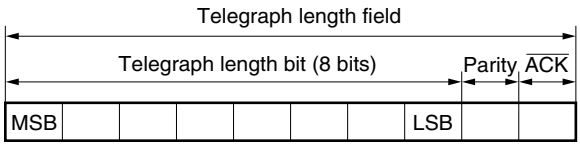


Table 16-6. Contents of Telegraph Length Bit

Telegraph Length Bit (Hex)	Number of Transmit Data Bytes
01H	1 byte
02H	2 bytes
FFH	255 bytes
00H	256 bytes

The operation of the telegraph length field differs depending on whether the master transmits data (when control bit 3 is 1) or receives data (when control bit 3 is 0).

(a) When master transmits data

The telegraph length bit and parity bit are output by the master unit and the synchronization signals of bits are output by the master unit. When the slave unit detects that the parity is even, it outputs the $\overline{\text{ACK}}$ signal, and starts outputting the data field. During broadcast communication, however, the slave unit outputs the NACK signal.

If the parity is odd, the slave unit judges that the telegraph length bit has not been correctly received, outputs the NACK signal, and returns to the standby (monitor) status. At this time, the master unit also returns to the standby status, and communication ends.

(b) When master receives data

The telegraph length bit and parity bit are output by the slave unit and the synchronization signals of bits are output by the master unit. If the master unit detects that the parity bit is even, it outputs the $\overline{\text{ACK}}$ signal.

If the parity bit is odd, the master unit judges that the telegraph length bit has not been correctly received, outputs the NACK signal, and returns to the standby status. At this time, the slave unit also returns to the standby status, and communication ends.

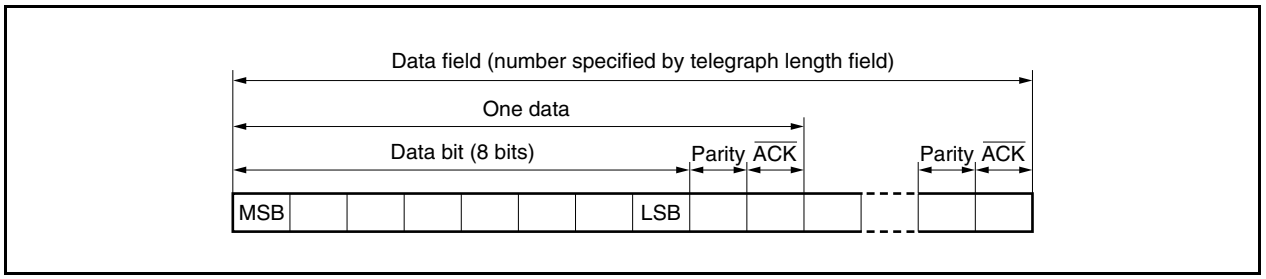
(7) Data field

This is data output by the transmission side.

The master unit transmits or receives data to or from a slave unit by using the data field.

The configuration of the data field is as shown below.

Figure 16-6. Data Field



Following the data bit, the parity bit and acknowledge bit are respectively output by the master unit and slave unit.

Use broadcast communication only for when the master unit transmits data. At this time, the acknowledge bit is ignored.

The operation differs as follows depending on whether the master transmits or receives data.

(a) When master transmits data

When the master units writes data to a slave unit, the master unit transmits the data bit and parity bit to the slave unit. If the parity is even and the receive data is not stored in the DR register when the slave unit has received the data bit and parity bit, the slave unit outputs an $\overline{ACK}$ signal. If the parity is odd or if the receive data is stored in the DR register, the slave unit rejects receiving the data, and outputs the NACK signal.

If the slave unit outputs the NACK signal, the master unit transmits the same data again. This operation continues until the master detects the $\overline{ACK}$ signal from the slave unit, or the data exceeds the maximum number of transmit bytes.

If there is more data and the maximum number of transmit bytes is not exceeded when the parity is even and when the slave unit outputs the $\overline{ACK}$ signal, the master unit transmits the next data.

During broadcast communication, the slave unit outputs the NACK signal, and the master unit transfers 1 byte of data at a time. If the parity is odd or the DR register is storing receive data after the slave unit has received the data bit and parity bit during broadcast communication, the slave unit judges that reception has not been performed correctly, and stops reception.

(b) When master receives data

When the master unit reads data from a slave unit, the master unit outputs a sync signal corresponding to all the read bits.

The slave unit outputs the contents of the data and parity bits to the bus in response to the sync signal from the master unit.

The master unit reads the data and parity bits output by the slave unit, and checks the parity.

If the parity is odd, or if the DR register is storing a receive data, the master unit rejects accepting the data, and outputs the NACK signal. If the maximum number of transmit bytes is within the value that can be transmitted in one communication frame, the master unit repeats reading the same data.

If the parity is even and the DR register is not storing a receive data, the master unit accepts the data and outputs the $\overline{\text{ACK}}$ signal. If the maximum number of transmit bytes is within the value that can be transmitted in one frame, the master unit reads the next data.

Caution Do not operate master reception in broadcast communication, because the slave unit cannot be defined and data transfer cannot be performed correctly.

(8) Parity bit

The parity bit is used to check to see if the transmit data has no error.

The parity bit is appended to each data of the master address, slave address, control, telegraph length, and data bits.

The parity is an even parity. If the number of bits in data that are '1' is odd, the parity bit is '1'. If the number of bits in the data that are '1' is even, the parity bit is '0'.

(9) Acknowledge bit

During normal communication (communication from one unit to another), an acknowledge bit is appended to the following locations to check if the data has been correctly received.

- End of slave address field
- End of control field
- End of telegraph length field
- End of data field

The definition of the acknowledge bit is as follows.

- 0: Indicates that the transmit data is recognized ($\overline{\text{ACK}}$ signal).
- 1: Indicates that the transmit data is not recognized (NACK signal).

During broadcast communication, however, the contents of the acknowledge bit are ignored.

(a) Last acknowledge bit of slave field

The last acknowledge bit of the slave field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the master address bit or slave address bit is incorrect
- If a timing error (error in bit format) occurs
- If a slave unit does not exist

(b) Last acknowledge bit of control field

The last acknowledge bit of the control field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the control bit is incorrect
- If control bit 3 is '1' (write operation) when the slave reception enable flag (BCR.ENSLVRX bit) is not set (1)[Note]
- If the control bit indicates reading of data (3H or 7H) when the slave transmission enable flag (BCR.ENSLVTX bit) is not set (1)
- If a unit other than that has set locking requests 3H, 6H, 7H, AH, BH, EH, or FH of the control bit when locking is set
- If the control bit indicates reading of lock addresses (4H, 5H) even when locking is not set
- If a timing error occurs
- If the control bit is undefined

Cautions 1. The $\overline{\text{ACK}}$ signal is always returned when the control data of the slave status request is received, even if the ENSLVTX bit = 0.

2. The NACK signal is returned by the acknowledge bit in the control field when the control data for data/command writing is received, even if the ENSLVRX bit = 0.
Slave reception can be disabled (communication stopped) by ENSLVRX bit only in the case of individual communication. In the case of broadcast communication, communication is maintained and the data request interrupt request signal (INTIE1) or IEBus end interrupt request signal (INTIE2) is generated.

(c) Last acknowledge bit of telegraph length field

The last acknowledge bit of the telegraph length field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the telegraph length bit is incorrect
- If a timing error occurs

(d) Last acknowledge bit of data field

The last acknowledge bit of the data field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the data bit is incorrect[Note]
- If a timing error occurs after the preceding acknowledge bit has been transmitted
- If the receive data is stored in the DR register and no more data can be received[Note]

Note In this case, when the communication executed is individual communication, if the maximum number of transmit bytes is within the value that can be transmitted in one frame, the transmission side executes transmission of that data field again. For broadcast communication, the transmission side does not execute transmission again, a communication error occurs on the reception side and reception stops.

16.1.7 Transfer data

(1) Slave status

The master unit can learn why the slave unit did not return the $\overline{ACK}$ signal by reading the slave status.

The slave status is determined according to the result of the last communication the slave unit has executed.

All the slave units can supply information on the slave status.

The configuration of the slave status is shown below.

Figure 16-7. Bit Configuration of Slave Status

MSB LSB

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 0[Note 1]	Function
0	Transmit data is not written in DR register
1	Transmit data is written in DR register

Bit 1[Note 2]	Function
0	Receive data is not stored in DR register
1	Receive data is stored in DR register

Bit 2	Function
0	Unit is not locked
1	Unit is locked

Bit 3	Function
0	Fixed to 0

Bit 4[Note 3]	Function
0	Slave transmission is stopped
1	Slave transmission is ready

Bit 5	Function
0	Fixed to 0

Bit 7	Bit 6	Function	
0	0	Mode 0	Indicates the highest mode supported by the unit[Note 4].
0	1	Mode 1	
1	0	Mode 2	
1	1	Not used	

Notes 1. After reset: Bit 0 is set to 1.

2. The receive buffer size is 1 byte.

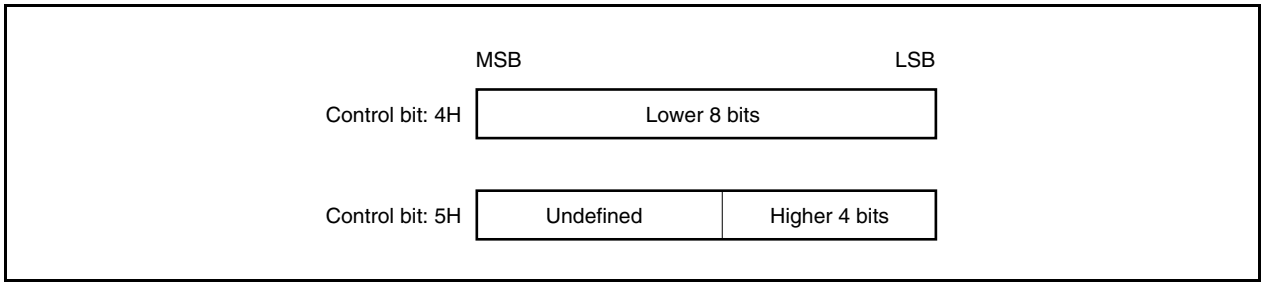
3. When the V850ES/SG1 serves as a slave unit, this bit corresponds to the status indicated by BCR.ENSLVTX bit.

4. Bits 7 and 6 are fixed to “10” because the μPD703252Y can support modes 1 and 2.

(2) Lock address

When the lock address is read (control bit: 4H or 5H), the address (12 bits) of the master unit that has issued the lock instruction is configured in 1-byte units as shown below and read.

Figure 16-8. Configuration of Lock Address



(3) Data

If the control bit indicates reading of data (3H or 7H), the data in the data buffer of the slave unit is read by the master unit.

If the control bit indicates writing of data (BH or FH), the data received by the slave unit is processed according to the operation rule of that slave unit.

(4) Locking and unlocking

The lock function is used when a message is transferred in two or more communication frames.

The unit that is locked does not receive data from units other than the one that has locked the unit (does not receive broadcast communication).

A unit is locked or unlocked as follows.

(a) Locking

If the communication frame is completed without succeeding to transmit or receive data of the number of bytes specified by the telegraph length bit after the telegraph length field has been transmitted or received ($\overline{ACK}$ = 0) by the control bit that specifies locking (3H, AH, or BH), the slave unit is locked by the master unit. At this time, the bit (bit 2) in the byte indicating the slave status is set to '1'.

(b) Unlocking

After transmitting or receiving data of the number of data bytes specified by the telegraph length bit in one communication frame by the control bit that has specified locking (3H, AH, or BH), or the control bit that has specified unlocking (6H), the slave unit is unlocked by the master unit. At this time, the bit related to locking (bit 2) in the byte indicating the slave status is reset to '0'.

Locking or unlocking is not performed during broadcast communication.

Locking and unlocking conditions are shown below.

Table 16-7. Lock Setting Conditions

Control Data	Broadcast Communication		Individual Communication	
	Communication End	Frame End	Communication End	Frame End
3H, 6H[Note]			Cannot be locked	Lock set
AH, BH	Cannot be locked	Cannot be locked	Cannot be locked	Lock set
0H, 4H, 5H, EH, FH	Cannot be locked	Cannot be locked	Cannot be locked	Cannot be locked

Note The frame end of control data 6H (slave status read/unlock) occurs when the parity in the data field is odd, and when the NACK signal from the IEBus unit is repeated up to the maximum number of transfer bytes with being output.

Table 16-8. Unlock Release Conditions (While Locked)

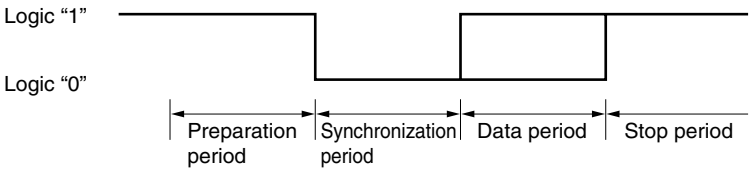
Control Data	Broadcast Communication from Lock Request Unit		Individual Communication from Lock Request Unit	
	Communication End	Frame End	Communication End	Frame End
3H, 6H[Note]			Unlocked	Remains locked
AH, BH	Unlocked	Unlocked	Unlocked	Remains locked
0H, 4H, 5H, EH, FH	Remains locked	Remains locked	Remains locked	Remains locked

Note The frame end of control data 6H (slave status read/unlock) occurs when the parity in the data field is odd, and when the NACK signal from the IEBus unit is repeated up to the maximum number of transfer bytes with being output.

16.1.8 Bit format

The format of the bits constituting the communication frame of the IEBus is shown below.

Figure 16-9. Bit Format of IEBus



Preparation period: First low-level (logic “1”) period
Synchronization period: Next high-level (logic “0”) period
Data period: Period indicating value of bit
Stop period: Last low-level (logic “1”) period

The synchronization period and data period are almost equal to each other in length.

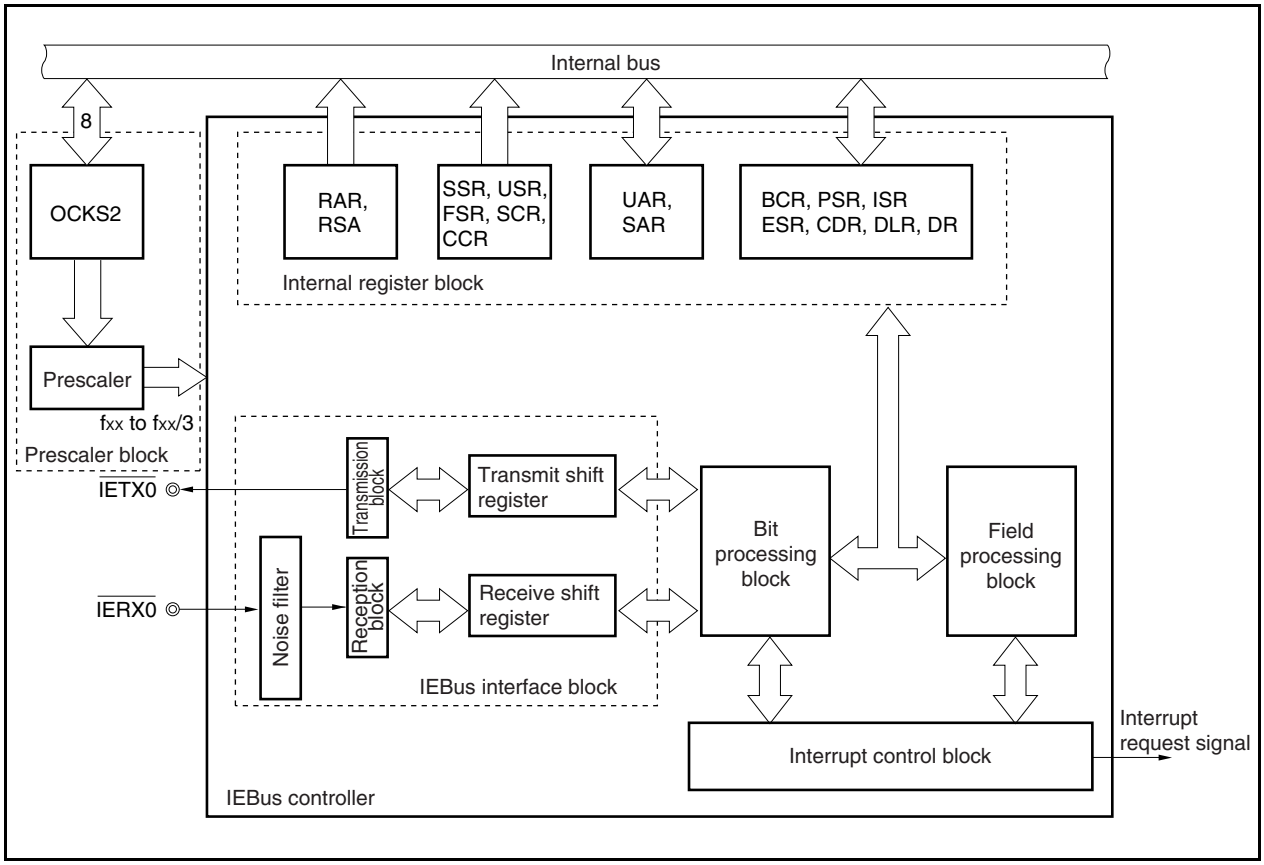
The IEBus synchronizes each bit. The specifications on the time of the entire bit and the time related to the period allocated to that bit differ depending on the type of the transmit bit, or whether the unit is the master unit or a slave unit.

The master and slave units monitor whether each period (preparation period, synchronization period, data period, and stop period) is output for specified time while they are in communication. If a period is not output for the specified time, the master and slave units report a timing error, immediately terminate communication and enter the standby status.

16.2 Configuration

The block diagram of the IEBus controller is shown below.

Figure 16-10. IEBus Controller Block Diagram



(1) Hardware configuration and functions

IEBus mainly consists of the following six internal blocks.

- Interrupt control block
- Internal registers
- Bit processing block
- Field processing block
- IEBus interface block
- Prescaler block

(a) Interrupt control block

This control block transfers interrupt request signals from the IEBus controller to the CPU.

(b) Internal registers

These registers set data to the control registers and fields that control IEBus (for the internal registers, see **16.3 Registers**).

(c) Bit processing block

This block generates and breaks down bit timing, and mainly consists of a bit sequence ROM, 8-bit preset timer, and comparator.

(d) Field processing block

This block generates each field in the communication frame, and mainly consists of a field sequence ROM, 4-bit down counter, and comparator.

(e) IEBus interface block

This is the interface block for an external driver/receiver, and mainly consists of a noise filter, shift register, and transmission/reception block (collision detector, parity detector, parity generator, and $\overline{\text{ACK}}$/NACK generator).

(f) Prescaler block

This block selects the clock to be supplied to the IEBus controller.

16.3 Registers

The registers that control the IEBus controller are shown below.

Table 16-9. Control Registers of IEBus Controller

Address	Function Register Name	Symbol	R/W	Bit Unit for Manipulation			After Reset
				1	8	16	
FFFFF348H	IEBus clock select register	OCKS2	R/W		√		00H
FFFFF360H	IEBus control register	BCR		√	√		
FFFFF361H	IEBus power save register	PSR		√	√		
FFFFF362H	IEBus slave status register	SSR	R	√	√		81H
FFFFF363H	IEBus unit status register	USR		√	√		00H
FFFFF364H	IEBus interrupt status register	ISR	R/W	√	√		
FFFFF365H	IEBus error status register	ESR		√	√		
FFFFF366H	IEBus unit address register	UAR				√	0000H
FFFFF368H	IEBus slave address register	SAR				√	
FFFFF36AH	IEBus partner address register	PAR	R			√	
FFFFF36CH	IEBus receive slave address register	RSA				√	
FFFFF36EH	IEBus control data register	CDR	R/W		√		00H
FFFFF36FH	IEBus telegraph length register	DLR			√		01H
FFFFF370H	IEBus data register	DR			√		00H
FFFFF371H	IEBus field status register	FSR	R		√		
FFFFF372H	IEBus success count register	SCR			√		01H
FFFFF373H	IEBus communication count register	CCR			√		20H

(1) IEBus control register (BCR)

The BCR register is an 8-bit register that controls the operations of the IEBus controller.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF360H

	<7>	<6>	<5>	<4>	<3>	2	1	0
BCR	ENIEBUS	MSTRQ	ALLRQ	ENSLVTX	ENSLVRX	0	0	0

ENIEBUS	Communication enable flag
0	IEBus unit stopped
1	IEBus unit active

MSTRQ	Master request flag
0	IEBus unit not requested as master
1	IEBus unit requested as master

ALLRQ	Broadcast request flag
0	Individual communication requested
1	Broadcast communication requested

ENSLVTX	Slave transmission enable flag
0	Slave transmission disabled
1	Slave transmission enabled

ENSLVRX	Slave reception enable flag
0	Slave reception disabled
1	Slave reception enabled

Cautions 1. While IEBus is operating as the master, writing to the BCR register (including bit manipulation instructions) is disabled until either the end of that communication or frame, or until communication is stopped by the occurrence of an arbitration-loss communication error. Master requests cannot therefore be multiplexed. However, the case when communication has been forcibly stopped (ENIEBUS flag = 0) is not problem.

2. If a bit manipulation instruction for the BCR register conflicts with a hardware reset of the MSTRQ bit, the BCR register may not operate normally. The following countermeasures are recommended in this case.

- **Because the hardware reset is instigated in the acknowledgment period of the slave address field, be sure to observe Caution 1 of (b) Master request flag (MSTRQ) below.**
- **Be sure to observe the caution above regarding writing to the BCR register.**

(a) Communication enable flag (ENIEBUS)...Bit 7

<Set/clear conditions>

Set: By software

Clear: By software

The IEBus controller participates in communication differently depending on the timing of setting the ENIEBUS bit (1), as follows.

Table 16-10. Timing of Setting ENIEBUS Bit and Participation in Communication

Timing of Setting ENIEBUS Bit	How IEBus Controller Participates in Communication
If communication is not performed on IEBus	Participates in communication from the next frame or starts communication.
If other bus master is communicating start bit while communication is in progress on IEBus	Participates in communication from that frame if the start bit is detected. If the start bit is not detected, participates in communication from the next frame.
If communication is in progress on IEBus after start bit from other bus master is detected	Participates in communication from the next frame.

If the ENIEBUS bit is cleared (0), communication is immediately stopped even while it is in progress, and the internal flags and registers are reset, with some exceptions. The registers that are not reset by the ENIEBUS bit are listed in the table below.

The IEBus controller does not respond even if another unit starts communication when the ENIEBUS bit = 0.

Table 16-11. Registers That Are Not Reset by ENIEBUS Bit

Registers Not Reset by ENIEBUS Bit	Remark
UAR	Not reset
SAR	Not reset
CDR	Data written from CPU is not reset but data received during communication is reset.
DLR	Data written from CPU is not reset but data received during communication is reset.
DR	Data written from CPU is not reset but data received during communication is reset.

Caution Before setting the ENIEBUS bit (1), the following registers must be set depending on the mode of communication to be started.

Table 16-12. Registers That Must Be Set Before Each Communication

Mode of Communication	Registers That Must Be Set in Advance
Master transmission	UAR, SAR, CDR, DLR, DR (first 1-byte data)
Master reception	UAR, SAR, CDR
Slave transmission[Note]	UAR, DLR, DR (first 1-byte data)[Note]
Slave reception	UAR

Note When starting slave transmission, information such as the value to be set to the DLR register and which data is to be returned (value to be set to the DR register) must be assigned in advance.

(b) Master request flag (MSTRQ)...Bit 6

<Set/clear conditions>

Set: By software

Clear: Cleared (0) by hardware when master communication is started and immediately before the start interrupt of the master occurs.

Cleared (0) by hardware before a communication error occurs.

When the ENIEBUS bit is cleared.

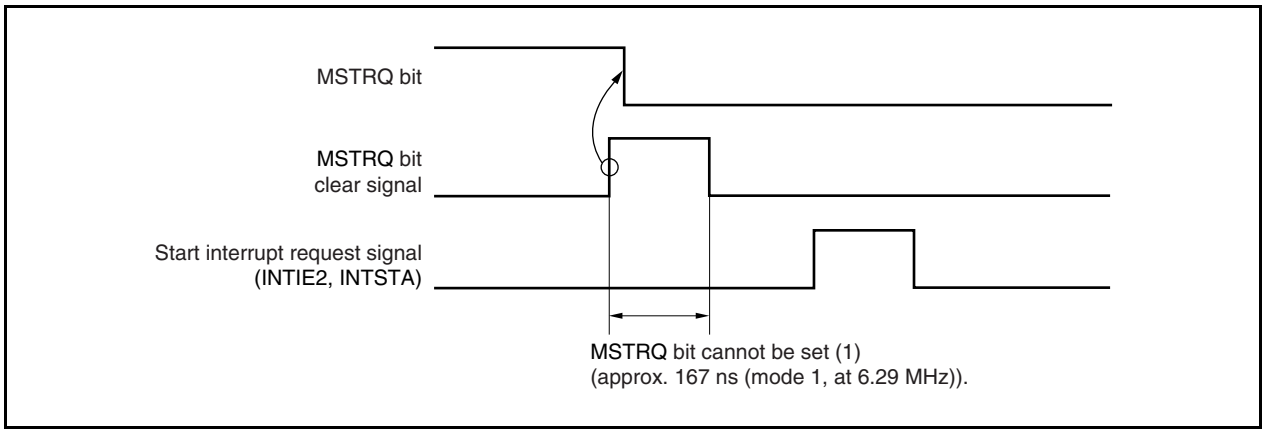
When the MSTRQ bit is set (1), the IEBus controller starts communication on IEBus as the master.

If communication is in progress on IEBus (if the start bit cannot be detected while the start bit is being communicated or if communication is in progress after the start bit has been detected), however, the controller waits until the current frame ends (holds the master request pending), outputs the start bit after the frame has ended, and starts communication as the master.

Cautions 1. If the IEBus controller has lost in arbitration, issue the master request again by software.

In doing so, set (1) the MSTRQ bit at a timing other than that illustrated below.

Figure 16-11. Timing at Which MSTRQ Bit Cannot Be Set



2. When a master request has been sent and bus mastership acquired, do not set the MSTRQ, ENSLVTX, or ENSLVRX bit until the end of communication (i.e. the communication end flag (ISR.ENDTRNS bit) or frame end flag (ISR.ENDFRAM bit) is set (1)) as setting these flags disables interrupt request signal generation. However, these flags can be set if communication has been aborted.

(c) Broadcast request flag (ALLRQ)...Bit 5

<Set/clear conditions>

Set: By software

Clear: By software

Caution When requesting broadcast communication, always set (1) the ALLRQ bit, then the MSTRQ bit.

(d) Slave transmission enable flag (ENSLVTX)...Bit 4

<Set/clear conditions>

Set: By software

Clear: By software

Cautions 1. The ENSLVTX bit must be set before the parity bit in the control field is received.

2. Clear the ENSLVTX bit (0) before setting the MSTRQ bit (1) when making a master request. This is to avoid transmission of the data of the DR register that tries master transmission if the controller loses in arbitration after master operation and if slave transmission is requested by the master.

3. When returning to an enabled state from a disabled state, transmission becomes valid from the next frame.

4. If control data (3H or 7H) for data/command writing is received when the ENSLVTX bit = 0, the NACK signal is returned by the acknowledge bit in the control field.

5. The status interrupt request signals (INTIE2, INTSTA) will be generated and communication continued when the control data of a slave status request is returned, even if the ENSLVTX bit = 0.

(e) Slave reception enable flag (ENSLVRX)...Bit 3

<Set/clear conditions>

Set: By software

Clear: By software

Cautions 1. The ENSLVRX bit must be set before the parity bit in the control field is received.

2. While the CPU is busy with other processing, slave reception can be prevented by clearing the ENSLVRX bit (0). During individual communication, the NACK signal is returned in the control field and communication is completed. During broadcast communication, communication cannot be completed because the acknowledge bit is ignored. However, the IEBus controller does not respond to the broadcast communication and does not generate an interrupt request signal.

3. When returning to an enabled state from a disabled state, transmission becomes valid from the next frame.

(2) IEBus power save register (PSR)

The PSR register is an 8-bit register that controls the internal clock and communication mode of the IEBus controller.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF361H

	<7>	<6>	5	4	3	2	1	0
PSR	ENCLK	IEMODE	0	0	0	0	0	0

ENCLK	Internal clock operation enable flag
0	Stop internal clock of IEBus controller
1	Enable internal clock of IEBus controller

IEMODE	IEBus communication mode setting flag
0	Set communication mode 1
1	Set communication mode 2

Cautions 1. Do not set the PSR register while communication is enabled (BCR.ENIEBUS bit = 1).

2. Be sure to clear bits 5 to 0 to "0".

(3) IEBus slave status register (SSR)

The SSR register is an 8-bit register that indicates the communication status of the slave unit. After receiving a slave status transmission request from the master, read this register by software, and write a slave status to the DR register to transmit the slave status. At this time, the telegraph length is automatically set to “01H”, so setting of the DLR register is not required (because it is preset by hardware).

Bits 6 and 7 indicate the highest mode supported by the unit, and are fixed to “10” (mode 2).

This register is read-only, in 8-bit or 1-bit units.

Reset sets this register to 81H.

After reset: 81H R Address: FFFFF362H

	7	6	5	<4>	3	<2>	<1>	<0>
SSR	1	0	0	STATSLV	0	STATLOCK	STATRX	STATTX

STATSLV	Slave transmission status flag
0	Slave transmission stops
1	Slave transmission enabled

STATLOCK	Lock status flag
0	Unlock status
1	Lock status

STATRX	DR register receive status
0	Receive data not stored in DR register
1	Receive data stored in DR register

STATTX	DR register transmit status
0	Transmit data not stored in DR register
1	Transmit data stored in DR register

(a) Slave transmission status flag (STATSLV)...Bit 4

Reflects the contents of the slave transmission enable flag (BCR.ENSLVTX bit).

(b) Lock status flag (STATLOCK)...Bit 2

Reflects the contents of the lock flag (USR.LOCK bit).

(c) DR register reception status (STATRX)...Bit 1

This flag indicates the DR register reception state.

(d) DR register transmission status (STATTX)...Bit 0

This flag indicates the DR register transmission state.

(4) IEBus unit status register (USR)

The USR register is an 8-bit register that indicates the IEBus unit status.

This register is read-only, in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R Address: FFFFF363H

	7	<6>	<5>	<4>	<3>	2	1	0
USR	0	SLVRQ	ARBIT	ALLTRNS	ACK	LOCK	0	0

SLVRQ	Slave request flag
0	No request from master to slave
1	Request from master to slave

ARBIT	Arbitration result flag
0	Arbitration loss not occurred
1	Arbitration loss occurred

ALLTRNS	Broadcast communication flag
0	Individual communication status
1	Broadcast communication status

ACK	Acknowledge transmission flag
0	NACK signal transmitted
1	$\overline{\text{ACK}}$ signal transmitted

LOCK	Lock status flag
0	Unit unlocked
1	Unit locked

(a) Slave request flag (SLVRQ)...Bit 6

A flag indicating whether there has been a slave request from the master.

<Set/clear conditions>

Set: When the unit is requested as a slave (if the condition in **Table 16-13 Slave Request Condition (SLVRQ Bit Setting Condition**) is satisfied), this flag is set (1) by hardware when the acknowledge period of the slave address field starts.

Clear: This flag is cleared (0) by hardware when the unit is not requested as a slave (if the condition in **Table 16-13 Slave Request Condition (SLVRQ Bit Setting Condition**) is not satisfied). The reset timing is the same as the set timing. If the unit is requested as a slave immediately after communication has been correctly received (when the SLVRQ bit = 1), and if a parity error occurs in the slave address field for that communication, the flag is not cleared.

Table 16-13. Slave Request Condition (SLVRQ Bit Setting Condition)

Status of Unit	Received Master Address	Communication Mode	Received Slave Address
Not locked	don't care	Individual	UAR register matching
		Broadcast	Group matching
			FFFH
Locked	Locked master matching	Individual	UAR register matching
		Broadcast	Group matching
			FFFH

Caution If a unit other than the locked master communicates with the unit while the unit is locked, the SLVRQ bit is not set but the $\overline{\text{ACK}}$ signal is returned to the slave address field. This is because communication must be continued, even if a unit other than the locked master returns the signal, if the control data is a slave status request.

(b) Arbitration result flag (ARBIT)...Bit 5

A flag that indicates the result of arbitration.

<Set/clear conditions>

Set: This flag is set (1) when the data output by the IEBus unit during the arbitration period does not match the bus line data.

Clear: This flag is cleared (0) by the start bit timing.

Cautions 1. The timing at which the arbitration result flag (ARBIT bit) is cleared differs depending on whether the unit outputs a start bit.

- **If start bit is output: The flag is cleared at the output start timing.**
- **If start bit is not output: The flag is cleared at the detection timing of the start bit (approx. 160 μs (mode 1, at 6.29 MHz) after output)**

2. The flag is cleared (0) at the detection timing of the start bit if the other unit outputs the start bit earlier and the unit does not output the start bit after the master request.

(c) Broadcast communication flag (ALLTRNS)...Bit 4

Flag indicating whether the unit is performing broadcast communication. The contents of the flag are updated in the broadcast field of each frame.

Except for initialization (reset) by system reset, the set/clear conditions vary depending on the receive data of the broadcast field bit.

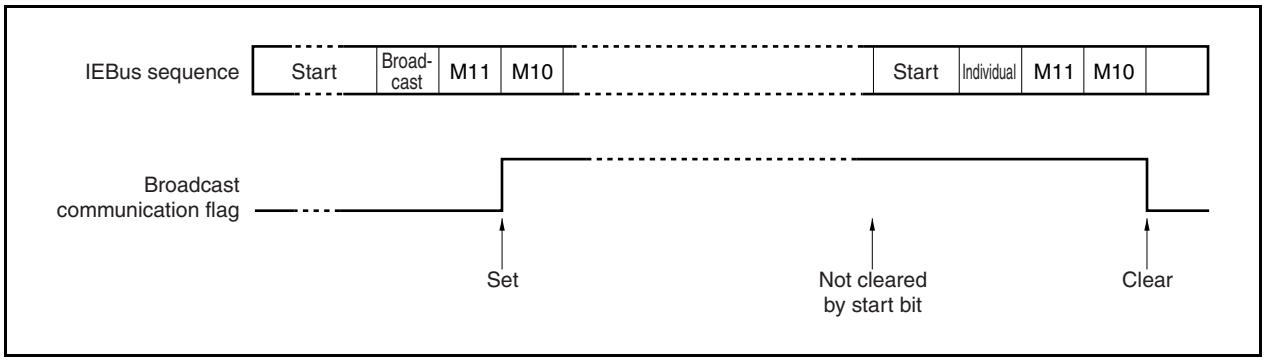
<Set/clear conditions>

Set: When “broadcast” is received by the broadcast field

Clear: When “individual” is received by the broadcast field, or upon the input of a system reset.

Caution The broadcast flag is updated regardless of whether IEBus is the communication target or not.

Figure 16-12. Example of Broadcast Communication Flag Operation



(d) Acknowledge transmission flag (ACK)...Bit 3

A flag that indicates whether the $\overline{ACK}$ signal has been transmitted in the acknowledge bit period of the acknowledge bit field when IEBus is the receiving unit. The contents of the flag are updated in the acknowledge bit period of each frame. However, if the internal circuit is initialized by the occurrence of a parity error, etc., the contents are not updated in the acknowledge bit period of that field.

(e) Lock status flag (LOCK)...Bit 2

A flag that indicates whether the unit is locked.

<Set/clear conditions>

Set: This flag is set (1) when the communication end flag (ISR.ENDTRNS bit) goes low and the frame end flag (ISR.ENDFRAM bit) goes high after receipt of a lock specification (3H, 6H, AH, BH) in the control field.

Clear: When the communication enable flag (BCR.ENIEBUS bit) is cleared (0).

When the communication end flag (ENDTRNS bit) is set (1) after receipt of a lock release (3H, 6H, AH, BH) in the control field.

Caution Lock specification/release is not possible in broadcast communication. In the lock status, individual communication from a unit other than the one that requests locking is not acknowledged. However, even communication from a unit other than the one that requests locking is acknowledged as long as the communication is a slave status request.

(5) IEBus interrupt status register (ISR)

The ISR register indicates the interrupt source when IEBus issues an interrupt request signal. This register is read to generate an interrupt request signal, after which the specified interrupt processing is carried out.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W[Note 1] Address: FFFFF364H

	7	<6>	<5>	<4>	<3>	<2>	1	0
ISR	0	IEERR	STARTF	STATUSF	ENDTRNS	ENDFRAM	0	0

IEERR	Communication error flag (during communication)
0	No communication error
1	Communication error

STARTF	Start interrupt flag
0	Start interrupt request signal did not occur
1	Start interrupt request signal occurred

STATUSF	Status transmission flag (slave)
0	No slave status/lock address (higher 4 bits, lower 8 bits) transmission request
1	Slave status/lock address (higher 4 bits, lower 8 bits) transmission request

ENDTRNS	Communication end flag
0	Communication does not end after the number of bytes set in the telegraph length field have been transferred
1	Communication ends after the number of bytes set in the telegraph length field have been transferred

ENDFRAM	Frame end flag
0	The frame (transfer of the maximum number of bytes) does not end
1	The frame (transfer of the maximum number of bytes) ends

Notes 1. Only the IEERR bit can be written, and only to 0 (i.e., the IEERR bit can only be cleared). The IEERR bit is not set (1) even if 1 is written to it.

2. Mode 1: 32 bytes
Mode 2: 128 bytes

(a) Communication error flag (IEERR)...Bit 6

A flag that indicates a communication error has occurred. When a communication error occurs, the INTIE2 and INTERR interrupt request signals are generated.

<Set/clear conditions>

Set: The flag is set (1) if a timing error, parity error (except in the data field), NACK reception error (except in the data field), underrun error, overrun error (that occurs during broadcast communication reception), or write error occurs.

Clear: By software

(b) Start interrupt flag (STARTF)...Bit 5

A flag that indicates the start interrupt. When a start interrupt occurs, the INTIE2 and INTSTA interrupt request signals are generated.

<Set/clear conditions>

Set: This flag is set (1) in the slave address field, upon a master request.
When IEBus is a slave unit, this flag is set (1) upon a request from the master (only if it was a slave request in the locked state from the unit requesting a lock).

Clear: This flag is cleared (0) if the status transmission interrupt, communication end interrupt, frame end interrupt, or INTIE1 interrupt request signal is generated.

(c) Status transmission flag (STATUSF)...Bit 4

A flag that indicates the master requested transmission of the slave status and lock address (higher 4 bits and lower 8 bits) when the controller was serving as a slave.

<Set/clear conditions>

Set: This flag is set (1) when 0H, 4H, 5H, or 6H is received in the control field from the master when the IEBus is a slave unit.

Clear: This flag is cleared (0) if the start interrupt, communication end interrupt, frame end interrupt, or INTIE1 interrupt request signal is generated.

(d) Communication end flag (ENDTRANS)...Bit 3

A flag that indicates whether communication ends after the number of bytes set in the telegraph length field have been transferred. When a communication error occurs, the INTIE2 and INTSTA interrupt request signals are generated.

<Set/clear conditions>

Set: This flag is set (1) when the count value of the SCR register is 00H.

Clear: This flag is cleared (0) if the start interrupt, status transmission interrupt, frame end interrupt (if the communication end interrupt does not occur), or INTIE1 interrupt request signal is generated.

(e) Frame end flag (ENDFRAM)...Bit 2

A flag that indicates whether communication ends after the maximum number of bytes (mode 1: 32 bytes, mode 2: 128 bytes) have been transferred.

<Set/clear conditions>

Set: This flag is set (1) when the count value of the CCR register is 00H.

Clear: This flag is cleared (0) if the start interrupt, status transmission interrupt, communication end interrupt (if the frame end interrupt does not occur), or INTIE1 interrupt request signal is generated

Cautions 1. If both the CCR and SCR registers are cleared to 00H, the ENDTRNS and ENDFRAM bits are set (1) at the same time.

2. If the last data field is the NACK signal when the maximum number of transmitted bytes is reached as a result of retransmitting the data, the ENDFRAM bit and IEERR (NACK reception error) bit are set at the same time.

(6) IEBus error status register (ESR)

The ESR register indicates the source of the communication error interrupt request signal of IEBus. Each bit of this register is set (1) as soon as the communication error flag (ISR.IEERR bit) is set (1). The source of a communication error, if any, can be identified by checking the contents of this register.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF365H

	<7>	<6>	<5>	<4>	<3>	<2>	1	<0>
ESR	TERR	PERR	NERR	UERR	OERR	WERR	0	DEFLAG

TERR	Timing error occurrence flag
0	Timing error did not occur
1	Timing error occurred

PERR	Parity error occurrence flag
0	Parity error did not occur
1	Parity error occurred

NERR	NACK reception error occurrence flag
0	NACK reception error does not occur
1	NACK reception error occurred

UERR	Underrun error occurrence flag
0	Underrun error did not occur
1	Underrun error occurred

OERR	Overrun error occurrence flag
0	Overrun error did not occur
1	Overrun error occurred

WERR	Write error occurrence flag
0	Write error did not occur
1	Write error occurred

DEFLAG	Third party error occurrence flag
0	Error occurred during communication with unit
1	Error occurred during communication with station other than unit

Cautions 1. Each bit can only be cleared (0). It cannot be set (1) even if 1 is written to it.

2. The value of the ESR register is updated when an error occurs. If the ESR register is read at this time, however, an undefined value is read. It is recommended to read the ESR register in error interrupt servicing.

3. If a communication error occurs, the IEBus controller returns to the default status and makes preparation for communication. If communication is started without the error corrected, the error flag accumulates the error. Correct the error before the next communication is started.

(a) Timing error occurrence flag (TERR)…Bit 7

<Set/clear conditions>

Set: This flag is set (1) if a timing error occurs.

Clear: By software

A timing error occurs if the high-/low-level width of the communication bit is not the defined value. The defined value of the high- and low-level width is set to the bit processing block and monitored by the internal timer. If a timing error occurs, the INTERR and INTIE2 interrupt request signals are generated.

(b) Parity error occurrence flag (PERR)…Bit 6

<Set/clear conditions>

Set: This flag is set (1) if a parity error occurs.

Clear: By software

A parity error occurs if the parity generated in each field does not match the received parity while the controller is serving as a receiver unit. If the parity does not match in the data field during individual communication, however, the NACK signal is returned and retransmission of data is requested. Therefore, the parity error does not occur.

Table 16-14. Operation if Parity Does Not Match

Field	Communication Mode	Operation if Parity Does Not Match
Master address field	Individual/broadcast	Parity error occurs.
Slave address field	Individual/broadcast	Parity error occurs.
Control data field	Individual/broadcast	Parity error occurs.
Telegraph length field	Individual/broadcast	Parity error occurs.
Data field	Individual	Retransmission is requested by returning NACK signal.
	Broadcast	Parity error occurs.

(c) NACK reception error occurrence flag (NERR)…Bit 5

<Set/clear conditions>

Set: This flag is set (1) if a NACK reception error occurs.

Clear: By software

A NACK reception error occurs if the NACK signal is received during the acknowledge bit period of the slave address field, control data field, or telegraph length field during individual communication, regardless of whether the controller is operating as the master or a slave. If the NACK signal is received during the acknowledge bit period of the data field, a NACK reception error does not occur because data is retransmitted. If the NACK signal is received during the acknowledge period of the last data field when the maximum number of transfer bytes is reached, the NACK reception error occurs.

The NACK reception error does not occur during broadcast communication because the $\overline{\text{ACK}}$/NACK signal is not identified.

The NACK reception error does not occur during third-party communication because only the timing/parity error is detected as an error.

(d) Underrun error occurrence flag (UERR) … Bit 4

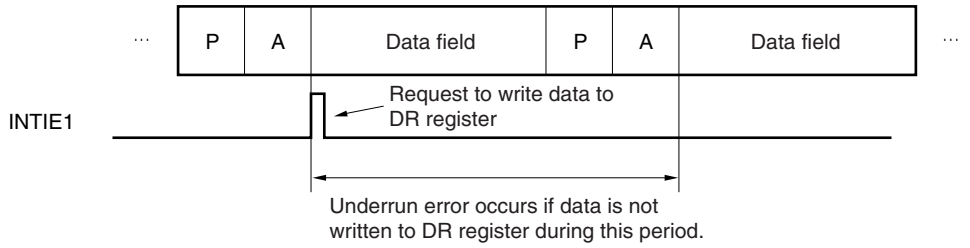
<Set/clear conditions>

Set: This flag is set (1) if an underrun error occurs.

Clear: By software

An underrun error occurs if the next data is not transmitted to the DR register in time before the $\overline{ACK}$ signal is received. If the NACK signal is received during individual communication and during the acknowledge bit period, the underrun error does not occur because the data is retransmitted.

Figure 16-13. Timing of Underrun Error Occurrence



Remark P: Parity bit

A: Acknowledge bit

(e) Overrun error occurrence flag (OERR) … Bit 3

<Set/clear conditions>

Set: This flag is set (1) if an overrun error occurs.

Clear: By software

<R> If 1-byte data is stored in the DR register while the IEBus controller serves as a receiver unit, the data request interrupt request signal (INTIE1) is generated, and the DR register is read by software. If this reading is delayed and the next data is received, an overrun error occurs.

Cautions 1. If the DR register is not read and the number of retransmitted data reaches the maximum number of transmitted bytes (32 bytes) after the overrun error has occurred, the frame end interrupt request signal (INTSTA or INTIE2) occurs. The overrun status is maintained until the DR register is read, even after the frame has ended.

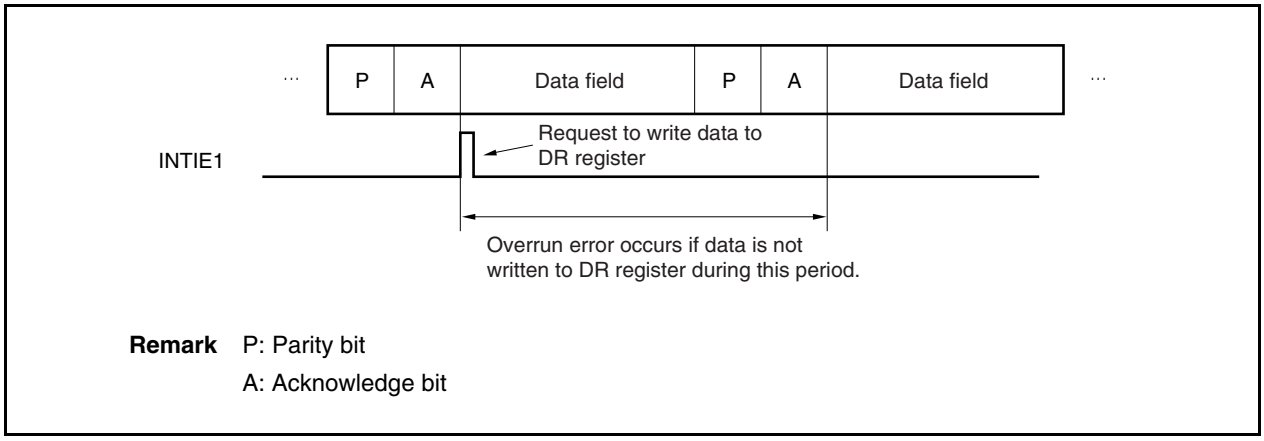
2. The overrun status is cleared only when the DR register is read and when the system is reset. Therefore, be sure to read the DR register in the communication error interrupt processing program.

3. The next data cannot be transmitted in the overrun status if it is 2 bytes or more. Because the data request interrupt request signal (INTIE1) does not occur, the transmit data cannot be set and an underrun error occurs. Therefore, be sure to execute transmission after clearing the overrun status.

Remark During individual communication reception, the NACK signal is returned during the acknowledge bit period of the next data. In response, the transmitter unit retransmits data. Therefore, the CCR register is decremented but the SCR register is not decremented.

During broadcast communication reception, the communication error interrupt request signal (INTIE2) is generated and reception is stopped. At this time, the DR register is not updated. The INTIE1 signal is not generated. The STATRX bit of the SSR register is held set (to 1). The overrun status is cleared when data is received after the DR register has been read.

Figure 16-14. Timing of Overrun Error Occurrence



Remark P: Parity bit

A: Acknowledge bit

(f) Write error occurrence flag (WERR) … Bit 2

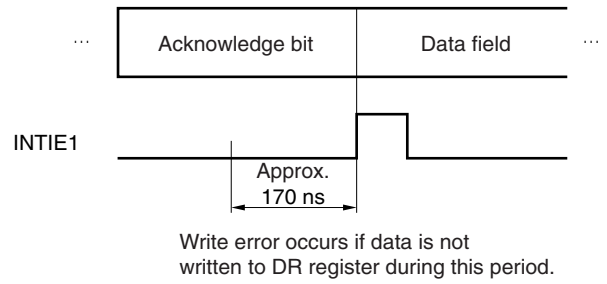
<Set/clear conditions>

Set: This flag is set (1) if a write error occurs.

Clear: By software

A write error occurs if the data written to the DR register is not transmitted in the data field during unit transmission. The timing of occurrence of a write error is illustrated below.

Figure 16-15. Timing of Write Error Occurrence



Cautions 1. Even when the WERR bit is set (1), the INTIE1 interrupt request signal may be generated.

2. If the NACK signal is returned, the WERR bit is not set because data is retransmitted.

(g) Third-party error occurrence flag (DEFLAG)…Bit 0

<Set/clear conditions>

Set: This flag is set (1) if a timing error or parity error occurs during communication regardless of the unit (during communication between third parties).

Clear: By software

Caution If an error occurs before the third-party communication starts even when the slave address field does not match that of the unit (for example, if the NACK signal is received when the received address does not match that of the unit in the slave address field (if the NERR bit is set (1))), the DEFLAG bit is not set (1).

Remark Communication between third parties may take place in the following two cases.

<1> If the received address in the slave address field does not match that of the unit (during individual communication: Matching with UAR register, during broadcast communication: Matching with group or FFFH) and if communication continues after the $\overline{\text{ACK}}$ signal has been received, the unit monitors that communication.

<2> If the unit cannot respond to the received control data in the control field during broadcast communication and if communication continues, the unit monitors that communication. For example, this happens when the unit receives control data FH from master during broadcast communication but the slave reception enable flag of the unit is disabled (BCR.ENSLVRX bit = 0) (the NACK signal is returned and communication ends during individual communication).

(7) IEBus unit address register (UAR)

The UAR register sets the unit address of an IEBus unit. This register must always be set before starting communication.

Sets the unit address (12 bits) to bits 11 to 0.

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

Reset sets this register to 0000H.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset	R/W
UAR	0	0	0	0													FFFFF366H	0000H	R/W

Caution Do not set the UAR register while communication is enabled (BCR.ENIEBUS bit = 1).

(8) IEBus slave address register (SAR)

During a master request, the value of this register is reflected in the value of the transmit data in the slave address field. The SAR register must always be set before starting communication.

The SAR register sets the slave address (12 bits) to bits 11 to 0.

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

Reset sets this register to 0000H.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset	R/W
SAR	0	0	0	0													FFFFF368H	0000H	R/W

<R> **Caution Be sure to set the SAR register in the following timing.**

- **When BCR.ENIEBUS bit = 0**
- **Between ENIEBUS bit = 1 and the first master request (set BCR.MSTRQ bit to (1))**
- **When ENIEBUS bit = 1, and MSTRQ bit = 0, between either the end of that communication, frame, or error and the next master request (set MSTRQ bit to (1))**

(9) IEBus partner address register (PAR)

The PAR register stores the master address value received in the master address field regardless of whether the unit is operating as the master or a slave.

If a request “4H” to read the lock address (lower 8 bits) is received from the master, read the value of this register by software, and write the data of the lower 8 bits to the DR register.

If a request “5H” to read the lock address (higher 4 bits) is received from the master, read the value of this register by software and write the data of bits 11 to 8 to the higher 4 bits of the DR register.

The PAR register sets the partner address (12 bits) to bits 11 to 0.

This register is read-only, in 16-bit units.

Reset sets this register to 0000H.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset	R/W
PAR	0	0	0	0													FFFFF36AH	0000H	R

Caution The PAR register stores an address value if the parity is correct and the unit is not locked when the parity period of the master address field expires. If the PAR register is read at this time, an undefined value is read.

(10) IEBus receive slave address register (RSA)

The RSA register stores the slave address value received in the slave address field regardless of whether the unit is operating as the master or a slave.

This register is read-only, in 16-bit units.

Reset sets this register to 0000H.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset	R/W
RSA	0	0	0	0													FFFFF36CH	0000H	R

Caution The RSA register stores an address value if the parity is correct and the unit is not locked when the parity period of the slave address field expires. If the RSA register is read at this time, an undefined value is read.

(11) IEBus control data register (CDR)

The CDR register can be read or written in 8-bit units.

Reset sets this register to 00H.

Remark The CDR register consists of a write register and a read register and data written to the CDR register cannot be read as is. The data read from this register is the data received by IEBus communication.

(a) When master unit

The data of the lower 4 bits is reflected in the data transmitted in the control field. During a master request, the CDR register must be set in advance before starting communication.

(b) When slave unit

The data received in the control field is written to the lower 4 bits.

When the status transmission flag (ISR.STATUSF bit) is set (1), an interrupt request signal (INTIE2) is issued, and each processing should be performed by software, according to the value of the lower 4 bits of the CDR register.

After reset: 00H R/W Address: FFFFF36EH

	7	6	5	4	3	2	1	0
CDR	0	0	0	0	MOD	SELCL2	SELCL1	SELCL0

MOD	SELCL2	SELCL1	SELCL0	Function
0	0	0	0	Read slave status
0	0	0	1	Undefined
0	0	1	0	Undefined
0	0	1	1	Read data and lock
0	1	0	0	Read lock address (lower 8 bits)
0	1	0	1	Read lock address (lower 4 bits)
0	1	1	0	Read slave status and unlock
0	1	1	1	Read data
1	0	0	0	Undefined
1	0	0	1	Undefined
1	0	1	0	Write command and lock
1	0	1	1	Write data and lock
1	1	0	0	Undefined
1	1	0	1	Undefined
1	1	1	0	Write command
1	1	1	1	Write data

Cautions 1. Because the slave unit must judge whether the received data is a "command" or "data", read the value of the CDR register after completing communication.

2. If the master unit sets an undefined value, the slave unit returns the NACK signal and communication is aborted. During broadcast communication, the master unit ignores the acknowledge bit and continues communication. Therefore, do not set an undefined value.

(c) Slave status return operation

When IEBus receives a request to transfer from master to slave status or a lock address request (control data: 0H, 6H), whether the $\overline{\text{ACK}}$/NACK signal in the control field is returned or not depends on the status of the IEBus unit.

(1) If 0H or 6H control data was received in the unlocked state → $\overline{\text{ACK}}$ signal returned
(2) If 4H or 5H control data was received in the unlocked state → NACK signal returned
(3) If 0H, 4H, 5H or 6H control data was received in the locked state from the unit that sent the lock request → $\overline{\text{ACK}}$ signal returned
(4) If 0H, 4H, or 5H control data was received in the locked state from other than the unit that sent the lock request → $\overline{\text{ACK}}$ signal returned
(5) If 6H control data was received in the locked state from other than the unit that sent the lock request → NACK signal returned

In all of the above cases, the acknowledgment of a slave status or lock request will cause the STATUSF bit to be set (1) and the status interrupt signal (INTIE2, INTSTA) to be generated. The generation timing is at the end of the control field parity bit (at the start of the acknowledge bit). However, if NACK is returned, a NACK receive error is generated after the acknowledge bit, and communication is terminated.

Figure 16-16. Interrupt Request Signal Generation Timing (for (1), (3), and (4))

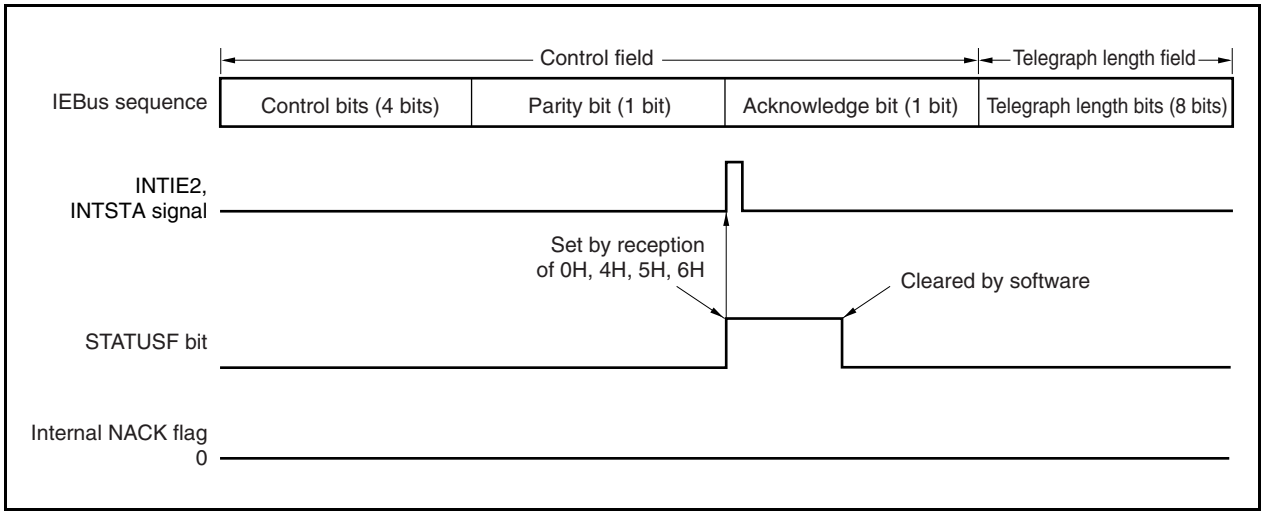
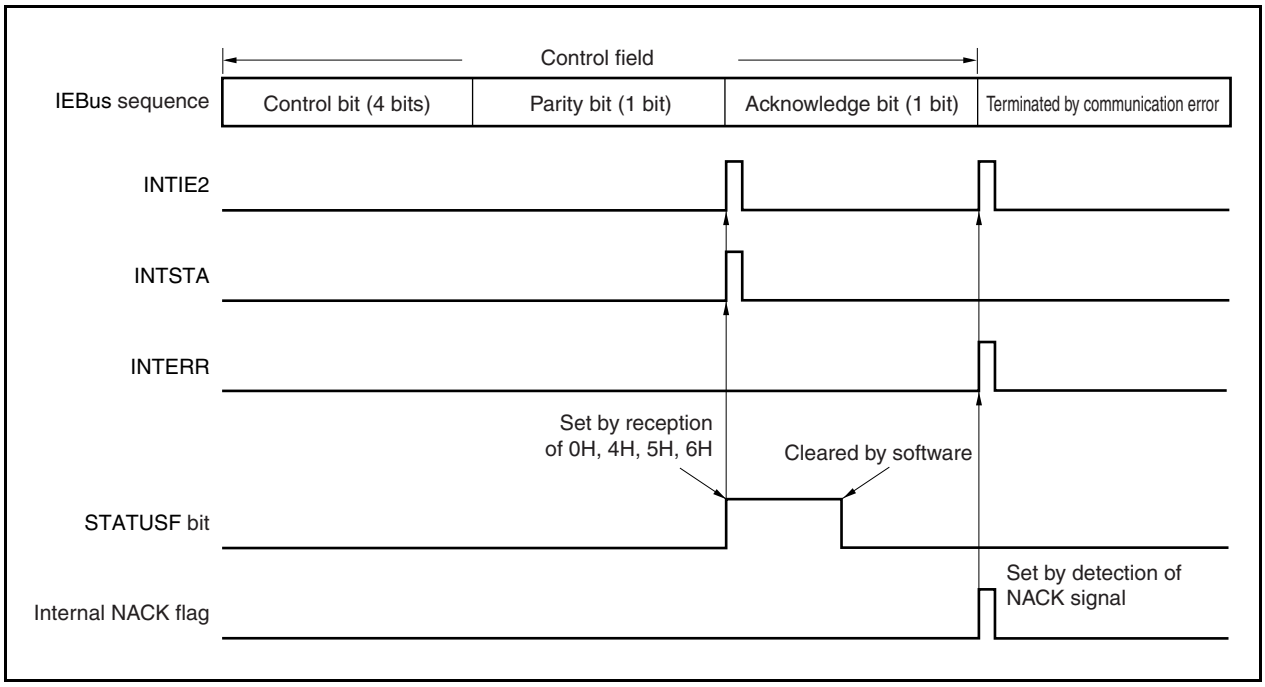
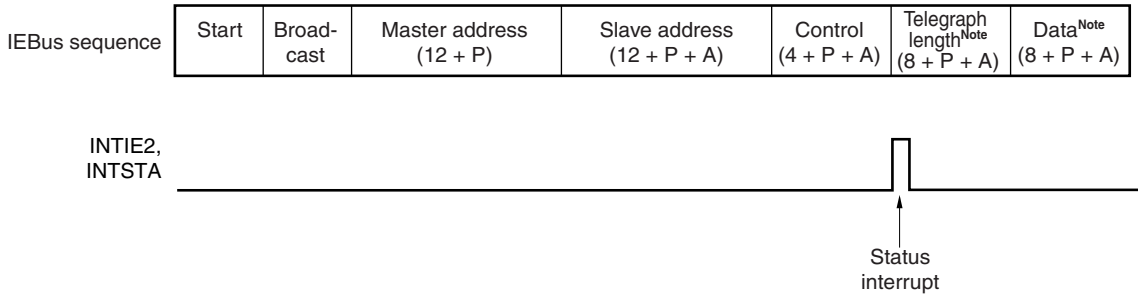


Figure 16-17. Interrupt Request Signal Generation Timing (for (2) and (5))



Because in (4) and (5) the communication was from other than the unit that sent the lock request while IEBus was in the locked state, the start or communication end interrupt request signals (INTIE2, INTSTA) are not generated, even if the IEBus unit is the communication target. The STATUSF bit is set (1) and the status interrupt request signals (INTIE2, INTSTA) are generated, however, if a slave status or lock address request is acknowledged. Note that even if the same control data is received while IEBus is in the locked state, the interrupt generation timing for INTIE2 and INTSTA differs depending on whether the master unit (3) or another unit (4) is requesting the locked state.

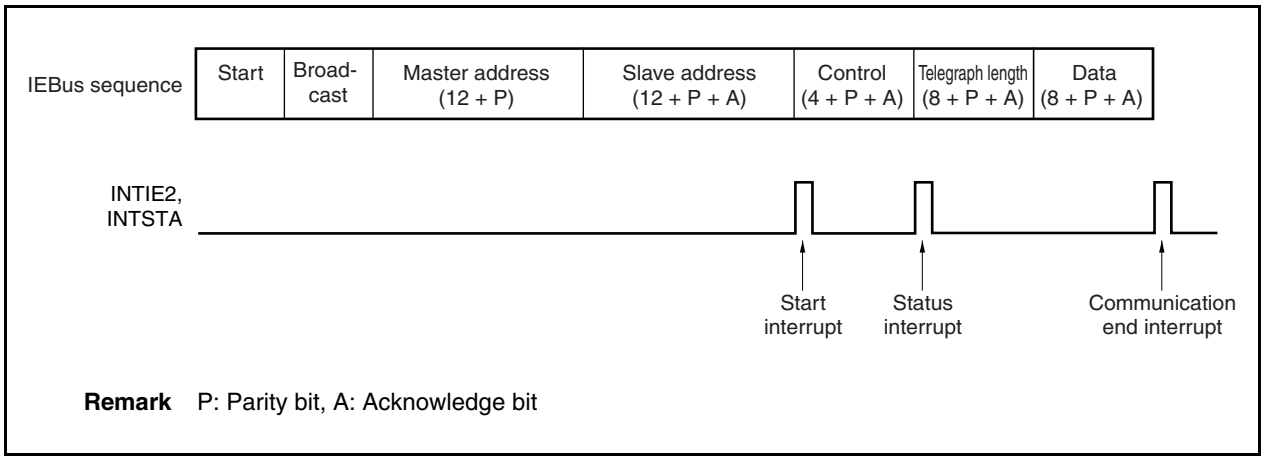
Figure 16-18. Timing of INTIE2 and INTSTA Interrupt Request Signal Generation in Locked State (for (4) and (5))



Note The telegraph length and data modes are not set in the case of (5) because the NACK signal is returned.

Remark P: Parity bit, A: Acknowledge bit

Figure 16-19. Timing of INTIE2 and INTSTA Interrupt Request Signal Generation in Locked State (for (3))



Remark P: Parity bit, A: Acknowledge bit

(12) IEBus telegraph length register (DLR)

The DLR register can be read or written in 8-bit units.

Reset sets this register to 01H.

(a) When transmission unit ... Master transmission, slave transmission

The data of this register is reflected in the data transmitted in the telegraph length field and indicates the number of bytes of the transmit data. The DLR register must be set in advance before transmission.

(b) When reception unit ... Master reception, slave reception

The receive data in the telegraph length field transmitted from the transmission unit is written to this register.

Remark The DLR register consists of a write register and a read register. Consequently, data written to this register cannot be read as is. The data that can be read is the data received during IEBus communication.

After reset: 01H R/W Address: FFFFF36FH

	7	6	5	4	3	2	1	0
DLR								

Bit								Setting value	Remaining number of communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
:	:	:	:	:	:	:	:	:	:
0	0	1	0	0	0	0	0	20H	32 bytes
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	256 bytes

Cautions 1. When the master issues a request (0H, 4H, 5H, or 6H) for transmission of a slave status or a lock address (higher 4 bits and lower 8 bits), 01H is transmitted as the telegraph length regardless of the contents of the DLR register. It is therefore not necessary to set the DLR register by software.

2. When the IEBus controller serves as a receiver unit, the DLR register stores a telegraph length if the value of the parity bit of the telegraph length field is correct. If the DLR register is read at this time, an undefined value is read.

(13) IEBus data register (DR)

The DR register sets the communication data (8 bits) to bits 7 to 0.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

Remark The DR register consists of a write register and a read register. Consequently, data written to this register cannot be read as is. The data that can be read is the data received during IEBus communication.

(a) When transmission unit

The data (1 byte) written to the DR register is stored in the transmit shift register of the IEBus interface block. It is then output from the most significant bit, and an interrupt request signal (INTIE1) is generated each time 1 byte has been transmitted. If the NACK signal is received after 1-byte data has been transferred during individual transfer, data is not transferred from the DR register to the transmit shift register, and the same data is retransmitted. At this time, INTIE1 signal is not generated.

INTIE1 signal is generated when the transmit shift register stores the DR register value. However, when the last byte and 32nd byte (the last byte of 1 communication frame) is stored in the transmit shift register, the INTIE1 signal is not generated.

<R> **(b) When reception unit**

One byte of the data received by the receive shift register of the IEBus interface block is stored in this register.

Each time 1 byte has been correctly received, an interrupt request signal (INTIE1) is generated.

After reset: 00H R/W Address: FFFFF370H

	7	6	5	4	3	2	1	0
DR								

Cautions 1. If the next data is not in time while the transmission unit is set, an underrun occurs, and a communication error interrupt request signal (INTIE2, INTERR) occurs, stopping transmission.

2. If data is not read in time before the next data is read when the IEBus controller functions as a receiver unit during individual communication reception, the NACK signal is returned by the acknowledge bit of the data field, requesting the master to retransmit the data. If the DR register is not read after the data has reached the maximum number of transmit bytes, however, the frame end interrupt request signal (INTIE2, INTSTA) and NACK reception error interrupt request signal (INTIE2, INTERR) are generated at the same time.

3. If data is not read in time before the next data is received when the IEBus controller functions as a receiver unit during broadcast communication reception, an overrun error occurs and the communication error interrupt request signal (INTIE2, INTERR) is generated.

4. When the IEBus controller serves as a receiver unit, the DR register stores receive data if the value of the parity bit of the data field is correct. If the DR register is read at this time, an undefined value is read.

(14) IEBus field status register (FSR)

The FSR register stores the status of the field status of the IEBus controller if an interrupt request signal (INTIE1, INTIE2, INTSTA, or INTERR) is generated.

This register is read-only, in 8-bit units.

Reset sets this register to 00H.

Cautions 1. If an interrupt request signal is generated during communication between third parties, the FSR register is cleared to 00H. However, because only an interrupt request signal that is generated if an error occurs is generated during communication between third parties, the error can be identified as that during communication between third parties, by reading third-party error flag (ESR.DEFLAG bit).

2. The FSR register updates the status information when an interrupt request signal is generated. If the FSR register is read at this time, however, an undefined value is read.

3. If another interrupt request signal is generated before the FSR register is read, the status information when the preceding interrupt occurred is updated by the status information when the new interrupt occurs.

4. Use the FSR register only for problem analysis; do not use it with the actual software.

After reset: 00H R Address: FFFFF371H

	7	6	5	4	3	2	1	0
FSR	0	0	0	0	0	0	FSTATE1	FSTATE0

Remark For the explanation of the FSTATE1 and FSTATE0 bits, see **Table 16-15 Field Status**.

Table 16-15. Field Status

Field Status	Explanation		
	Master/Slave	Field	Transmission/Reception
Slave transmission status FSR = 00H	Slave operation	Start field	Reception
		Master address field	
		Slave address field	
		Control data field	
		Telegraph length field	
		Data field	
Slave transmission status FSR = 01H	Slave operation	Telegraph length field	Transmission
		Data field	
Master reception status FSR = 02H	Master operation	Telegraph length field	Reception
		Data field	
Master transmission status FSR = 03H	Master operation	Start field	Transmission
		Master address field	
		Slave address field	
		Control data field	
		Telegraph length field	
		Data field	

(15) IEBus success count register (SCR)

The SCR register indicates the number of remaining communication bytes.

The count value of the counter in which the value set by the DLR register is decremented by the $\overline{ACK}$ signal in the data field is read from this register. When the count value has reached “00H”, the communication end flag (ISR.ENDTRNS bit) is set (1).

This register is read-only, in 8-bit units.

Reset sets this register to 00H.

After reset: 01H R Address: FFFFF372H

	7	6	5	4	3	2	1	0
SCR								

Bit								Setting value	Remaining number of communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
:	:	:	:	:	:	:	:	:	:
0	0	1	0	0	0	0	0	20H	32 bytes
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	0 bytes (end of communication) or 256 bytes[Note]

Note The actual counter consists of 9 bits. When “00H” is read, it cannot be judged whether the remaining number of communication data bytes is 0 (end of communication) or 256. Therefore, either the communication end flag (ENDTRNS bit) is used, or if “00H” is read when the first interrupt occurs at the beginning of communication, the remaining number of communication data bytes is judged to be 256.

Caution The SCR register is updated when the parity period of the telegraph field expires and when the $\overline{ACK}$ signal of the data field is received. If the SCR register is read at this time, however, an undefined value is read.

(16) IEBus communication count register (CCR)

The CCR register indicates the number of bytes remaining from the communication byte number specified by the communication mode.

This register indicates the number of transfer bytes.

The maximum number of transmitted bytes per frame defined in each mode (mode 1: 32 bytes, mode 2: 128 bytes) is preset to this register. The count value of the counter that is decremented during the acknowledge bit period of the data field regardless of the $\overline{\text{ACK}}$/NACK signal is read from this register. Whereas the SCR register is decremented during normal communication ($\overline{\text{ACK}}$ signal), the CCR register is decremented when 1 byte has been communicated, regardless of whether the signal is $\overline{\text{ACK}}$ or NACK. When the count value has reached “00H”, the frame end flag (ISR.ENDFRAM bit) is set (1).

The preset value of the maximum number of transmitted bytes per frame is 20H (32 bytes) in mode 1 and 80H (128 bytes) in mode 2.

This register is read-only, in 8-bit units.

Reset sets this register to 20H.

After reset: 20H R Address: FFFFF373H

	7	6	5	4	3	2	1	0
CCR								

Caution The maximum number of transmit bytes is preset to the CCR register when the start bit is transmitted or received, and the register is decremented when the parity period of the data field expires. If the CCR register is read at this time, however, an undefined value is read.

(17) IEBus clock select register (OCKS2)

The OCKS2 register selects the clock of IEBus. The main clock frequencies that can be used are shown below. No other main clock frequencies can be used.

This register can be read or written in 8-bit units.

Reset sets this register to 00H.

- 6.0 MHz/6.291456 MHz (6.29 MHz)
- 12.0 MHz/12.582912 MHz (12.58 MHz)
- 18.0 MHz/18.874368 MHz (18.87 MHz)

After reset: 00H R/W Address: FFFFF348H

	7	6	5	4	3	2	1	0
OCKS2	0	0	0	OCKSEN2	OCKSTH2	0	OCKS21	OCKS20

OCKSEN2	IEBus clock operation specification
0	IEBus clock operation stops
1	IEBus clock operation enabled

OCKSTH2	OCKS21	OCKS20	IEBus clock selection
0	0	0	$f_{XX}/2$ (when f_{XX} = 12.0 MHz or f_{XX} = 12.58 MHz)
0	0	1	$f_{XX}/3$ (when f_{XX} = 18.0 MHz or f_{XX} = 18.87 MHz)
1	0	0	f_{XX} (when f_{XX} = 6.0 MHz or f_{XX} = 6.29 MHz)
Other than above			Setting prohibited

16.4 Interrupt Operations of IEBus Controller

16.4.1 Interrupt control block

Interrupt request signal

<1>	Communication error	IEERR
	(i) Timing error:	TERR
	(ii) Parity error:	PERR
	(iii) NACK receive error:	NERR
	(iv) Underrun error:	UERR
	(v) Overrun error:	OERR
	(vi) Write error:	WERR
<2>	Start interrupt	STARTF
<3>	Status communication	STATUSF
<4>	End of communication	ENDTRNS
<5>	End of frame	ENDFRAM
<6>	Transmit data write request	$\overline{\text{STATTX}}$
<7>	Receive data read request	STATRX

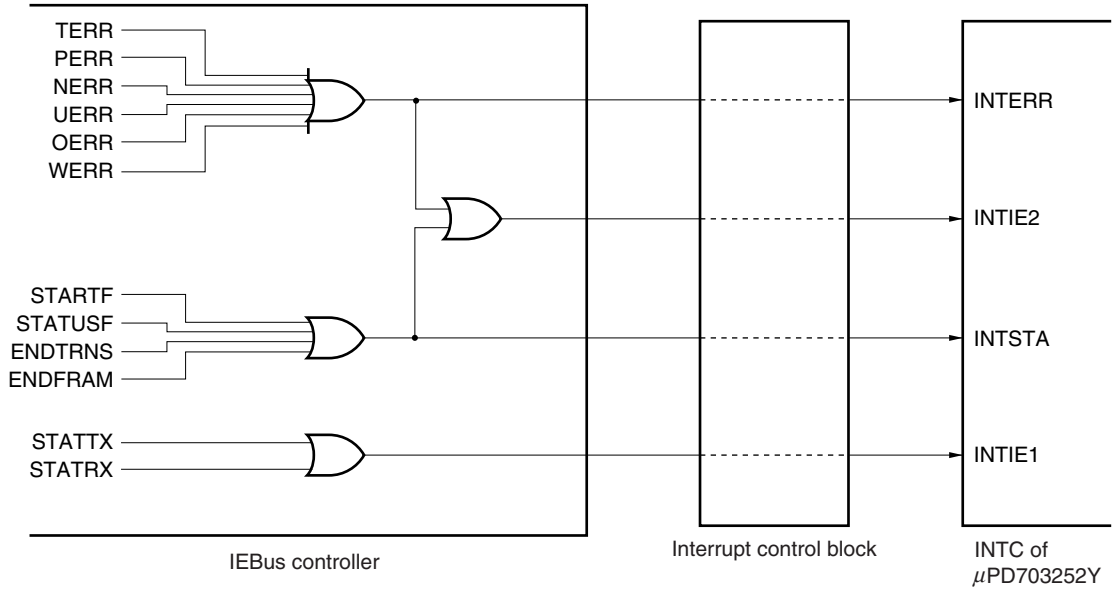
A communication error <1> occurs if any of the above error sources (i) to (vi) is generated.

These error sources are assigned to the error status register (ESR) (see **Table 16-18 Communication Error Source Processing List**).

The above interrupt signals <1> to <5> are assigned to the ISR register (see **Table 16-17 Interrupt Source List**).

The configuration of the interrupt control block is illustrated below.

Figure 16-20. Configuration of Interrupt Control Block



Cautions 1. The logical sum (OR) output of the STATRX and $\overline{\text{STATTX}}$ signals is treated as an interrupt request signal (INTIE1).

2. The logical sum (OR) output of the TERR, PERR, NERR, UERR, OERR, and WERR signals is treated as a communication error (IEERR) or an interrupt request signal (INTERR).

3. The logical sum (OR) output of the STARTF, STATUSF, ENDTRNS, and ENDFRAM signals is treated as an interrupt request signal (INTSTA).

4. The logical sum (OR) output of the IEERR, STARTF, STATUSF, ENDTRNS, and ENDFRAM signals (logical sum (OR) output of INTSTA and INTERR signals) is treated as an interrupt request signal (INTIE2).

Table 16-16. Interrupt Request Signal Generation Source List

Interrupt Source		Symbol	Interrupt Request Signal			
			INTIE1	INTIE2	INTERR	INTSTA
Communication error interrupt		IEERR		√	√	
	Timing error	TERR				
	Parity error	PERR				
	NACK reception error	NERR				
	Underrun error	UERR				
	Overrun error	OERR				
	Write error	WERR				
Start interrupt		STARTF		√		√
Status transmission		STATUSF		√		√
End of communication		ENDTRNS		√		√
End of frame		ENDFRAM		√		√
Transmit data write request		$\overline{\text{STATTX}}$	√			
Receive data write request		STATRX	√			

16.4.2 Example of identifying interrupt

The IEBus controller processes interrupts in the following two ways.

- Using three interrupt request signals: INTIE1, INTERR, and INTSTA
- Using two interrupt request signals: INTIE1 and INTIE2

Caution Mask the interrupt sources that are not used so that the interrupts do not occur.

How an interrupt is identified in each of the above cases is explained below.

(1) When INTIE1, INTERR, and INTSTA signals are used

Figure 16-21. Example of Identifying INTIE1 Signal Interrupt (When INTIE1, INTERR, and INTSTA Signals Are Used)

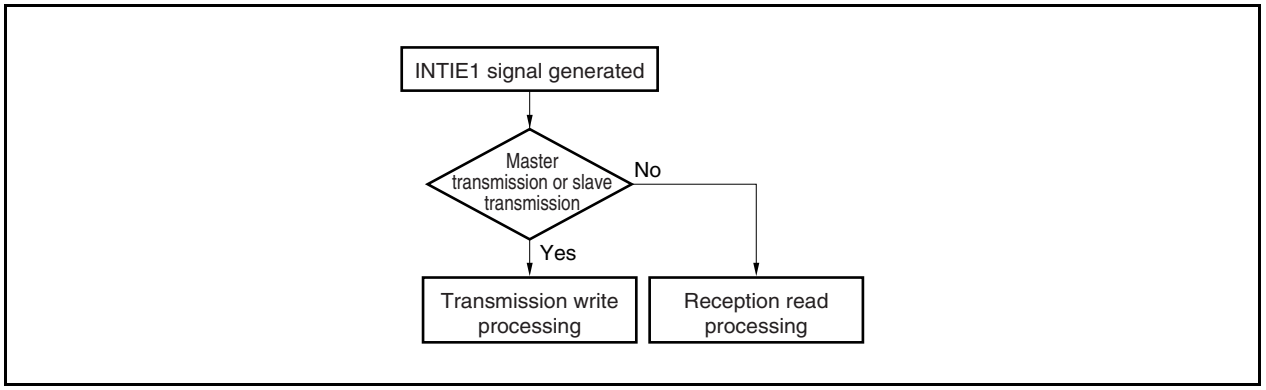


Figure 16-22. Example of Identifying INTERR Signal Interrupt (When INTIE1, INTERR, and INTSTA Signals Are Used)

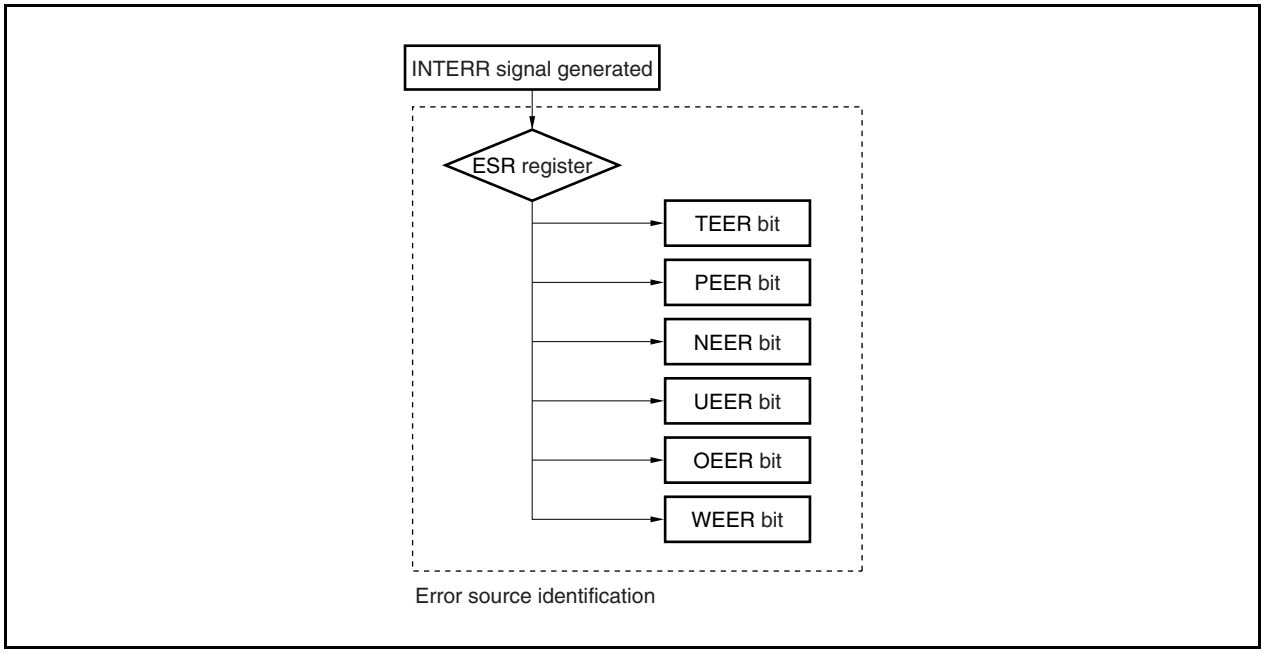
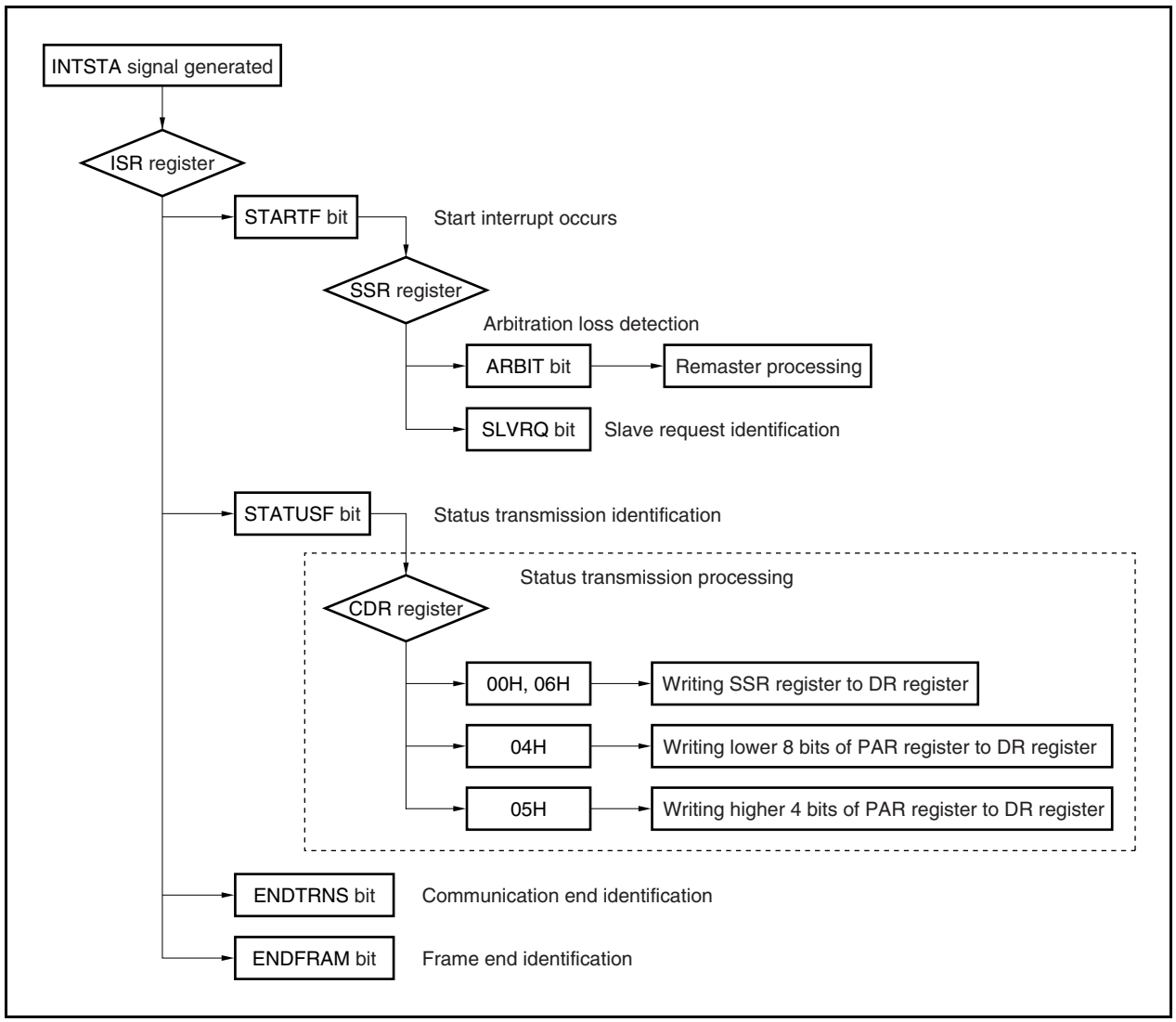


Figure 16-23. Example of Identifying INTSTA Signal Interrupt (When INTIE1, INTERR, and INTSTA Signals Are Used)



(2) When INTIE1 and INTIE2 signals are used

Figure 16-24. Example of Identifying INTIE1 Signal Interrupt (When INTIE1 and INTIE2 Signals Are Used)

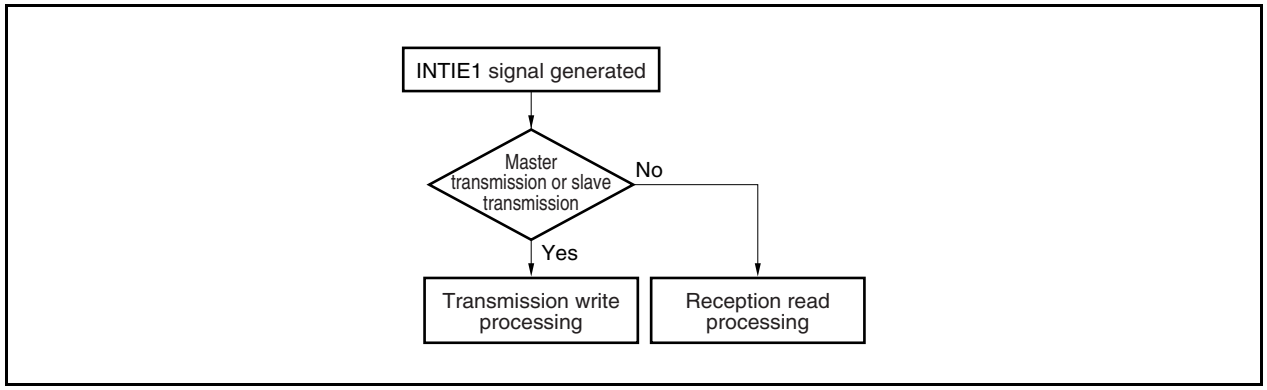
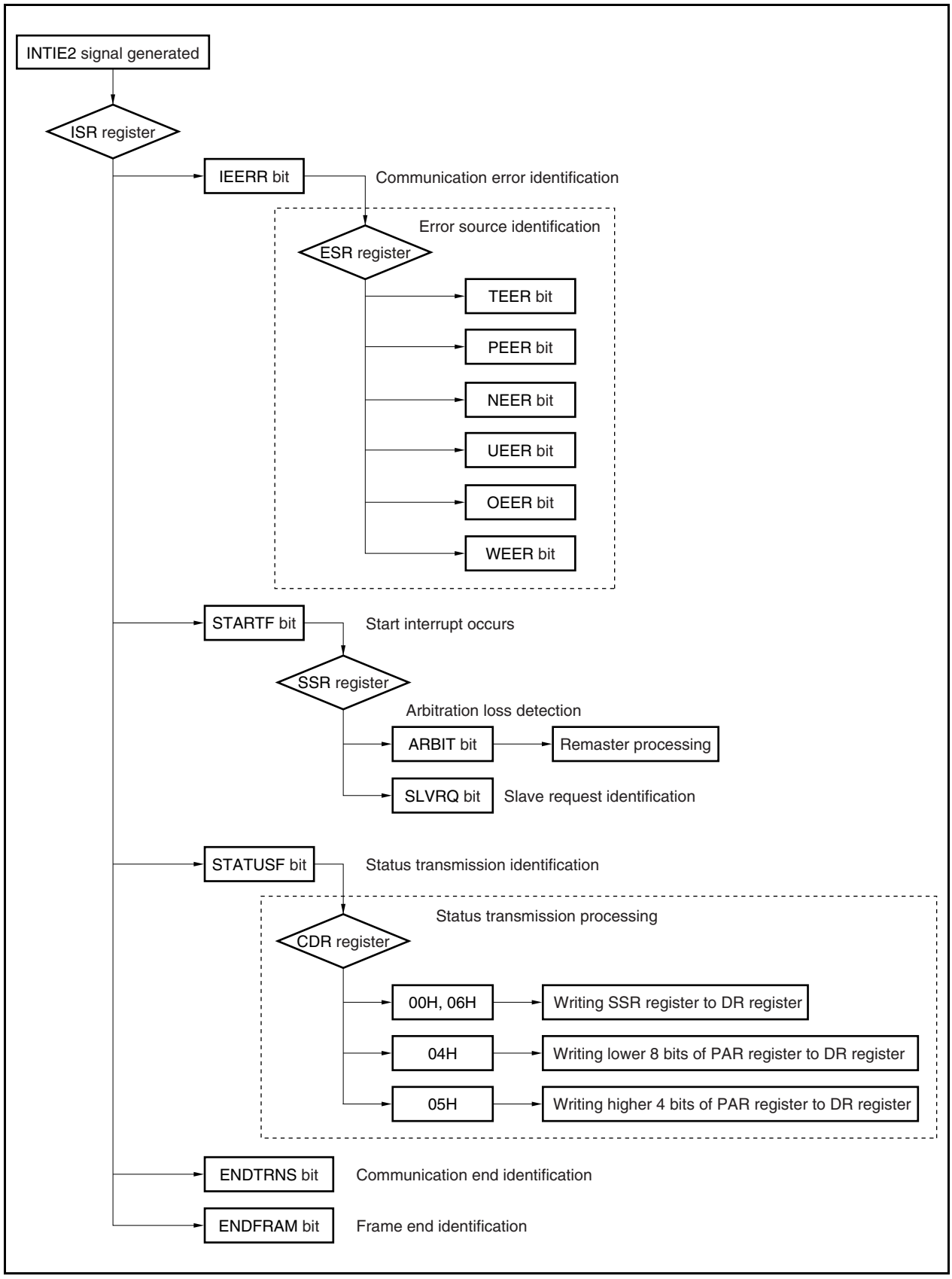


Figure 16-25. Example of Identifying INTIE2 Signal Interrupt (When INTIE1 and INTIE2 Signals Are Used)



16.4.3 Interrupt source list

<R> Interrupt request signals can be specified via software. The interrupt sources are listed below.

Table 16-17. Interrupt Source List

Interrupt Source		Condition of Generation		Software Processing After Generation of Interrupt Request Signal	Remark
		Unit	Field		
Communication error	Timing error	Master/slave	All fields	Undo communication processing	Communication error is logical sum (OR) output of timing error, parity error, NACK reception error, underrun error, overrun error, and write error.
	Parity error	Reception	Other than data (individual)		
			All fields (broadcast)		
	NACK reception	Reception (Transmission)	Other than data (individual)		
	Underrun error	Transmission	Data		
	Overrun error	Reception	Data (broadcast)		
	Write error	Transmission	Data		
Start interrupt		Master	Slave/address	Slave request judgment Arbitration judgment (If lost, remaster processing) Communication preparation processing	Interrupt always occurs if lost in arbitration during master request
		Slave	Slave/address	Slave request judgment Communication preparation processing	Generated only during slave request
Status transmission		Slave	Control	Refer to transmission processing example such as slave status.	Interrupt occurs regardless of slave transmission enable flag Interrupt occurs if NACK is returned in the control field.
<R> End of communication		<R> Transmission	Data	Communication end processing	Set if SCR register is cleared to 00H
<R>		<R> Reception	Data	Communication end processing Receive data processing	
End of frame		Transmission	Data	Retransmission preparation processing	Set if CCR register is cleared to 00H
		Reception	Data	Re-reception preparation processing	
Transmit data write		Transmission	Data	Reading of transmit data[Note]	Set after transfer transmission data to internal shift register This does not occur when the last data is transferred.
Receive data read		Reception	Data	Reading of received data[Note]	Set after normal data reception

<R> **Note** If software manipulation is not executed.

16.4.4 Communication error source processing list

The following table shows the occurrence conditions of the communication errors (timing error, NACK reception error, overrun error, underrun error, parity error, and write error), error processing by the IEBus controller, and examples of processing by software.

Table 16-18. Communication Error Source Processing List (1/2)

		Timing Error			
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	If bit specification timing is not correct			
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	• Reception stops. • INTIE2 signal occurs • To start bit waiting status **Remark** Communication between other units does not end.		• Transmission stops. • INTIE2 signal occurs • To start bit waiting status	
	Software processing	• Error processing (such as retransmission request)		• Error processing (such as retransmission request)	
Individual communication	Hardware processing	• Reception stops. • INTIE2 signal occurs • NACK signal is returned. • To start bit waiting status		• Transmission stops. • INTIE2 signal occurs • To start bit waiting status	
	Software processing	• Error processing (such as retransmission request)		• Error processing (such as retransmission request)	

		NACK Reception Error				
Occurrence condition	Unit status	Reception		Transmission		
	Occurrence condition	Unit NACK signal transmission		Unit NACK signal transmission		
	Location of occurrence	Other than data field	Data field	Other than data field	Data field	NACK signal reception of data of 32nd byte
Broadcast communication	Hardware processing	–	–	–	–	–
	Software processing	–	–	–	–	–
Individual communication	Hardware processing	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status	• INTIE2 signal does not occur. • Data retransmitted by other unit is received.	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status	• INTIE2 signal does not occur. • Retrans-mission processing	• INTIE2 signal occurs. • To start bit waiting status
	Software processing	• Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)

Table 16-18. Communication Error Source Processing List (2/2)

		Overrun Error		Underrun Error/Write Error	
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	DR register cannot be read in time before the next data is received.		DR register cannot be written in time before the next data is transmitted.	
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	–	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status **Remarks 1.** Communication between other units does not end. **2.** Data cannot be received until the overrun status is cleared.	–	• Transmission stops. • INTIE2 signal occurs. • To start bit waiting status
	Software processing	–	• DR register is read and overrun status is cleared. • Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)
Individual communication	Hardware processing	–	• INTIE2 signal does not occur. • NACK signal is returned. • Data is retransmitted from other unit. **Remark** Data cannot be received until overrun status is cleared.	–	• Transmission stops. • INTIE2 signal occurs. • To start bit waiting status
	Software processing	–	• DR register is read and overrun status is cleared. • Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)

		Parity Error			
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	Received data and received parity do not match.		–	
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status **Remark** Communication between other units does not end.		–	–
	Software processing	• Error processing (such as retransmission request)		–	–
Individual communication	Hardware processing	• Reception stops. • INTIE2 signal occurs. • To start bit waiting status	• Reception does not stop. • INTIE2 signal does not occur. • NACK signal is returned. • Data retransmitted by other unit is received.	–	–
	Software processing	• Error processing (such as retransmission request)	–	–	–

16.5 Interrupt Request Signal Generation Timing and Main CPU Processing

<R> 16.5.1 Master transmission

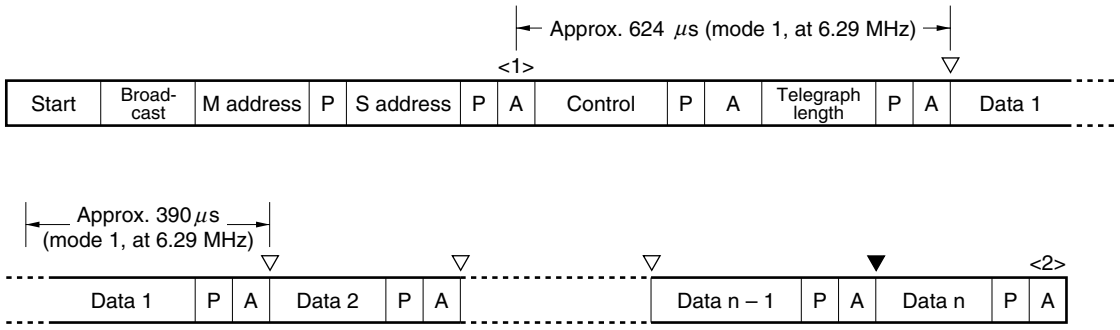
Initial preparation processing:

Sets a unit address, slave address, control data, telegraph length, and the first byte of the transmit data.

Communication start processing:

Set the BCR register (enable communication, master request, and slave reception).

Figure 16-26. Master Transmission



<1> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing
↓
Judgment of slave request → Slave reception processing
↓
Judgment of arbitration result → Remaster request processing

<2> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing
↓
Judgment of end of communication → End of communication processing
↓
Judgment of end of frame → Recommunication processing
(See **16.5.1 (2) Recommunication processing**)

Note This processing is necessary only when the INTIE2 interrupt request signal is used as the start interrupt, and is not necessary when the INTSTA interrupt request signal is used (in this case, the error processing is performed by using the INTERR interrupt request signal).

<R> **Remarks 1.** ▽: Interrupt request signal (INTIE1) occurrence (See **16.5.1 (1) Interrupt request signal (INTIE1) occurrence**)

2. ▼: An interrupt request signal (INTIE1) does not occur.

3. n = Final number of data bytes

(1) Interrupt request signal (INTIE1) occurrence

If the NACK signal is received from the slave in the data field, an interrupt request signal (INTIE1) is not issued to the interrupt controller (INTC), and the same data is retransmitted by hardware.

If the transmit data is not written in time during the period of writing the next data, a communication error interrupt request signal (INTERR) occurs due to occurrence of underrun, and communication ends midway.

(2) Recommunication processing

In the vector interrupt servicing in <2> in Figure 16-26, it is judged whether the data has been correctly transmitted within one frame. If the data has not been correctly transmitted (if the number of data to be transmitted in one frame could not be transmitted), the data must be retransmitted in the next frame, or the remainder of the data must be transmitted.

16.5.2 Master reception

Before performing master reception, it is necessary to notify the unit that will be the slave of slave transmission. Therefore, more than two communication frames are necessary for master reception.

The slave unit prepares the transmit data, sets (1) the slave transmission enable flag (BCR.ENSLVTX bit), and waits.

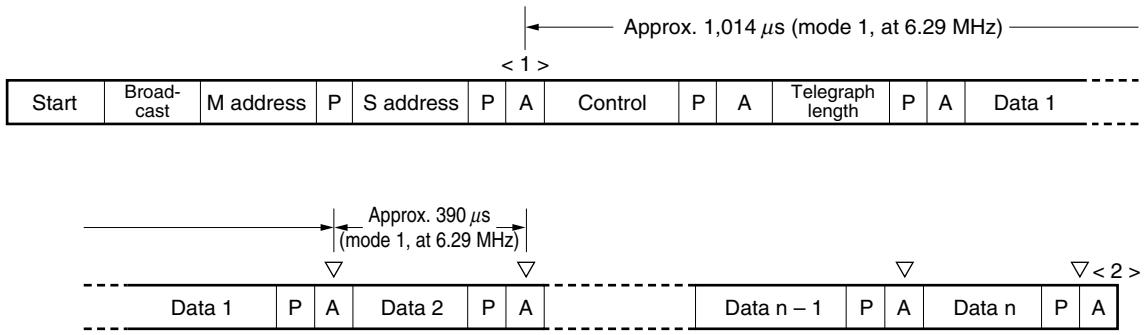
Initial preparation processing:

Set a unit address, slave address, and control data.

Communication start processing:

Set the BCR register (enable communication and master request).

Figure 16-27. Master Reception



<1> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing
↓
Judgment of slave request → Slave processing
↓
Judgment of arbitration result → Remaster request processing

<2> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing
↓
Judgment of end of communication → End of communication processing
↓
Judgment of end of frame → Frame end processing (See **16.5.2 (2) Frame end processing**)

Note This processing is necessary only when the INTIE2 interrupt request signal is used as the start interrupt, and is not necessary when the INTSTA interrupt request signal is used (in this case, the error processing is performed by using the INTERR interrupt request signal).

<R> **Remarks 1.** ▽: Interrupt request signal (INTIE1) occurrence (See **16.5.2 (1) Interrupt request signal (INTIE1) occurrence**)

2. n = Final number of data bytes

(1) Interrupt request signal (INTIE1) occurrence

If the NACK signal is transmitted (hardware processing) in the data field, an interrupt request signal (INTIE1) is not issued to the INTC, and the same data is retransmitted from the slave.

If the receive data is not read by the time the next data is received, the hardware automatically transmits the NACK signal.

(2) Frame end processing

In the vector interrupt servicing in <2> in Figure 16-27, it is judged whether the data has been correctly received within one frame. If the data has not been correctly received (if the number of data to be received in one frame could not be received), a request to retransmit the data must be made to the slave in the next communication frame.

16.5.3 Slave transmission

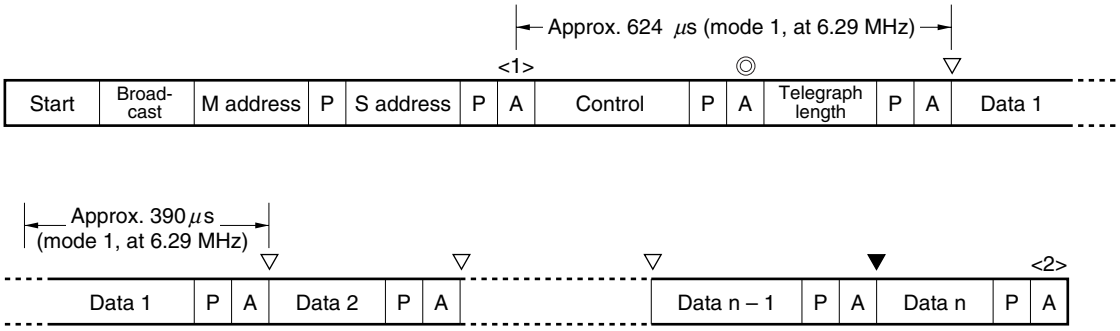
Initial preparation processing:

Set a unit address, telegraph length, and the first byte of the transmit data.

Communication start processing:

Set the BCR register (enable communication, slave transmission, and slave reception).

Figure 16-28. Slave Transmission



<1> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of slave request

<2> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Frame end processing (See **16.5.3 (2) Frame end processing**)

Note This processing is necessary only when the INTIE2 interrupt request signal is used as the start interrupt, and is not necessary when the INTSTA interrupt request signal is used (in this case, the error processing is performed by using the INTERR interrupt request signal).

<R> **Remarks 1.** ▽: Interrupt request signal (INTIE1) occurrence (See **16.5.3 (1) Interrupt request signal (INTIE1) occurrence**).

2. ▼: An interrupt request signal (INTIE1) does not occur.

3. ◎: Interrupt request signal (INTIE2) occurrence

An interrupt request signal occurs only when 0H, 4H, 5H, or 6H is received in the control field in the slave status (for the slave status response operation during the locked status, see **16.3 (11) IEBus control data register (CDR)**).

4. n = Final number of data bytes

(1) Interrupt request signal (INTIE1) occurrence

If the NACK signal is received from the master in the data field, an interrupt request signal (INTIE1) is not issued to the INTC, and the same data is retransmitted by hardware.

If the transmit data is not written in time during the period of writing the next data, a communication error interrupt request signal (INTERR) occurs due to occurrence of underrun, and communication is abnormally ended.

(2) Frame end processing

In the vector interrupt servicing in <2> in Figure 16-28, it is judged whether the data has been correctly transmitted within one frame. If the data has not been correctly transmitted (if the number of data to be transmitted in one frame could not be transmitted), the data must be retransmitted in the next frame, or the remaining data must be transmitted.

16.5.4 Slave reception

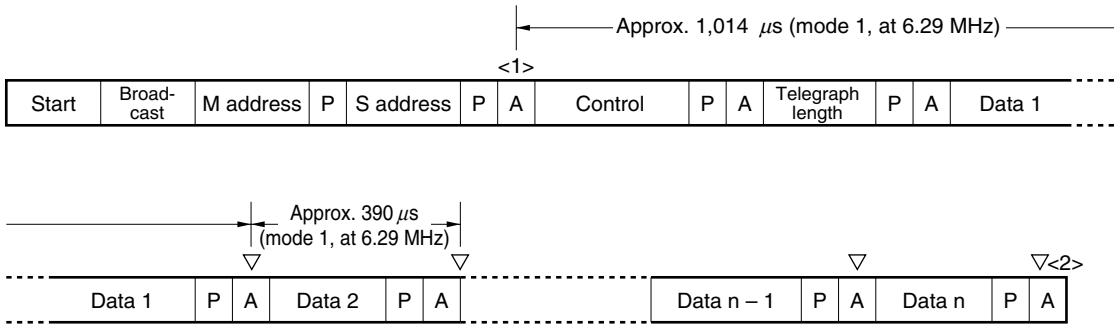
Initial preparation processing:

Set a unit address.

Communication start processing:

Set the BCR register (enable communication, disables slave transmission, and enables slave reception).

Figure 16-29. Slave Reception



<1> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of slave request → Slave processing

<2> Interrupt request signal (INTIE2, INTSTA) occurrence

Judgment of occurrence of error[Note] → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Frame end processing (See **16.5.4 (2) Frame end processing**).

Note This processing is necessary only when the INTIE2 interrupt request signal is used as the start interrupt, and is not necessary when the INTSTA interrupt request signal is used (in this case, the error processing is performed by using the INTERR interrupt request signal).

<R> **Remarks 1.** ▽: Interrupt request signal (INTIE1) occurrence (See **16.5.4 (1) Interrupt request signal (INTIE1) occurrence**).

2. n = Final number of data bytes

(1) Interrupt request signal (INTIE1) occurrence

If the NACK signal is transmitted in the data field, an interrupt request signal (INTIE1) is not issued to the INTC, and the same data is retransmitted from the master.

If the receive data is not read by the time the next data is received, the NACK signal is automatically transmitted.

(2) Frame end processing

In the vector interrupt servicing in <2> in Figure 16-29, it is judged whether the data has been correctly received within one frame.

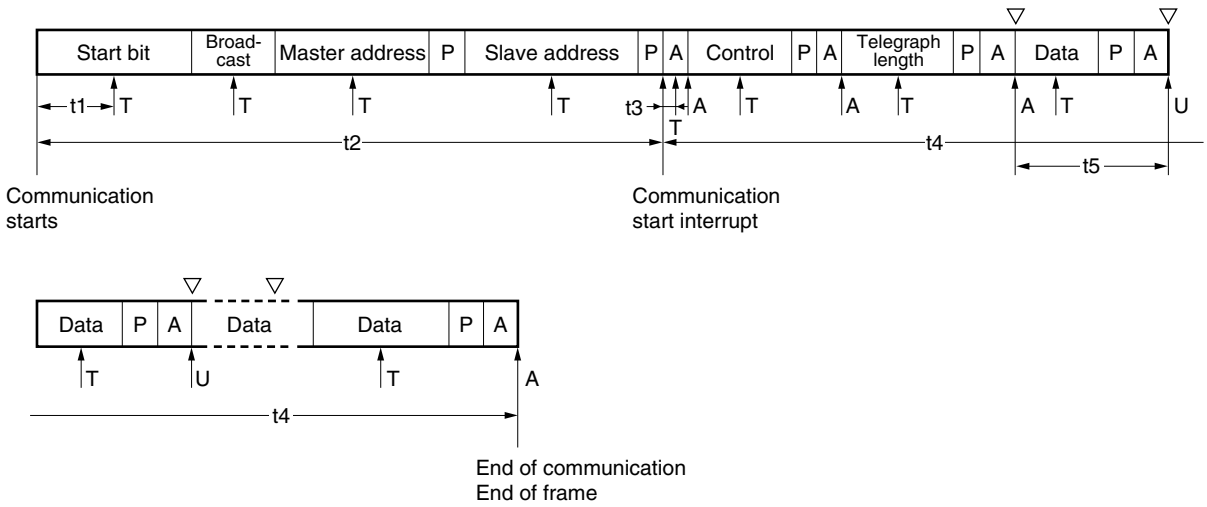
16.5.5 Interval of occurrence of interrupt request signal for IEBus control

Each control interrupt request signal must occur at each point of communication and perform the necessary processing until the next interrupt request signal occurs. Therefore, the IEBus control block is controlled by software, taking the shortest time of this interrupt request signal occurrence interval into consideration.

The locations at which the following interrupt request signals may occur are indicated by ↑ in the field where it may occur. ↑ does not mean that the interrupt request signal occurs at each of the points indicated by ↑. If an error interrupt request signal (timing error, parity error, or NACK receive error) occurs, the IEBus internal circuit is initialized. As a result, the following interrupt request signal does not occur in that communication frame.

(1) Master transmission

Figure 16-30. Master Transmission (Interval of Interrupt Request Signal Occurrence)

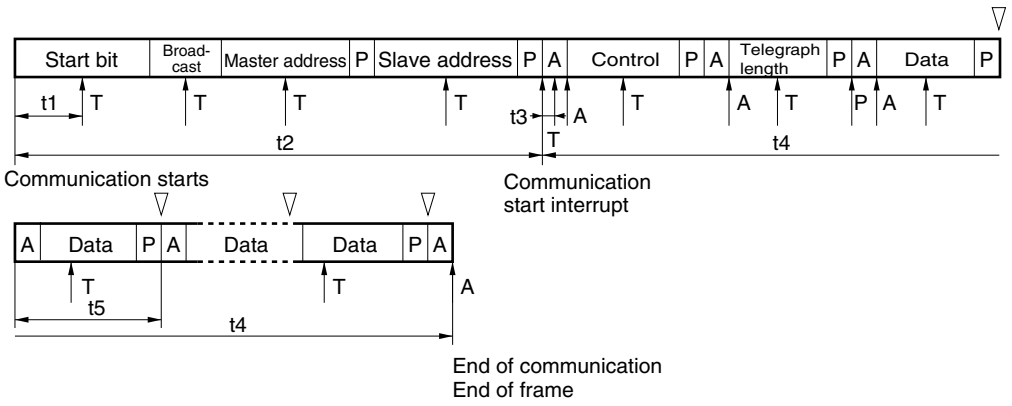


Remarks 1. T: Timing error
A: NACK receive error
U: Underrun error
▽: Data set interrupt (INTIE1)

2. End of frame occurs at the end of 32-byte data.
Values in parentheses indicate MIN. value (IEBus: mode 1, at 6.29 MHz).
t1: Communication starts → timing error (approx. 93 μs)
t2: Communication starts → communication start interrupt (approx. 1,282 μs)
t3: Communication start interrupt → timing error (approx. 15 μs)
t4: Communication start interrupt → end of communication (approx. 1,012 μs)
t5: Transmission data request interrupt interval (approx. 375 μs)

(2) Master reception

Figure 16-31. Master Reception (Interval of Interrupt Request Signal Occurrence)



Remarks 1. T: Timing error

P: Parity error

A: NACK receive error

▽: Data set interrupt request signal (INTIE1)

2. End of frame occurs at the end of 32-byte data.

Values in parentheses indicate MIN. value (IEBus: mode 1, at 6.29 MHz).

t1: Communication starts → timing error (approx. 93 μs)

t2: Communication starts → communication start interrupt (approx. 1,282 μs)

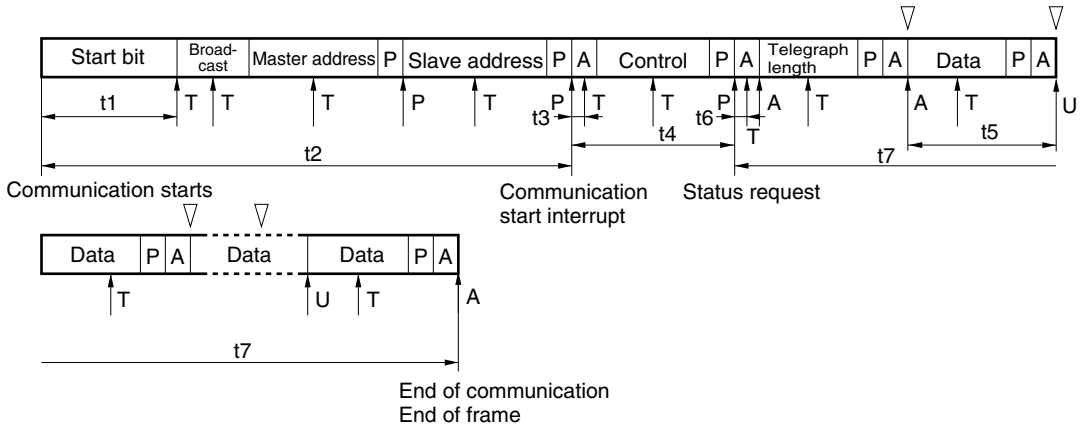
t3: Communication start interrupt → timing error (approx. 15 μs)

t4: Communication start interrupt → end of communication (approx. 1,012 μs)

t5: Receive data read interval (approx. 375 μs)

(3) Slave transmission

Figure 16-32. Slave Transmission (Interval of Interrupt Request Signal Occurrence)

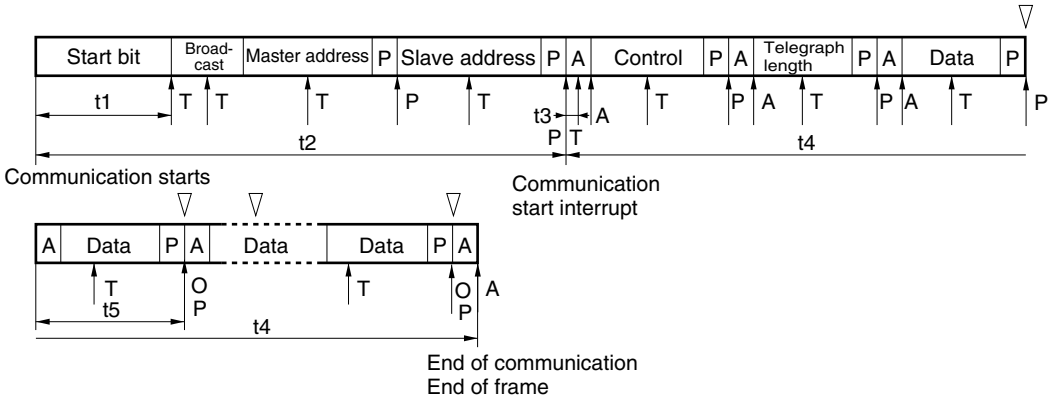


Remarks 1. T: Timing error
P: Parity error
A: NACK receive error
U: Underrun error
▽: Data set interrupt (INTIE1)

2. End of frame occurs at the end of 32-byte data.
Values in parentheses indicate MIN. value (IEBus: mode 1, at 6.29 MHz).
t1: Communication starts → timing error (approx. 196 μs)
t2: Communication starts → communication start interrupt (approx. 1,192 μs)
t3: Communication start interrupt → timing error (approx. 15 μs)
t4: Communication start interrupt → status request (approx. 225 μs)
t5: Transmission data request interrupt interval (approx. 375 μs)
t6: Status request → timing error (approx. 15 μs)
t7: Status request → end of communication (approx. 787 μs)

(4) Slave reception

Figure 16-33. Slave Reception (Interval of Interrupt Request Signal Occurrence)



Remarks 1. T: Timing error
P: Parity error
A: NACK receive error
O: Overrun error
▽: Data set interrupt (INTIE1)

2. End of frame occurs at the end of 32-byte data.
Values in parentheses indicate MIN. value (IEBus: mode 1, at 6.29 MHz).
t1: Communication starts → timing error (approx. 196 μs)
t2: Communication starts → communication start interrupt (approx. 1,192 μs)
t3: Communication start interrupt → timing error (approx. 15 μs)
t4: Communication start interrupt → end of communication (approx. 1,012 μs)
t5: Receive data read interval (approx. 375 μs)

<R>

CHAPTER 17 CAN CONTROLLER

The CAN controller is provided only in the μPD703253Y.

Caution The CAN controller is allocated in the programmable peripheral I/O area.
Before using the CAN controller, enable use of the programmable peripheral I/O area by using the BPC register.
For details, refer to 3.4.7 Programmable peripheral I/O registers.

17.1 Overview

The μPD703253Y features an on-chip 1-channel CAN (Controller Area Network) controller that complies with the CAN protocol as standardized in ISO 11898.

17.1.1 Features

- Compliant with ISO 11898 and tested according to ISO/DIS 16845 (CAN conformance test)
- Standard frame and extended frame transmission/reception enabled
- Transfer rate: 1 Mbps max. (CAN clock input ≥ 8 MHz)
- 32 message buffers/channels
- Receive/transmit history list function
- Automatic block transmission function
- Multi-buffer receive block function
- Mask setting of four patterns is possible for each channel

17.1.2 Overview of functions

Table 17-1 presents an overview of the CAN controller functions.

Table 17-1. Overview of Functions

Function	Details
Protocol	CAN protocol ISO 11898 (standard and extended frame transmission/reception)
Baud rate	Maximum 1 Mbps (CAN clock input ≥ 8 MHz)
Data storage	Storing messages in the CAN RAM
Number of messages	• 32 message buffers/channels • Each message buffer can be set to be either a transmit message buffer or a receive message buffer.
Message reception	• Unique ID can be set to each message buffer. • Mask setting of four patterns is possible for each channel. • A receive completion interrupt is generated each time a message is received and stored in a message buffer. • Two or more receive message buffers can be used as a FIFO receive buffer (multi-buffer receive block function). • Receive history list function
Message transmission	• Unique ID can be set to each message buffer. • Transmit completion interrupt for each message buffer • Message buffer numbers 0 to 7 specified as transmit message buffers can be used for automatic block transfer. Message transmission interval is programmable (automatic block transmission function (hereafter referred to as “ABT”)). • Transmission history list function
Remote frame processing	Remote frame processing by transmit message buffer
Diagnostic function	• Readable error counters • “Valid protocol operation flag” for verification of bus connections • Receive-only mode • Single-shot mode • CAN protocol error type decoding • Self-test mode
Release from bus-off state	• Can be forcibly released from bus-off by software (timing restrictions are ignored). • Cannot be automatically released from bus-off (release request by software is required).
Power save mode	• CAN sleep mode (can be woken up by CAN bus) • CAN stop mode (cannot be woken up by CAN bus)

17.1.3 Configuration

The CAN controller is composed of the following four blocks.

(1) NPB interface

This functional block provides an NPB (NEC Peripheral I/O Bus) interface and means of transmitting and receiving signals between the CAN module and the host CPU.

(2) MCM (Memory Control Module)

This functional block controls access to the CAN protocol layer and to the CAN RAM within the CAN module.

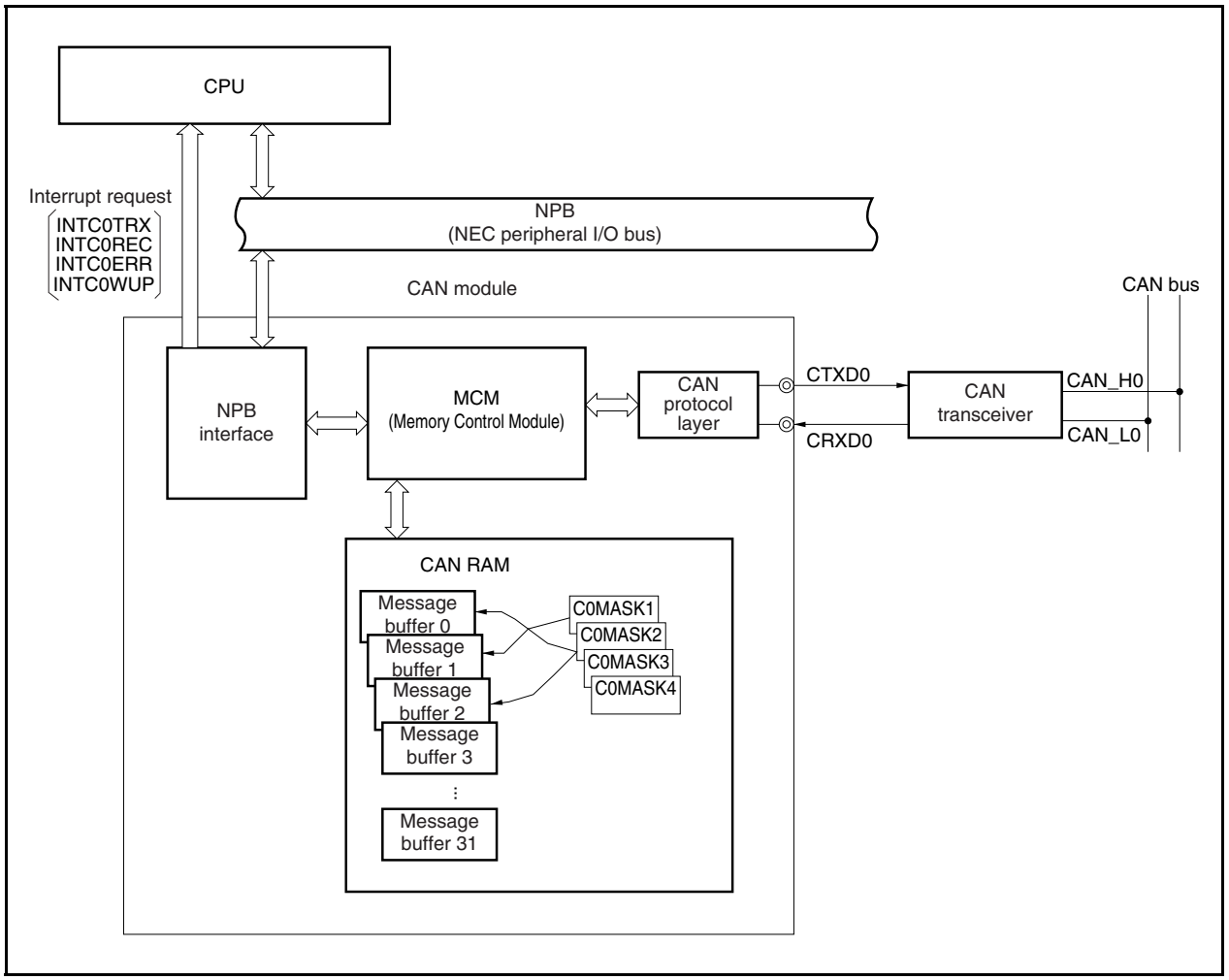
(3) CAN protocol layer

This functional block is involved in the operation of the CAN protocol and its related settings.

(4) CAN RAM

This is the CAN memory functional block, which is used to store message IDs, message data, etc.

Figure 17-1. Block Diagram of CAN Module

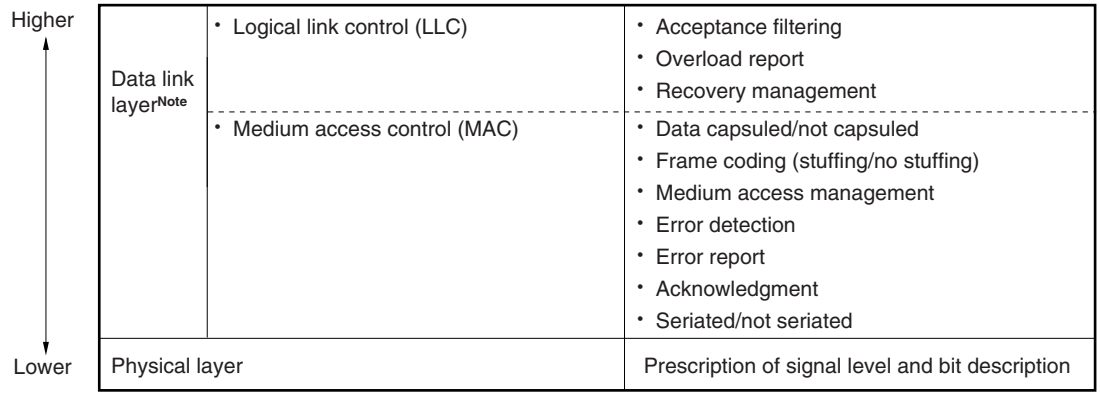


17.2 CAN Protocol

CAN (Controller Area Network) is a high-speed multiplex communication protocol for real-time communication in automotive applications (class C). CAN is prescribed by ISO 11898. For details, refer to the ISO 11898 specifications.

The CAN specification is generally divided into two layers: a physical layer and a data link layer. In turn, the data link layer includes logical link and medium access control. The composition of these layers is illustrated below.

Figure 17-2. Composition of Layers



Note CAN controller specification

17.2.1 Frame format

(1) Standard format frame

- The standard format frame uses 11-bit identifiers, which means that it can handle up to 2,048 messages.

(2) Extended format frame

- The extended format frame uses 29-bit (11 bits + 18 bits) identifiers, which increases the number of messages that can be handled to $2{,}048 \times 2^{18}$ messages.
- An extended format frame is set when “recessive level” (CMOS level of “1”) is set for both the SRR and IDE bits in the arbitration field.

17.2.2 Frame types

The following four types of frames are used in the CAN protocol.

Table 17-2. Frame Types

Frame Type	Description
Data frame	Frame used to transmit data
Remote frame	Frame used to request a data frame
Error frame	Frame used to report error detection
Overload frame	Frame used to delay the next data frame or remote frame

(1) Bus value

The bus values are divided into dominant and recessive.

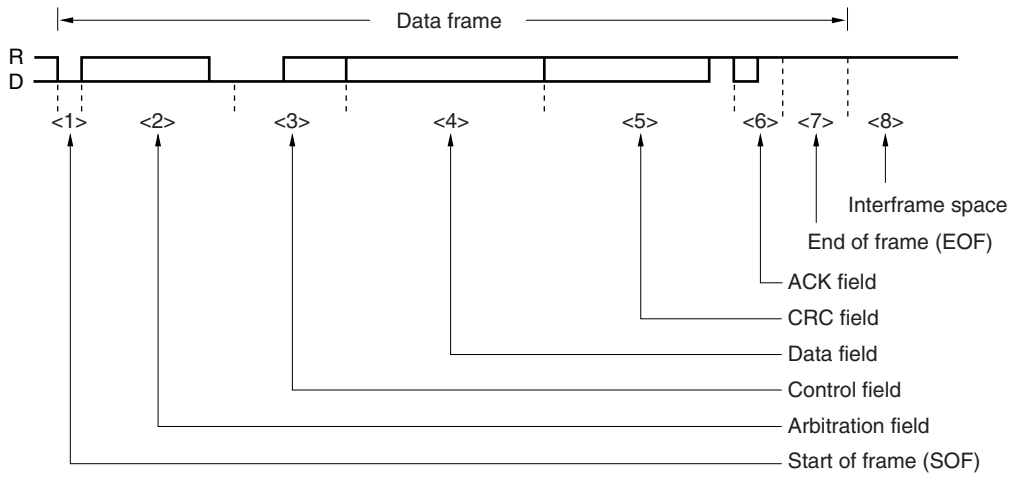
- Dominant level is indicated by logical 0.
- Recessive level is indicated by logical 1.
- When a dominant level and a recessive level are transmitted simultaneously, the bus value becomes dominant level.

17.2.3 Data frame and remote frame

(1) Data frame

A data frame is composed of seven fields.

Figure 17-3. Data Frame

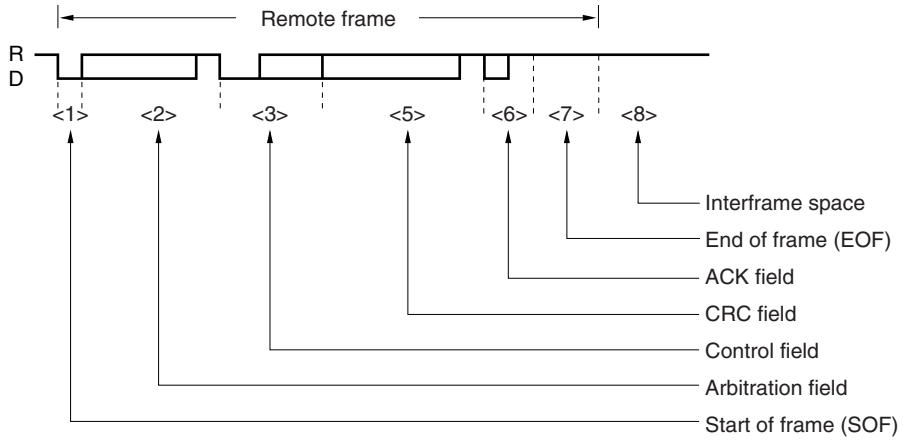


Remark D: Dominant = 0
R: Recessive = 1

(2) Remote frame

A remote frame is composed of six fields.

Figure 17-4. Remote Frame



Remarks 1. The data field is not transferred even if the control field's data length code is not "0000B".

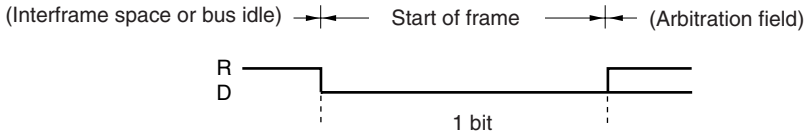
2. D: Dominant = 0
R: Recessive = 1

(3) Description of fields

<1> Start of frame (SOF)

The start of frame field is located at the start of a data frame or remote frame.

Figure 17-5. Start of Frame (SOF)



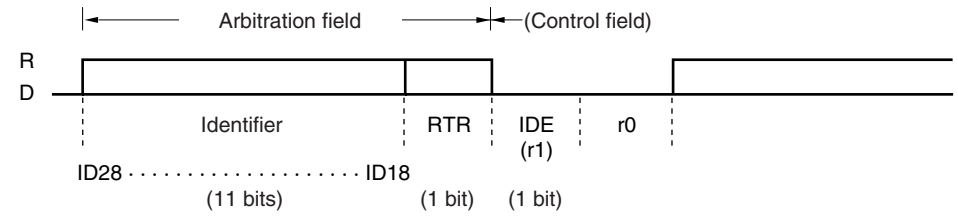
Remark D: Dominant = 0
R: Recessive = 1

- If a dominant level is detected in the bus idle state, a hardware synchronization is performed (the current TQ is assigned to be the SYNC segment).
- If a dominant level is sampled at the sample point following such a hardware synchronization, the bit is assigned to be a SOF. If a recessive level is detected, the protocol layer returns to the bus idle state and regards the preceding dominant pulse as a noise only. In this case an error frame is not generated.

<2> Arbitration field

The arbitration field is used to set the priority, data frame/remote frame, and frame format.

Figure 17-6. Arbitration Field (in Standard Format Mode)



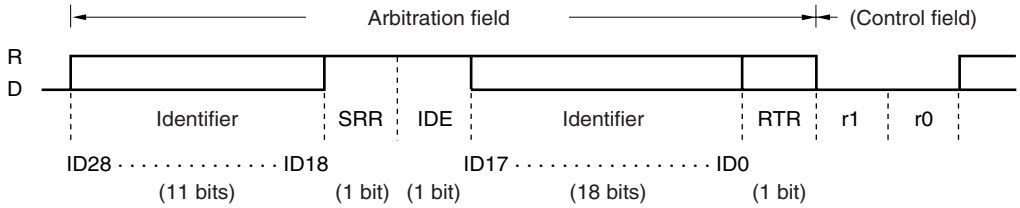
Cautions 1. ID28 to ID18 are identifiers.

2. An identifier is transmitted MSB first.

Remark D: Dominant = 0

R: Recessive = 1

Figure 17-7. Arbitration Field (in Extended Format Mode)



Cautions 1. ID28 to ID18 are identifiers.

2. An identifier is transmitted MSB first.

Remark D: Dominant = 0

R: Recessive = 1

Table 17-3. RTR Frame Settings

Frame Type	RTR Bit
Data frame	0 (D)
Remote frame	1 (R)

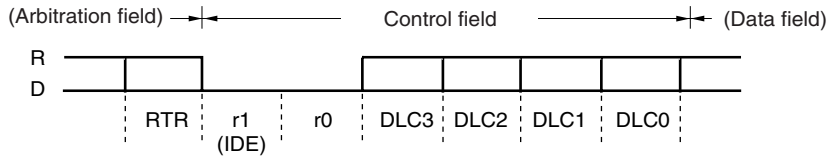
Table 17-4. Frame Format Setting (IDE Bit) and Number of Identifier (ID) Bits

Frame Format	SRR Bit	IDE Bit	Number of Bits
Standard format mode	None	0 (D)	11 bits
Extended format mode	1 (R)	1 (R)	29 bits

<3> Control field

The control field sets “DLC” as the number of data bytes in the data field (DLC = 0 to 8).

Figure 17-8. Control Field



Remark D: Dominant = 0
R: Recessive = 1

In a standard format frame, the control field’s IDE bit is the same as the r1 bit.

Table 17-5. Data Length Setting

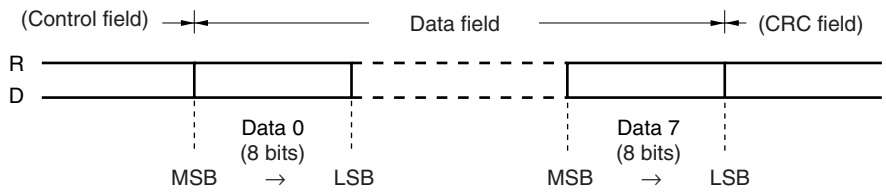
Data Length Code				Data Byte Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0 bytes
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	0	0	0	8 bytes
Other than above				8 bytes regardless of the value of DLC3 to DLC0

Caution In the remote frame, there is no data field even if the data length code is not 0000B.

<4> Data field

The data field contains the amount of data (byte units) set by the control field. Up to 8 units of data can be set.

Figure 17-9. Data Field

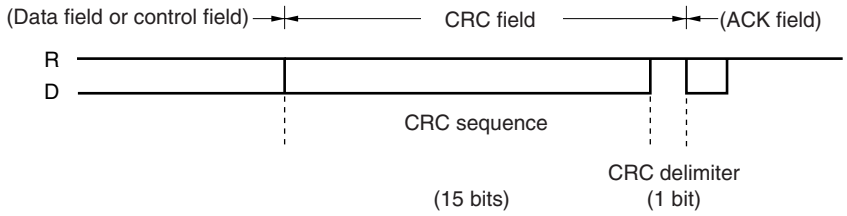


Remark D: Dominant = 0
R: Recessive = 1

<5> CRC field

The CRC field is a 16-bit field that is used to check for errors in transmit data.

Figure 17-10. CRC Field



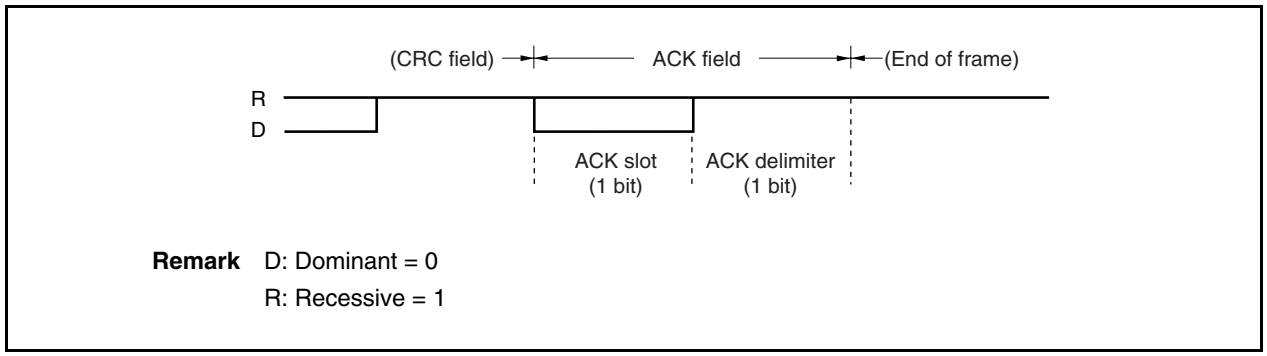
Remark D: Dominant = 0
R: Recessive = 1

- The polynomial P(X) used to generate the 15-bit CRC sequence is expressed as follows.
  $P(X) = X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1$
- Transmitting node: Transmits the CRC sequence calculated from the data (before bit stuffing) in the start of frame, arbitration field, control field, and data field.
- Receiving node: Compares the CRC sequence calculated using data bits that exclude the stuffing bits in the receive data with the CRC sequence in the CRC field. If the two CRC sequences do not match, the node issues an error frame.

<6> ACK field

The ACK field is used to acknowledge normal reception.

Figure 17-11. ACK Field



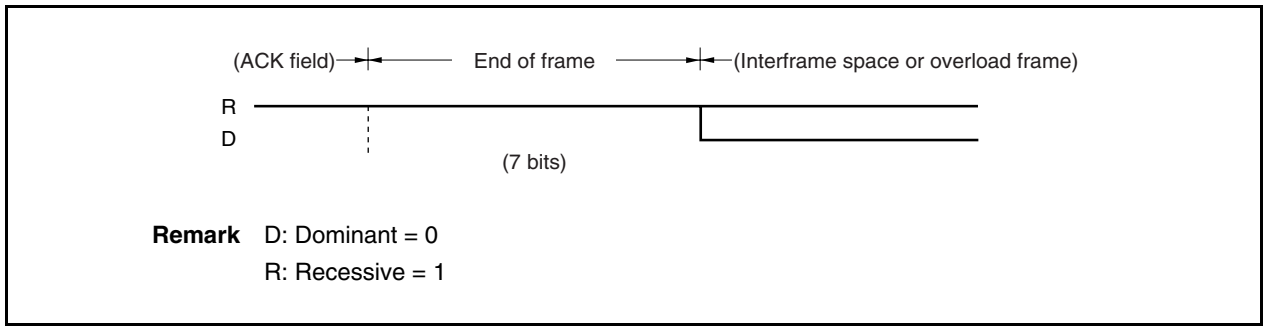
Remark D: Dominant = 0
R: Recessive = 1

- If no CRC error is detected, the receiving node sets the ACK slot to the dominant level.
- The transmitting node outputs two recessive-level bits.

<7> End of frame (EOF)

The end of frame field indicates the end of data frame/remote frame.

Figure 17-12. End of Frame (EOF)



Remark D: Dominant = 0
R: Recessive = 1

<8> Interframe space

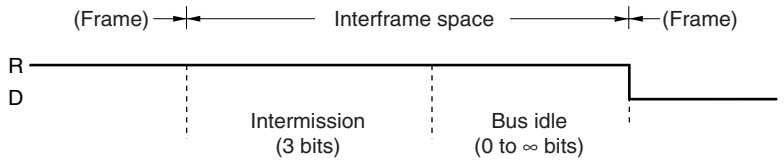
The interframe space is inserted after a data frame, remote frame, error frame, or overload frame to separate one frame from the next.

- The bus state differs depending on the error status.

(a) Error active node

The interframe space consists of a 3-bit intermission field and a bus idle field.

Figure 17-13. Interframe Space (Error Active Node)



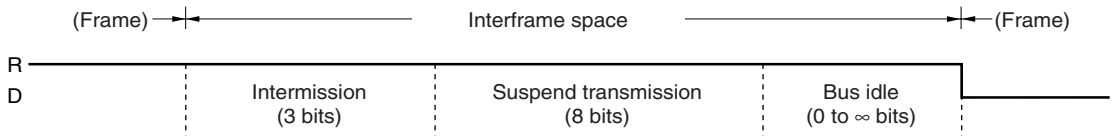
Remarks 1. Bus idle: State in which the bus is not used by any node.

2. D: Dominant = 0
R: Recessive = 1

(b) Error passive node

The interframe space consists of an intermission field, a suspend transmission field, and a bus idle field.

Figure 17-14. Interframe Space (Error Passive Node)



Remarks 1. Bus idle: State in which the bus is not used by any node.
Suspend transmission: Sequence of 8 recessive-level bits transmitted from the node in the error passive status.

2. D: Dominant = 0
R: Recessive = 1

Usually, the intermission field is 3 bits. If the transmitting node detects a dominant level at the third bit of the intermission field, however, it executes transmission.

- Operation in error status

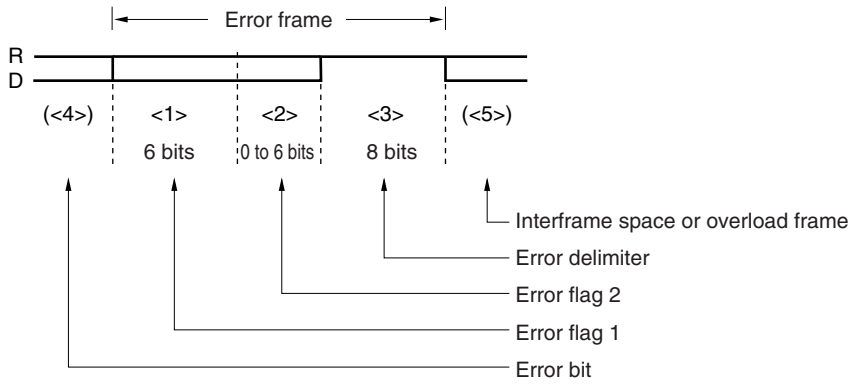
Table 17-6. Operation in Error Status

Error Status	Operation
Error active	A node in this status can transmit immediately after a 3-bit intermission.
Error passive	A node in this status can transmit 8 bits after the intermission.

17.2.4 Error frame

An error frame is output by a node that has detected an error.

Figure 17-15. Error Frame



Remark D: Dominant = 0
R: Recessive = 1

Table 17-7. Definition of Error Frame Fields

No.	Name	Bit Count	Definition
<1>	Error flag 1	6	Error active node: Outputs 6 dominant-level bits consecutively. Error passive node: Outputs 6 recessive-level bits consecutively. If another node outputs a dominant level while one node is outputting a passive error flag, the passive error flag is not cleared until the same level is detected 6 bits in a row.
<2>	Error flag 2	0 to 6	Nodes receiving error flag 1 detect bit stuff errors and issues this error flag.
<3>	Error delimiter	8	Outputs 8 recessive-level bits consecutively. If a dominant level is detected at the 8th bit, an overload frame is transmitted from the next bit.
<4>	Error bit	–	The bit at which the error was detected. The error flag is output from the bit next to the error bit. In the case of a CRC error, this bit is output following the ACK delimiter.
<5>	Interframe space/overload frame	–	An interframe space or overload frame starts from here.

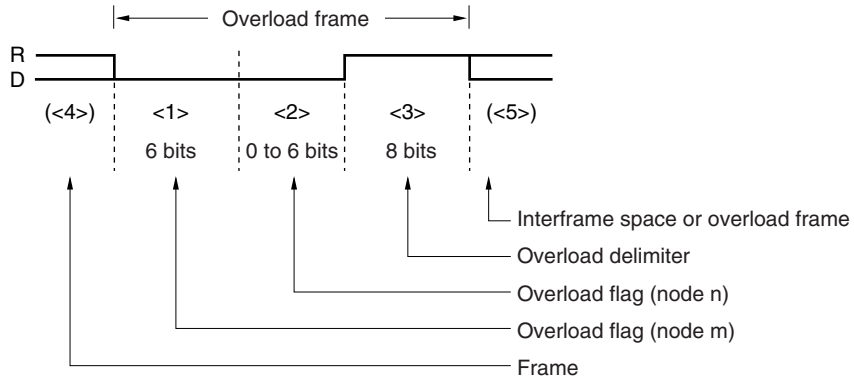
17.2.5 Overload frame

An overload frame is transmitted under the following conditions.

- When the receiving node has not completed the reception operation[Note]
- If a dominant level is detected at the first two bits during intermission
- If a dominant level is detected at the last bit (7th bit) of the end of frame or at the last bit (8th bit) of the error delimiter/overload delimiter

Note In this CAN controller, all receive frames can be loaded without outputting an overload frame because of the enough high-speed internal processing.

Figure 17-16. Overload Frame



Remark D: Dominant = 0
R: Recessive = 1
Node n ≠ node m

Table 17-8. Definition of Overload Frame Fields

No	Name	Bit Count	Definition
<1>	Overload flag	6	Outputs 6 dominant-level bits consecutively.
<2>	Overload flag from other node	0 to 6	The node that received an overload flag in the interframe space outputs an overload flag.
<3>	Overload delimiter	8	Outputs 8 recessive-level bits consecutively. If a dominant level is detected at the 8th bit, an overload frame is transmitted from the next bit.
<4>	Frame	–	Output following an end of frame, error delimiter, or overload delimiter.
<5>	Interframe space/overload frame	–	An interframe space or overload frame starts from here.

17.3 Functions

17.3.1 Determining bus priority

(1) When a node starts transmission:

- During bus idle, the node that output data first transmits the data.

(2) When more than one node starts transmission:

- The node that consecutively outputs the dominant level for the longest from the first bit of the arbitration field has the bus priority (if a dominant level and a recessive level are simultaneously transmitted, the dominant level is taken as the bus value).
- The transmitting node compares its output arbitration field and the data level on the bus.

Table 17-9. Determining Bus Priority

Level match	Continuous transmission
Level mismatch	Continuous transmission

(3) Priority of data frame and remote frame

- When a data frame and a remote frame are on the bus, the data frame has priority because its RTR bit, the last bit in the arbitration field, carries a dominant level.

Remark If the extended-format data frame and the standard-format remote frame conflict on the bus (if ID28 to ID18 of both of them are the same), the standard-format remote frame takes priority.

17.3.2 Bit stuffing

Bit stuffing is used to establish synchronization by appending 1 bit of inverted-level data if the same level continues for 5 bits, in order to prevent a burst error.

Table 17-10. Bit Stuffing

Transmission	During the transmission of a data frame or remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, 1 inverted-level bit of data is inserted before the following bit.
Reception	During the reception of a data frame or remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, reception is continued after deleting the next bit.

17.3.3 Multi masters

As the bus priority (a node which acquires transmission rights) is determined by the identifier, any node can be the bus master.

17.3.4 Multi cast

Although there is one transmitting node, two or more nodes can receive the same data at the same time because the same identifier can be set to two or more nodes.

17.3.5 CAN sleep mode/CAN stop mode function

The CAN sleep mode/CAN stop mode function puts the CAN controller in waiting mode to achieve low power consumption.

The controller is woken up from the CAN sleep mode by bus operation but it is not woken up from the CAN stop mode by bus operation (the CAN stop mode is controlled by CPU access).

17.3.6 Error control function

(1) Error types

Table 17-11. Error Types

Type	Description of Error		Detection State	
	Detection Method	Detection Condition	Transmission/ Reception	Field/Frame
Bit error	Comparison of the output level and level on the bus	Mismatch of levels	Transmitting/ receiving node	Bit that is outputting data on the bus at the start of frame to end of frame, error frame and overload frame.
Stuff error	Check of the receive data at the stuff bit	6 consecutive bits of the same output level	Receiving node	Start of frame to CRC sequence
CRC error	Comparison of the CRC sequence generated from the receive data and the received CRC sequence	Mismatch of CRC	Receiving node	CRC field
Form error	Field/frame check of the fixed format	Detection of fixed format violation	Receiving node	CRC delimiter ACK field End of frame Error frame Overload frame
ACK error	Check of the ACK slot by the transmitting node	Detection of recessive level in ACK slot	Transmitting node	ACK slot

(2) Output timing of error frame

Table 17-12. Output Timing of Error Frame

Type	Output Timing
Bit error, stuff error, form error, ACK error	Error frame output is started at the timing of the bit following the detected error.
CEC error	Error frame output is started at the timing of the bit following the ACK delimiter.

(3) Processing in case of error

The transmission node re-transmits the data frame or remote frame after the error frame. (However, it does not re-transmit the frame in the single-shot mode.)

(4) Error state

(a) Types of error states

The following three types of error states are defined by the CAN specification.

- Error active
- Error passive
- Bus-off

These types of error states are classified by the values of the C0ERC.TEC7 to C0ERC.TEC0 bits (transmission error counter bits) and the C0ERC.REC6 to C0ERC.REC0 bits (reception error counter bits) as shown in Table 17-13.

The present error state is indicated by the C0INFO register.

When each error counter value becomes equal to or greater than the error warning level (96), the C0INFO.TECS0 or C0INFO.RECS0 bit is set to 1. In this case, the bus state must be tested because it is considered that the bus has a serious fault. An error counter value of 128 or more indicates an error passive state and the TECS1 or RECS1 bit is set to 1.

- If the value of the transmission error counter is greater than or equal to 256 (actually, the transmission error counter does not indicate a value greater than or equal to 256), the bus-off state is reached and the C0INFO.BOFF bit is set to 1.
- If only one node is active on the bus at startup (i.e., when the bus is connected only to the local station), ACK is not returned even if data is transmitted. Consequently, re-transmission of the error frame and data is repeated. In the error passive state, however, the transmission error counter is not incremented and the bus-off state is not reached.

Table 17-13. Types of Error States

Type	Operation	Value of Error Counter	Indication of C0INFO Register	Operation Specific to Error State
Error active	Transmission	0 to 95	TECS1, TECS0 = 00	• Outputs an active error flag (6 consecutive dominant-level bits) on detection of the error.
	Reception	0 to 95	RECS1, RECS0 = 00	
	Transmission	96 to 127	TECS1, TECS0 = 01	
	Reception	96 to 127	RECS1, RECS0 = 01	
Error passive	Transmission	128 to 255	TECS1, TECS0 = 11	• Outputs a passive error flag (6 consecutive recessive-level bits) on detection of the error. • Transmits 8 recessive-level bits, in between transmissions, following an intermission (suspend transmission).
	Reception	128 or more	RECS1, RECS0 = 11	
Bus-off	Transmission	256 or more (not indicated)[Note]	BOFF = 1, TECS1, TECS0 = 11	• Communication is not possible. However, when the frame is received, no messages are stored and the following operations are performed. <1> REC is incremented/decremented. <2> VALID bit is set. • If the initialization mode is set, after request to transit to an operation mode other than the initialization mode, 11 consecutive recessive-level bits are generated 128 times, and then the error counter is reset to 0 and the error active state can be restored.

Note The value of the transmit error counter (TEC) does not carry any meaning if BOFF has been set. If an error that increments the value of the transmission error counter by 8 while the counter value is in a range of 248 to 255 occurs, the counter is not incremented and the bus-off state is assumed.

(b) Error counter

The error counter counts up when an error has occurred, and counts down upon successful transmission and reception. The error counter counts up immediately after error detection.

Table 17-14. Error Counter

State	Transmission Error Counter (TEC7 to TEC0 Bits)	Reception Error Counter (REC6 to REC0 Bits)
Receiving node detects an error (except bit error in the active error flag or overload flag).	No change	+1 (REPS bit = 0)
Receiving node detects dominant level following error flag of error frame.	No change	+8 (REPS bit = 0)
Transmitting node transmits an error flag. [As exceptions, the error counter does not change in the following cases.] <1> ACK error is detected in error passive state and dominant level is not detected while the passive error flag is being output. <2> A stuff error is detected in an arbitration field that transmitted a recessive level as a stuff bit, but a dominant level is detected.	+8	No change
Bit error detection while active error flag or overload flag is being output (error-active transmitting node)	+8	No change
Bit error detection while active error flag or overload flag is being output (error-active receiving node)	No change	+8 (REPS bit = 0)
When the node detects 14 consecutive dominant-level bits from the beginning of the active error flag or overload flag, and then subsequently detects 8 consecutive dominant-level bits. When the node detects 8 consecutive dominant levels after a passive error flag	+8 (transmitting)	+8 (receiving, REPS bit = 0)
When the transmitting node has completed transmission without error (±0 if error counter = 0)	–1	No change
When the receiving node has completed reception without error	No change	• –1 (1 ≤ REC6 to REC0 ≤ 127, REPS bit = 0) • ±0 (REC6 to REC0 = 0, REPS bit = 0) • Any value of 119 to 127 is set (REPS bit = 1)

(c) Occurrence of bit error in intermission

An overload frame is generated.

Caution If an error occurs, it is controlled according to the contents of the transmission error counter and reception error counter before the error occurred. The value of the error counter is incremented after the error flag has been output.

(5) Recovery from bus-off state

When the CAN module is in the bus-off state, the transmission pins (CTXD0) cut off from the CAN bus always output the recessive level.

The CAN module recovers from the bus-off state in the following bus-off recovery sequence.

<1> Request to enter the CAN initialization mode

<2> Request to enter a CAN operation mode

(a) Recovery operation through normal recovery sequence

(b) Forced recovery operation that skips recovery sequence

(a) Recovery from bus-off state through normal recovery sequence

The CAN module first issues a request to enter the initialization mode (refer to timing <1> in Figure 17-17). This request will be immediately acknowledged, and the C0CTRL.OPMODE2 to OPMODE0 bit is cleared to 000B. Processing such as analyzing the fault that has caused the bus-off state, re-defining the CAN module and message buffer using application software, or stopping the operation of the CAN module can be performed by clearing the C0GMCTRL.GOM bit to 0.

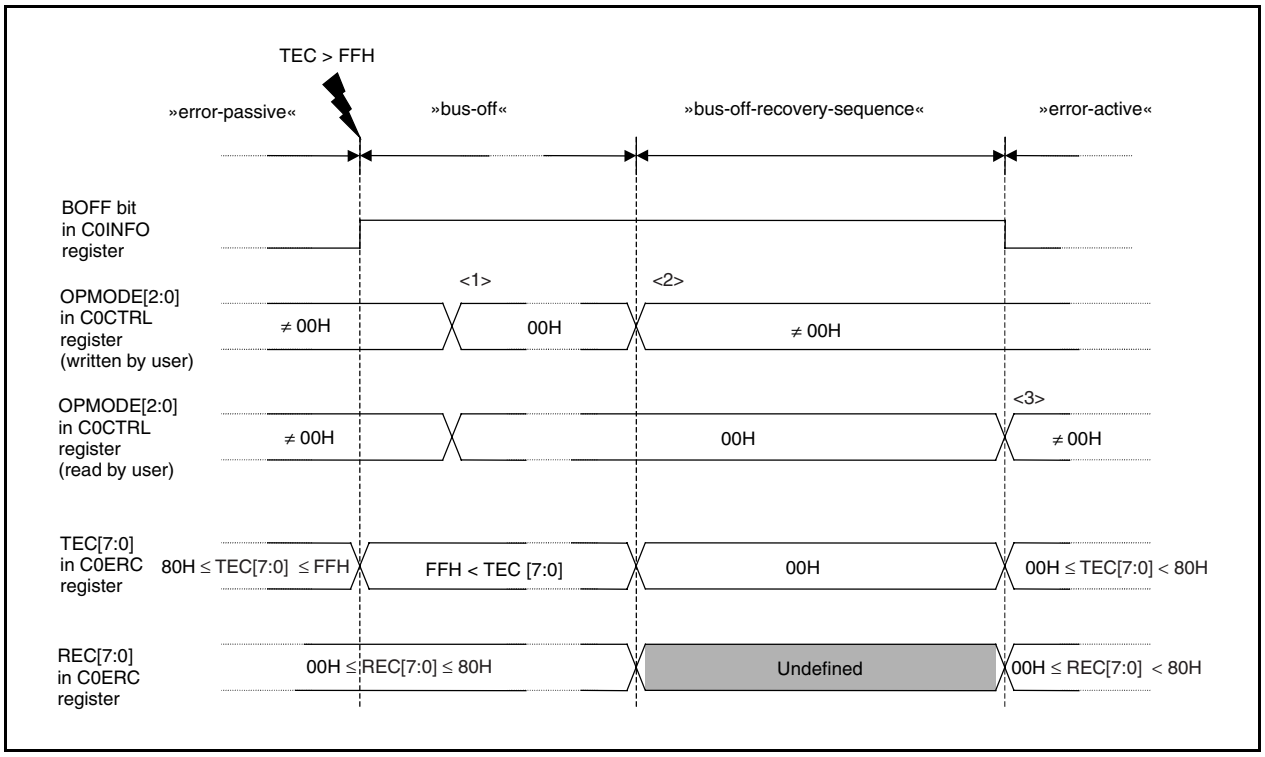
Next, the module requests to change the mode from the initialization mode to an operation mode (refer to timing <2> in Figure 17-17). This starts an operation to recover the CAN module from the bus-off state. The conditions under which the module can recover from the bus-off state are defined by the CAN protocol ISO 11898, and it is necessary to detect 11 consecutive recessive-level bits more than 128 times At this time, the request to change the mode to an operation mode is held pending until the recovery conditions are satisfied. When the recovery conditions are satisfied (refer to timing <3> in Figure 17-17), the CAN module can enter the operation mode it has requested. Until the CAN module enters this operation mode, it stays in the initialization mode. Whether the CAN module has completed transition to any other operation mode can be confirmed by reading OPMODE2 to OPMODE0. Before transition to any other operation mode is completed, OPMODE2 to OPMODE0 bits = 000B is read.

During the bus-off period and bus-off recovery sequence, the C0INFO.BOFF bit stays set (to 1). In the bus-off recovery sequence, the reception error counter (C0ERC.REC0 to C0ERC.REC6) counts the number of times 11 consecutive recessive-level bits have been detected on the bus. Therefore, the recovery state can be checked by reading the REC0 to REC6 bits.

Cautions 1. If a request to change the mode from the initialization mode to any operation mode to execute the bus-off recovery sequence again during a bus-off recovery sequence, the bus-off recovery sequence starts from the beginning and 11 contiguous recessive bits are counted 128 times again on the bus.

2. In the bus-off recovery sequence, the REC0 to REC6 bits counts up (+1) each time 11 consecutive recessive-level bits have been detected. Even during the bus-off period, the CAN module can enter the CAN sleep mode or CAN stop mode. To be released from the bus-off state, the module must enter the initialization mode once. If the module is in the CAN sleep mode or CAN stop mode, however, it cannot directly enter the initialization mode. In this case, the bus off recovery sequence is started at the same time as the CAN sleep mode is released even without shifting to the initialization mode. In addition to clearing the C0CTRL.PSMODE1 and C0CTRL.PSMODE0 bits by software, the bus off recovery sequence is also started due to wakeup by dominant edge detection on the CAN bus (when the CAN clock is supplied, the PSMODE0 bit must be cleared by software after the dominant edge has been detected).

Figure 17-17. Recovery from Bus-off State Through Normal Recovery Sequence



(b) Forced recovery operation that skips bus-off recovery sequence

The CAN module can be forcibly released from the bus-off state, regardless of the bus state, by skipping the bus-off recovery sequence. Here is the procedure.

First, the CAN module requests to enter the initialization mode. For the operation and points to be noted at this time, see **17.3.6 (5) (a) Recovery from bus-off state through normal recovery sequence**.

Next, the module requests to enter an operation mode. At the same time, the C0CTRL.CCERC bit must be set to 1.

As a result, the bus-off recovery sequence defined by the CAN protocol ISO 11898 is skipped, and the module immediately enters the operation mode. In this case, the module is connected to the CAN bus after it has monitored 11 consecutive recessive-level bits. For details, refer to the processing in Figure 17-53.

Caution This function is not defined by the CAN protocol ISO 11898. When using this function, thoroughly evaluate its effect on the network system.

(6) Initializing CAN module error counter register (C0ERC) in initialization mode

If it is necessary to initialize the C0ERC and C0INFO registers for debugging or evaluating a program, they can be initialized to the default value by setting the C0CTRL.CCERC bit in the initialization mode. When initialization has been completed, the CCERC bit is automatically cleared to 0.

Cautions 1. This function is enabled only in the initialization mode. Even if the CCERC bit is set to 1 in a CAN operation mode, the C0ERC and C0INFO registers are not initialized.

2. The CCERC bit can be set at the same time as the request to enter a CAN operation mode.

17.3.7 Baud rate control function

(1) Prescaler

The CAN controller has a prescaler that divides the clock (f_{CAN}) supplied to CAN. This prescaler generates a CAN protocol layer base clock (f_{TQ}) that is the CAN module system clock (f_{CANMOD}) divided by 1 to 256 (see **17.6 (12) CAN0 module bit rate prescaler register (C0BRP)**).

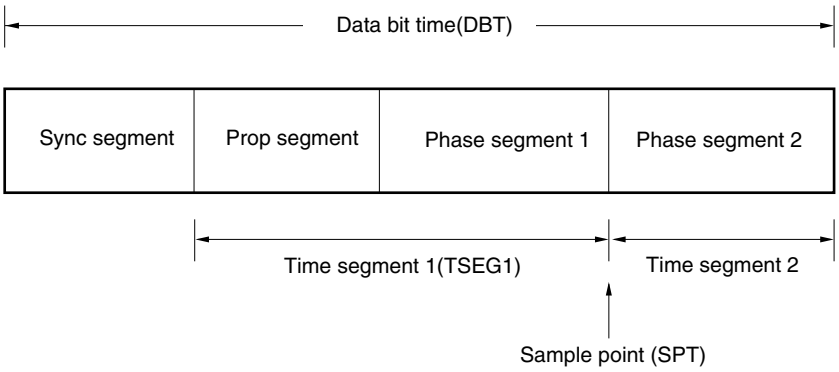
(2) Data bit time (8 to 25 time quanta)

One data bit time is defined as shown in **Figure 17-18**.

1 Time Quanta = $1/f_{TQ}$

The CAN controller sets the data bit time by replacing it with the bit timing parameters such as time segment 1, time segment 2, and reSynchronization Jump Width (SJW), as shown in Figure 17-18. Time segment 1 is equivalent to the total of the propagation (prop) segment and phase segment 1 that are defined by the CAN protocol specification. Time segment 2 is equivalent to phase segment 2.

Figure 17-18. Segment Setting

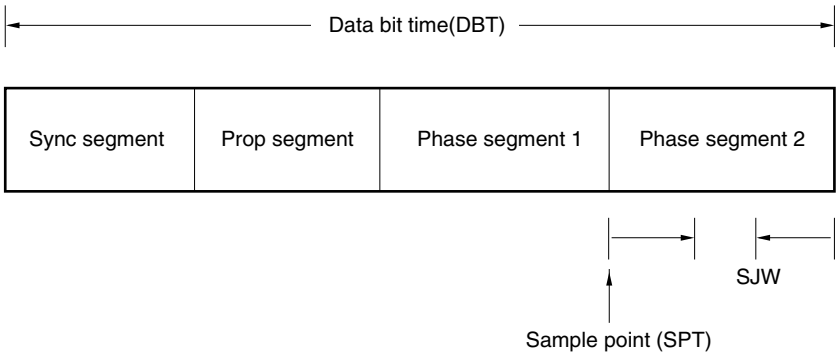


Segment name	Settable range	Notes on setting to conform to CAN specification
Time segment 1 (TSEG1)	2TQ to 16TQ	–
Time segment 2 (TSEG2)	1TQ to 8TQ	IPT of the CAN controller is 0TQ. To conform to the CAN protocol specification, therefore, a length equal to or less than phase segment 1 must be set here. This means that the length of time segment 1 minus 1TQ is the settable upper limit of time segment 2.
reSynchronization Jump Width (SJW)	1TQ to 4TQ	The length of time segment 1 minus 1TQ or 4TQ, whichever smaller.

Remark IPT: Information Processing Time
TQ: Time Quanta

Remark The CAN protocol specification defines the segments constituting the data bit time as shown in Figure 17-19.

Figure 17-19. Configuration of Data Bit Time Defined by CAN Specification



Segment name	Segment length	Description
Sync segment (Synchronization segment)	1	This segment starts at the edge where the level changes from recessive to dominant when hardware synchronization is established.
Prop segment (Propagation segment)	Programmable to 1 to 8, or greater	This segment absorbs the delay of the output buffer, CAN bus, and input buffer. The length of this segment is set so that ACK is returned before the start of phase segment 1. Time of prop segment ≥ (Delay of output buffer) + 2 × (Delay of CAN bus) + (Delay of input buffer)
Phase segment 1 (Phase buffer segment 1)	Programmable to 1 to 8	
Phase segment 2 (Phase buffer segment 2)	Phase segment 1 or IPT, whichever greater	This segment compensates for an error in the data bit time. The longer this segment, the wider the permissible range but the slower the communication speed.
SJW (reSynchronization Jump Width)	Programmable from 1TQ to segment 1TQ to 4TQ, whichever is smaller	This width sets the upper limit of expansion or contraction of the phase segment during resynchronization.

Remark IPT: Information Processing Time
TQ: Time Quanta

(3) Synchronizing data bit

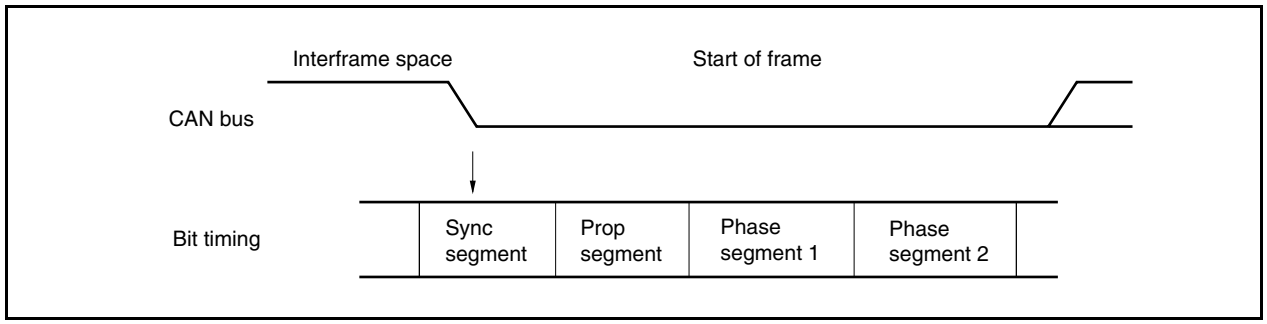
- The receiving node establishes synchronization by a level change on the bus because it does not have a sync signal.
- The transmitting node transmits data in synchronization with the bit timing of the transmitting node.

(a) Hardware synchronization

This synchronization is established when the receiving node detects the start of frame in the interframe space.

- When a falling edge is detected on the bus, that TQ means the sync segment and the next segment is the prop segment. In this case, synchronization is established regardless of SJW.

Figure 17-20. Hardware Synchronization to Detect Dominant Level During Bus Idle

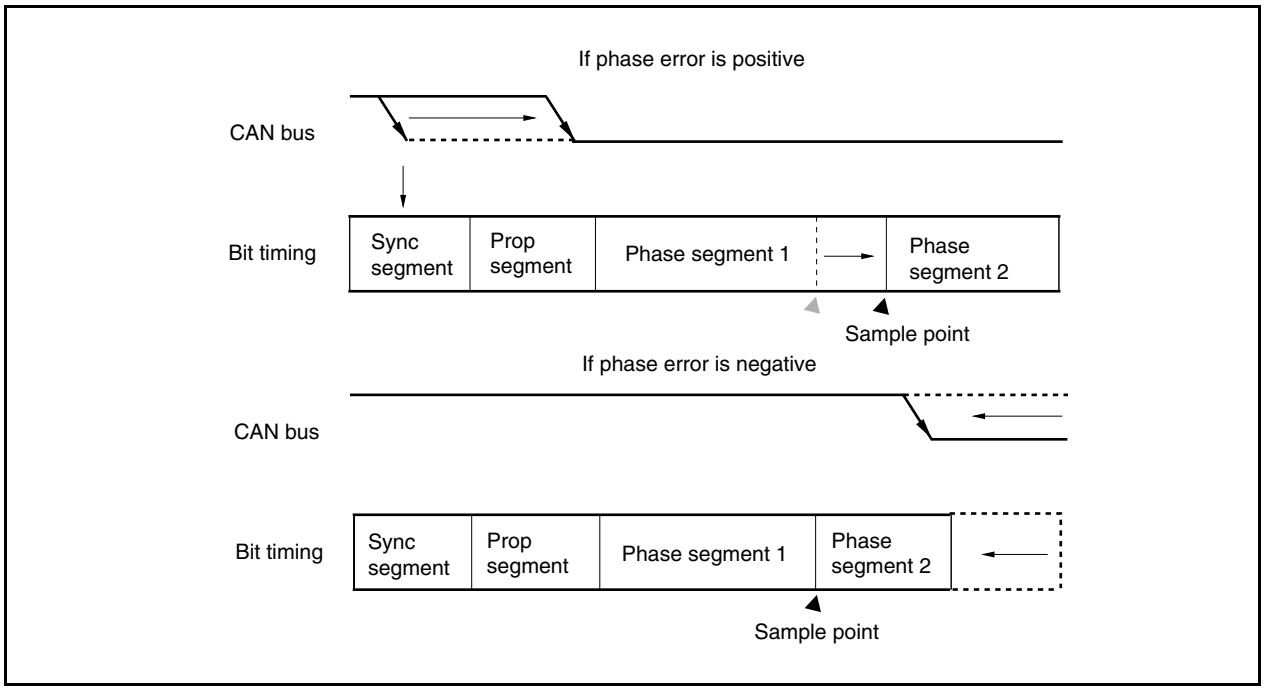


(b) Resynchronization

Synchronization is established again if a level change is detected on the bus during reception (only if a recessive level was sampled previously).

- The phase error of the edge is given by the relative position of the detected edge and sync segment.
  <Sign of phase error>
  0: If the edge is within the sync segment
  Positive: If the edge is before the sample point (phase error)
  Negative: If the edge is after the sample point (phase error)
  If phase error is positive: Phase segment 1 is longer by specified SJW.
  If phase error is negative: Phase segment 2 is shorter by specified SJW.
- The sample point of the data of the receiving node moves relatively due to the “discrepancy” in the baud rate between the transmitting node and receiving node.

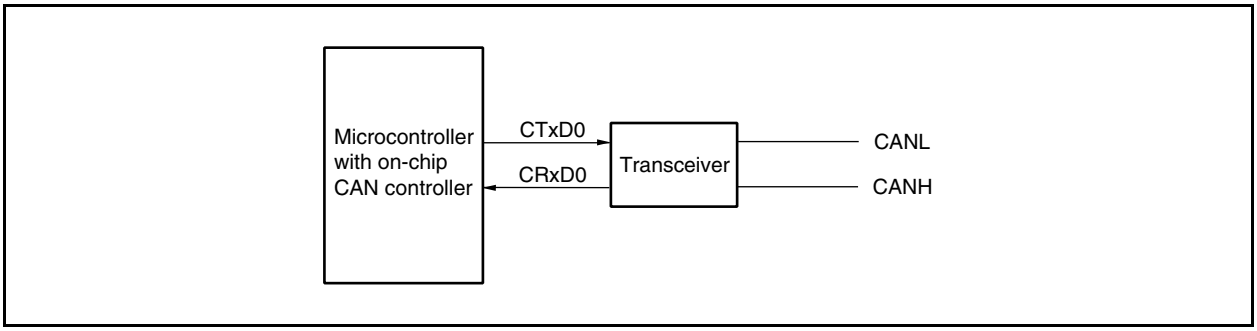
Figure 17-21. Resynchronization



17.4 Connection with Target System

The microcontroller with on-chip CAN controller has to be connected to the CAN bus using an external transceiver.

Figure 17-22. Connection to CAN Bus



17.5 Internal Registers of CAN Controller

17.5.1 CAN controller configuration

Table 17-15. List of CAN Controller Registers

Item	Register Name
CAN global registers	CAN0 global control register (C0GMCTRL)
	CAN0 global clock selection register (C0GMCS)
	CAN0 global automatic block transmission control register (C0GMABT)
	CAN0 global automatic block transmission delay setting register (C0GMABTD)
CAN module registers	CAN0 module mask 1 register (C0MASK1L, C0MASK1H)
	CAN0 module mask 2 register (C0MASK2L, C0MASK2H)
	CAN0 module mask 3 register (C0MASK3L, C0MASK3H)
	CAN0 module mask 4 registers (C0MASK4L, C0MASK4H)
	CAN0 module control register (C0CTRL)
	CAN0 module last error information register (C0LEC)
	CAN0 module information register (C0INFO)
	CAN0 module error counter register (C0ERC)
	CAN0 module interrupt enable register (C0IE)
	CAN0 module interrupt status register (C0INTS)
	CAN0 module bit rate prescaler register (C0BRP)
	CAN0 module bit rate register (C0BTR)
	CAN0 module last in-pointer register (C0LIPT)
	CAN0 module receive history list register (C0RGPT)
	CAN0 module last out-pointer register (C0LOPT)
	CAN0 module transmit history list register (C0TGPT)
Message buffer registers	CAN0 message data byte 01 register m (C0MDATA01m)
	CAN0 message data byte 0 register m (C0MDATA0m)
	CAN0 message data byte 1 register m (C0MDATA1m)
	CAN0 message data byte 23 register m (C0MDATA23m)
	CAN0 message data byte 2 register m (C0MDATA2m)
	CAN0 message data byte 3 register m (C0MDATA3m)
	CAN0 message data byte 45 register m (C0MDATA45m)
	CAN0 message data byte 4 register m (C0MDATA4m)
	CAN0 message data byte 5 register m (C0MDATA5m)
	CAN0 message data byte 67 register m (C0MDATA67m)
	CAN0 message data byte 6 register m (C0MDATA6m)
	CAN0 message data byte 7 register m (C0MDATA7m)
	CAN0 message data length register m (C0MDLCm)
	CAN0 message configuration register m (C0MCONFm)
	CAN0 message ID register m (C0MIDLm, C0MIDHm)
	CAN0 message control register m (C0MCTRLm)

Remarks 1. The CAN global register is defined as C0GM <register function>.
The CAN module register is defined as C0 <register function>.
The message buffer register is defined as C0M <register function>.

2. m = 00 to 31

17.5.2 Register access type

Table 17-16. Register Access Types (1/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC000H	CAN0 global control register	C0GMCTRL	R/W			√	0000H
03FEC002H	CAN0 global clock selection register	C0GMCS			√		0FH
03FEC006H	CAN0 global automatic block transmission register	C0GMABT				√	0000H
03FEC008H	CAN0 global automatic block transmission delay setting register	C0GMABTD			√		00H
03FEC040H	CAN0 module mask 1 register	C0MASK1L				√	Undefined
03FEC042H		C0MASK1H				√	Undefined
03FEC044H	CAN0 module mask 2 register	C0MASK2L				√	Undefined
03FEC046H		C0MASK2H				√	Undefined
03FEC048H	CAN0 module mask 3 register	C0MASK3L				√	Undefined
03FEC04AH		C0MASK3H				√	Undefined
03FEC04CH	CAN0 module mask 4 register	C0MASK4L				√	Undefined
03FEC04EH		C0MASK4H				√	Undefined
03FEC050H	CAN0 module control register	C0CTRL				√	0000H
03FEC052H	CAN0 module last error code register	C0LEC			√		00H
03FEC053H	CAN0 module information register	C0INFO	R		√		00H
03FEC054H	CAN0 module error counter register	C0ERC				√	0000H
03FEC056H	CAN0 module interrupt enable register	C0IE	R/W			√	0000H
03FEC058H	CAN0 module interrupt status register	C0INTS				√	0000H
03FEC05AH	CAN0 module bit-rate prescaler register	C0BRP			√		FFH
03FEC05CH	CAN0 module bit-rate register	C0BTR				√	370FH
03FEC05EH	CAN0 module last in-pointer register	C0LIPT	R		√		Undefined
03FEC060H	CAN0 module receive history list register	C0RGPT	R/W			√	xx02H
03FEC062H	CAN0 module last out-pointer register	C0LOPT	R		√		Undefined
03FEC064H	CAN0 module transmit history list register	C0TGPT	R/W			√	xx02H

Table 17-16. Register Access Types (2/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC100H	CAN0 message data byte 01 register 00	C0MDATA0100	R/W			√	Undefined
03FEC100H	CAN0 message data byte 0 register 00	C0MDATA000			√		Undefined
03FEC101H	CAN0 message data byte 1 register 00	C0MDATA100			√		Undefined
03FEC102H	CAN0 message data byte 23 register 00	C0MDATA2300				√	Undefined
03FEC102H	CAN0 message data byte 2 register 00	C0MDATA200			√		Undefined
03FEC103H	CAN0 message data byte 3 register 00	C0MDATA300			√		Undefined
03FEC104H	CAN0 message data byte 45 register 00	C0MDATA4500				√	Undefined
03FEC104H	CAN0 message data byte 4 register 00	C0MDATA400			√		Undefined
03FEC105H	CAN0 message data byte 5 register 00	C0MDATA500			√		Undefined
03FEC106H	CAN0 message data byte 67 register 00	C0MDATA6700				√	Undefined
03FEC106H	CAN0 message data byte 6 register 00	C0MDATA600			√		Undefined
03FEC107H	CAN0 message data byte 7 register 00	C0MDATA700			√		Undefined
03FEC108H	CAN0 message data length register 00	C0MDLC00			√		0000xxxxB
03FEC109H	CAN0 message configuration register 00	C0MCONF00			√		Undefined
03FEC10AH	CAN0 message identifier register 00	C0MIDL00				√	Undefined
03FEC10CH		C0MIDH00				√	Undefined
03FEC10EH	CAN0 message control register 00	C0MCTRL00				√	00x00000 000xx000B
03FEC120H	CAN0 message data byte 01 register 01	C0MDATA0101				√	Undefined
03FEC120H	CAN0 message data byte 0 register 01	C0MDATA001			√		Undefined
03FEC121H	CAN0 message data byte 1 register 01	C0MDATA101			√		Undefined
03FEC122H	CAN0 message data byte 23 register 01	C0MDATA2301				√	Undefined
03FEC122H	CAN0 message data byte 2 register 01	C0MDATA201			√		Undefined
03FEC123H	CAN0 message data byte 3 register 01	C0MDATA301			√		Undefined
03FEC124H	CAN0 message data byte 45 register 01	C0MDATA4501				√	Undefined
03FEC124H	CAN0 message data byte 4 register 01	C0MDATA401			√		Undefined
03FEC125H	CAN0 message data byte 5 register 01	C0MDATA501			√		Undefined
03FEC126H	CAN0 message data byte 67 register 01	C0MDATA6701				√	Undefined
03FEC126H	CAN0 message data byte 6 register 01	C0MDATA601			√		Undefined
03FEC127H	CAN0 message data byte 7 register 01	C0MDATA701			√		Undefined
03FEC128H	CAN0 message data length register 01	C0MDLC01			√		0000xxxxB
03FEC129H	CAN0 message configuration register 01	C0MCONF01			√		Undefined
03FEC12AH	CAN0 message identifier register 01	C0MIDL01				√	Undefined
03FEC12CH		C0MIDH01				√	Undefined
03FEC12EH	CAN0 message control register 01	C0MCTRL01				√	00x00000 000xx000B

Table 17-16. Register Access Types (3/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC140H	CAN0 message data byte 01 register 02	C0MDATA0102	R/W			√	Undefined
03FEC140H	CAN0 message data byte 0 register 02	C0MDATA002			√		Undefined
03FEC141H	CAN0 message data byte 1 register 02	C0MDATA102			√		Undefined
03FEC142H	CAN0 message data byte 23 register 02	C0MDATA2302				√	Undefined
03FEC142H	CAN0 message data byte 2 register 02	C0MDATA202			√		Undefined
03FEC143H	CAN0 message data byte 3 register 02	C0MDATA302			√		Undefined
03FEC144H	CAN0 message data byte 45 register 02	C0MDATA4502				√	Undefined
03FEC144H	CAN0 message data byte 4 register 02	C0MDATA402			√		Undefined
03FEC145H	CAN0 message data byte 5 register 02	C0MDATA502			√		Undefined
03FEC146H	CAN0 message data byte 67 register 02	C0MDATA6702				√	Undefined
03FEC146H	CAN0 message data byte 6 register 02	C0MDATA602			√		Undefined
03FEC147H	CAN0 message data byte 7 register 02	C0MDATA702			√		Undefined
03FEC148H	CAN0 message data length register 02	C0MDLC02			√		0000xxxxB
03FEC149H	CAN0 message configuration register 02	C0MCONF02			√		Undefined
03FEC14AH	CAN0 message identifier register 02	C0MIDL02				√	Undefined
03FEC14CH		C0MIDH02				√	Undefined
03FEC14EH	CAN0 message control register 02	C0MCTRL02				√	00x00000 000xx000B
03FEC160H	CAN0 message data byte 01 register 03	C0MDATA0103				√	Undefined
03FEC160H	CAN0 message data byte 0 register 03	C0MDATA003			√		Undefined
03FEC161H	CAN0 message data byte 1 register 03	C0MDATA103			√		Undefined
03FEC162H	CAN0 message data byte 23 register 03	C0MDATA2303				√	Undefined
03FEC162H	CAN0 message data byte 2 register 03	C0MDATA203			√		Undefined
03FEC163H	CAN0 message data byte 3 register 03	C0MDATA303			√		Undefined
03FEC164H	CAN0 message data byte 45 register 03	C0MDATA4503				√	Undefined
03FEC164H	CAN0 message data byte 4 register 03	C0MDATA403			√		Undefined
03FEC165H	CAN0 message data byte 5 register 03	C0MDATA503			√		Undefined
03FEC166H	CAN0 message data byte 67 register 03	C0MDATA6703				√	Undefined
03FEC166H	CAN0 message data byte 6 register 03	C0MDATA603			√		Undefined
03FEC167H	CAN0 message data byte 7 register 03	C0MDATA703			√		Undefined
03FEC168H	CAN0 message data length register 03	C0MDLC03			√		0000xxxxB
03FEC169H	CAN0 message configuration register 03	C0MCONF03			√		Undefined
03FEC16AH	CAN0 message identifier register 03	C0MIDL03				√	Undefined
03FEC16CH		C0MIDH03				√	Undefined
03FEC16EH	CAN0 message control register 03	C0MCTRL03				√	00x00000 000xx000B

Table 17-16. Register Access Types (4/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC180H	CAN0 message data byte 01 register 04	C0MDATA0104	R/W			√	Undefined
03FEC180H	CAN0 message data byte 0 register 04	C0MDATA004			√		Undefined
03FEC181H	CAN0 message data byte 1 register 04	C0MDATA104			√		Undefined
03FEC182H	CAN0 message data byte 23 register 04	C0MDATA2304				√	Undefined
03FEC182H	CAN0 message data byte 2 register 04	C0MDATA204			√		Undefined
03FEC183H	CAN0 message data byte 3 register 04	C0MDATA304			√		Undefined
03FEC184H	CAN0 message data byte 45 register 04	C0MDATA4504				√	Undefined
03FEC184H	CAN0 message data byte 4 register 04	C0MDATA404			√		Undefined
03FEC185H	CAN0 message data byte 5 register 04	C0MDATA504			√		Undefined
03FEC186H	CAN0 message data byte 67 register 04	C0MDATA6704				√	Undefined
03FEC186H	CAN0 message data byte 6 register 04	C0MDATA604			√		Undefined
03FEC187H	CAN0 message data byte 7 register 04	C0MDATA704			√		Undefined
03FEC188H	CAN0 message data length register 04	C0MDLC04			√		0000xxxxB
03FEC189H	CAN0 message configuration register 04	C0MCONF04			√		Undefined
03FEC18AH	CAN0 message identifier register 04	C0MIDL04				√	Undefined
03FEC18CH		C0MIDH04				√	Undefined
03FEC18EH	CAN0 message control register 04	C0MCTRL04				√	00x00000 000xx000B
03FEC1A0H	CAN0 message data byte 01 register 05	C0MDATA0105				√	Undefined
03FEC1A0H	CAN0 message data byte 0 register 05	C0MDATA005			√		Undefined
03FEC1A1H	CAN0 message data byte 1 register 05	C0MDATA105			√		Undefined
03FEC1A2H	CAN0 message data byte 23 register 05	C0MDATA2305				√	Undefined
03FEC1A2H	CAN0 message data byte 2 register 05	C0MDATA205			√		Undefined
03FEC1A3H	CAN0 message data byte 3 register 05	C0MDATA305			√		Undefined
03FEC1A4H	CAN0 message data byte 45 register 05	C0MDATA4505				√	Undefined
03FEC1A4H	CAN0 message data byte 4 register 05	C0MDATA405			√		Undefined
03FEC1A5H	CAN0 message data byte 5 register 05	C0MDATA505			√		Undefined
03FEC1A6H	CAN0 message data byte 67 register 05	C0MDATA6705				√	Undefined
03FEC1A6H	CAN0 message data byte 6 register 05	C0MDATA605			√		Undefined
03FEC1A7H	CAN0 message data byte 7 register 05	C0MDATA705			√		Undefined
03FEC1A8H	CAN0 message data length register 05	C0MDLC05			√		0000xxxxB
03FEC1A9H	CAN0 message configuration register 05	C0MCONF05			√		Undefined
03FEC1AAH	CAN0 message identifier register 05	C0MIDL05				√	Undefined
03FEC1ACH		C0MIDH05				√	Undefined
03FEC1AEH	CAN0 message control register 05	C0MCTRL05				√	00x00000 000xx000B

Table 17-16. Register Access Types (5/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC1C0H	CAN0 message data byte 01 register 06	C0MDATA0106	R/W			√	Undefined
03FEC1C0H	CAN0 message data byte 0 register 06	C0MDATA006			√		Undefined
03FEC1C1H	CAN0 message data byte 1 register 06	C0MDATA106			√		Undefined
03FEC1C2H	CAN0 message data byte 23 register 06	C0MDATA2306				√	Undefined
03FEC1C2H	CAN0 message data byte 2 register 06	C0MDATA206			√		Undefined
03FEC1C3H	CAN0 message data byte 3 register 06	C0MDATA306			√		Undefined
03FEC1C4H	CAN0 message data byte 45 register 06	C0MDATA4506				√	Undefined
03FEC1C4H	CAN0 message data byte 4 register 06	C0MDATA406			√		Undefined
03FEC1C5H	CAN0 message data byte 5 register 06	C0MDATA506			√		Undefined
03FEC1C6H	CAN0 message data byte 67 register 06	C0MDATA6706				√	Undefined
03FEC1C6H	CAN0 message data byte 6 register 06	C0MDATA606			√		Undefined
03FEC1C7H	CAN0 message data byte 7 register 06	C0MDATA706			√		Undefined
03FEC1C8H	CAN0 message data length register 06	C0MDLC06			√		0000xxxxB
03FEC1C9H	CAN0 message configuration register 06	C0MCONF06			√		Undefined
03FEC1CAH	CAN0 message identifier register 06	C0MIDL06				√	Undefined
03FEC1CCH		C0MIDH06				√	Undefined
03FEC1CEH	CAN0 message control register 06	C0MCTRL06				√	00x00000 000xx000B
03FEC1E0H	CAN0 message data byte 01 register 07	C0MDATA0107				√	Undefined
03FEC1E0H	CAN0 message data byte 0 register 07	C0MDATA007			√		Undefined
03FEC1E1H	CAN0 message data byte 1 register 07	C0MDATA107			√		Undefined
03FEC1E2H	CAN0 message data byte 23 register 07	C0MDATA2307				√	Undefined
03FEC1E2H	CAN0 message data byte 2 register 07	C0MDATA207			√		Undefined
03FEC1E3H	CAN0 message data byte 3 register 07	C0MDATA307			√		Undefined
03FEC1E4H	CAN0 message data byte 45 register 07	C0MDATA4507				√	Undefined
03FEC1E4H	CAN0 message data byte 4 register 07	C0MDATA407			√		Undefined
03FEC1E5H	CAN0 message data byte 5 register 07	C0MDATA507			√		Undefined
03FEC1E6H	CAN0 message data byte 67 register 07	C0MDATA6707				√	Undefined
03FEC1E6H	CAN0 message data byte 6 register 07	C0MDATA607			√		Undefined
03FEC1E7H	CAN0 message data byte 7 register 07	C0MDATA707			√		Undefined
03FEC1E8H	CAN0 message data length register 07	C0MDLC07			√		0000xxxxB
03FEC1E9H	CAN0 message configuration register 07	C0MCONF07			√		Undefined
03FEC1EAH	CAN0 message identifier register 07	C0MIDL07				√	Undefined
03FEC1ECH		C0MIDH07				√	Undefined
03FEC1EEH	CAN0 message control register 07	C0MCTRL07				√	00x00000 000xx000B

Table 17-16. Register Access Types (6/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC200H	CAN0 message data byte 01 register 08	C0MDATA0108	R/W			√	Undefined
03FEC200H	CAN0 message data byte 0 register 08	C0MDATA008			√		Undefined
03FEC201H	CAN0 message data byte 1 register 08	C0MDATA108			√		Undefined
03FEC202H	CAN0 message data byte 23 register 08	C0MDATA2308				√	Undefined
03FEC202H	CAN0 message data byte 2 register 08	C0MDATA208			√		Undefined
03FEC203H	CAN0 message data byte 3 register 08	C0MDATA308			√		Undefined
03FEC204H	CAN0 message data byte 45 register 08	C0MDATA4508				√	Undefined
03FEC204H	CAN0 message data byte 4 register 08	C0MDATA408			√		Undefined
03FEC205H	CAN0 message data byte 5 register 08	C0MDATA508			√		Undefined
03FEC206H	CAN0 message data byte 67 register 08	C0MDATA6708				√	Undefined
03FEC206H	CAN0 message data byte 6 register 08	C0MDATA608			√		Undefined
03FEC207H	CAN0 message data byte 7 register 08	C0MDATA708			√		Undefined
03FEC208H	CAN0 message data length register 08	C0MDLC08			√		0000xxxxB
03FEC209H	CAN0 message configuration register 08	C0MCONF08			√		Undefined
03FEC20AH	CAN0 message identifier register 08	C0MIDL08				√	Undefined
03FEC20CH		C0MIDH08				√	Undefined
03FEC20EH	CAN0 message control register 08	C0MCTRL08				√	00x00000 000xx000B
03FEC220H	CAN0 message data byte 01 register 09	C0MDATA0109				√	Undefined
03FEC220H	CAN0 message data byte 0 register 09	C0MDATA009			√		Undefined
03FEC221H	CAN0 message data byte 1 register 09	C0MDATA109			√		Undefined
03FEC222H	CAN0 message data byte 23 register 09	C0MDATA2309				√	Undefined
03FEC222H	CAN0 message data byte 2 register 09	C0MDATA209			√		Undefined
03FEC223H	CAN0 message data byte 3 register 09	C0MDATA309			√		Undefined
03FEC224H	CAN0 message data byte 45 register 09	C0MDATA4509				√	Undefined
03FEC224H	CAN0 message data byte 4 register 09	C0MDATA409			√		Undefined
03FEC225H	CAN0 message data byte 5 register 09	C0MDATA509			√		Undefined
03FEC226H	CAN0 message data byte 67 register 09	C0MDATA6709				√	Undefined
03FEC226H	CAN0 message data byte 6 register 09	C0MDATA609			√		Undefined
03FEC227H	CAN0 message data byte 7 register 09	C0MDATA709			√		Undefined
03FEC228H	CAN0 message data length register 09	C0MDLC09			√		0000xxxxB
03FEC229H	CAN0 message configuration register 09	C0MCONF09			√		Undefined
03FEC22AH	CAN0 message identifier register 09	C0MIDL09				√	Undefined
03FEC22CH		C0MIDH09				√	Undefined
03FEC22EH	CAN0 message control register 09	C0MCTRL09				√	00x00000 000xx000B

Table 17-16. Register Access Types (7/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC240H	CAN0 message data byte 01 register 10	C0MDATA0110	R/W			√	Undefined
03FEC240H	CAN0 message data byte 0 register 10	C0MDATA010			√		Undefined
03FEC241H	CAN0 message data byte 1 register 10	C0MDATA110			√		Undefined
03FEC242H	CAN0 message data byte 23 register 10	C0MDATA2310				√	Undefined
03FEC242H	CAN0 message data byte 2 register 10	C0MDATA210			√		Undefined
03FEC243H	CAN0 message data byte 3 register 10	C0MDATA310			√		Undefined
03FEC244H	CAN0 message data byte 45 register 10	C0MDATA4510				√	Undefined
03FEC244H	CAN0 message data byte 4 register 10	C0MDATA410			√		Undefined
03FEC245H	CAN0 message data byte 5 register 10	C0MDATA510			√		Undefined
03FEC246H	CAN0 message data byte 67 register 10	C0MDATA6710				√	Undefined
03FEC246H	CAN0 message data byte 6 register 10	C0MDATA610			√		Undefined
03FEC247H	CAN0 message data byte 7 register 10	C0MDATA710			√		Undefined
03FEC248H	CAN0 message data length register 10	C0MDLC10			√		0000xxxxB
03FEC249H	CAN0 message configuration register 10	C0MCONF10			√		Undefined
03FEC24AH	CAN0 message identifier register 10	C0MIDL10				√	Undefined
03FEC24CH		C0MIDH10				√	Undefined
03FEC24EH	CAN0 message control register 10	C0MCTRL10				√	00x00000 000xx000B
03FEC260H	CAN0 message data byte 01 register 11	C0MDATA0111				√	Undefined
03FEC260H	CAN0 message data byte 0 register 11	C0MDATA011			√		Undefined
03FEC261H	CAN0 message data byte 1 register 11	C0MDATA111			√		Undefined
03FEC262H	CAN0 message data byte 23 register 11	C0MDATA2311				√	Undefined
03FEC262H	CAN0 message data byte 2 register 11	C0MDATA211			√		Undefined
03FEC263H	CAN0 message data byte 3 register 11	C0MDATA311			√		Undefined
03FEC264H	CAN0 message data byte 45 register 11	C0MDATA4511				√	Undefined
03FEC264H	CAN0 message data byte 4 register 11	C0MDATA411			√		Undefined
03FEC265H	CAN0 message data byte 5 register 11	C0MDATA511			√		Undefined
03FEC266H	CAN0 message data byte 67 register 11	C0MDATA6711				√	Undefined
03FEC266H	CAN0 message data byte 6 register 11	C0MDATA611			√		Undefined
03FEC267H	CAN0 message data byte 7 register 11	C0MDATA711			√		Undefined
03FEC268H	CAN0 message data length register 11	C0MDLC11			√		0000xxxxB
03FEC269H	CAN0 message configuration register 11	C0MCONF11			√		Undefined
03FEC26AH	CAN0 message identifier register 11	C0MIDL11				√	Undefined
03FEC26CH		C0MIDH11				√	Undefined
03FEC26EH	CAN0 message control register 11	C0MCTRL11				√	00x00000 000xx000B

Table 17-16. Register Access Types (8/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC280H	CAN0 message data byte 01 register 12	C0MDATA0112	R/W			√	Undefined
03FEC280H	CAN0 message data byte 0 register 12	C0MDATA012			√		Undefined
03FEC281H	CAN0 message data byte 1 register 12	C0MDATA112			√		Undefined
03FEC282H	CAN0 message data byte 23 register 12	C0MDATA2312				√	Undefined
03FEC282H	CAN0 message data byte 2 register 12	C0MDATA212			√		Undefined
03FEC283H	CAN0 message data byte 3 register 12	C0MDATA312			√		Undefined
03FEC284H	CAN0 message data byte 45 register 12	C0MDATA4512				√	Undefined
03FEC284H	CAN0 message data byte 4 register 12	C0MDATA412			√		Undefined
03FEC285H	CAN0 message data byte 5 register 12	C0MDATA512			√		Undefined
03FEC286H	CAN0 message data byte 67 register 12	C0MDATA6712				√	Undefined
03FEC286H	CAN0 message data byte 6 register 12	C0MDATA612			√		Undefined
03FEC287H	CAN0 message data byte 7 register 12	C0MDATA712			√		Undefined
03FEC288H	CAN0 message data length register 12	C0MDLC12			√		0000xxxxB
03FEC289H	CAN0 message configuration register 12	C0MCONF12			√		Undefined
03FEC28AH	CAN0 message identifier register 12	C0MIDL12				√	Undefined
03FEC28CH		C0MIDH12				√	Undefined
03FEC28EH	CAN0 message control register 12	C0MCTRL12				√	00x00000 000xx000B
03FEC2A0H	CAN0 message data byte 01 register 13	C0MDATA0113				√	Undefined
03FEC2A0H	CAN0 message data byte 0 register 13	C0MDATA013			√		Undefined
03FEC2A1H	CAN0 message data byte 1 register 13	C0MDATA113			√		Undefined
03FEC2A2H	CAN0 message data byte 23 register 13	C0MDATA2313				√	Undefined
03FEC2A2H	CAN0 message data byte 2 register 13	C0MDATA213			√		Undefined
03FEC2A3H	CAN0 message data byte 3 register 13	C0MDATA313			√		Undefined
03FEC2A4H	CAN0 message data byte 45 register 13	C0MDATA4513				√	Undefined
03FEC2A4H	CAN0 message data byte 4 register 13	C0MDATA413			√		Undefined
03FEC2A5H	CAN0 message data byte 5 register 13	C0MDATA513			√		Undefined
03FEC2A6H	CAN0 message data byte 67 register 13	C0MDATA6713				√	Undefined
03FEC2A6H	CAN0 message data byte 6 register 13	C0MDATA613			√		Undefined
03FEC2A7H	CAN0 message data byte 7 register 13	C0MDATA713			√		Undefined
03FEC2A8H	CAN0 message data length register 13	C0MDLC13			√		0000xxxxB
03FEC2A9H	CAN0 message configuration register 13	C0MCONF13			√		Undefined
03FEC2AAH	CAN0 message identifier register 13	C0MIDL13				√	Undefined
03FEC2ACH		C0MIDH13				√	Undefined
03FEC2AEH	CAN0 message control register 13	C0MCTRL13				√	00x00000 000xx000B

Table 17-16. Register Access Types (9/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC2C0H	CAN0 message data byte 01 register 14	C0MDATA0114	R/W			√	Undefined
03FEC2C0H	CAN0 message data byte 0 register 14	C0MDATA014			√		Undefined
03FEC2C1H	CAN0 message data byte 1 register 14	C0MDATA114			√		Undefined
03FEC2C2H	CAN0 message data byte 23 register 14	C0MDATA2314				√	Undefined
03FEC2C2H	CAN0 message data byte 2 register 14	C0MDATA214			√		Undefined
03FEC2C3H	CAN0 message data byte 3 register 14	C0MDATA314			√		Undefined
03FEC2C4H	CAN0 message data byte 45 register 14	C0MDATA4514				√	Undefined
03FEC2C4H	CAN0 message data byte 4 register 14	C0MDATA414			√		Undefined
03FEC2C5H	CAN0 message data byte 5 register 14	C0MDATA514			√		Undefined
03FEC2C6H	CAN0 message data byte 67 register 14	C0MDATA6714				√	Undefined
03FEC2C6H	CAN0 message data byte 6 register 14	C0MDATA614			√		Undefined
03FEC2C7H	CAN0 message data byte 7 register 14	C0MDATA714			√		Undefined
03FEC2C8H	CAN0 message data length register 14	C0MDLC14			√		0000xxxxB
03FEC2C9H	CAN0 message configuration register 14	C0MCONF14			√		Undefined
03FEC2CAH	CAN0 message identifier register 14	C0MIDL14				√	Undefined
03FEC2CCH		C0MIDH14				√	Undefined
03FEC2CEH	CAN0 message control register 14	C0MCTRL14				√	00x00000 000xx000B
03FEC2E0H	CAN0 message data byte 01 register 15	C0MDATA0115				√	Undefined
03FEC2E0H	CAN0 message data byte 0 register 15	C0MDATA015			√		Undefined
03FEC2E1H	CAN0 message data byte 1 register 15	C0MDATA115			√		Undefined
03FEC2E2H	CAN0 message data byte 23 register 15	C0MDATA2315				√	Undefined
03FEC2E2H	CAN0 message data byte 2 register 15	C0MDATA215			√		Undefined
03FEC2E3H	CAN0 message data byte 3 register 15	C0MDATA315			√		Undefined
03FEC2E4H	CAN0 message data byte 45 register 15	C0MDATA4515				√	Undefined
03FEC2E4H	CAN0 message data byte 4 register 15	C0MDATA415			√		Undefined
03FEC2E5H	CAN0 message data byte 5 register 15	C0MDATA515			√		Undefined
03FEC2E6H	CAN0 message data byte 67 register 15	C0MDATA6715				√	Undefined
03FEC2E6H	CAN0 message data byte 6 register 15	C0MDATA615			√		Undefined
03FEC2E7H	CAN0 message data byte 7 register 15	C0MDATA715			√		Undefined
03FEC2E8H	CAN0 message data length register 15	C0MDLC15			√		0000xxxxB
03FEC2E9H	CAN0 message configuration register 15	C0MCONF15			√		Undefined
03FEC2EAH	CAN0 message identifier register 15	C0MIDL15				√	Undefined
03FEC2ECH		C0MIDH15				√	Undefined
03FEC2EEH	CAN0 message control register 15	C0MCTRL15				√	00x00000 000xx000B

Table 17-16. Register Access Types (10/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC300H	CAN0 message data byte 01 register 16	C0MDATA0116	R/W			√	Undefined
03FEC300H	CAN0 message data byte 0 register 16	C0MDATA016			√		Undefined
03FEC301H	CAN0 message data byte 1 register 16	C0MDATA116			√		Undefined
03FEC302H	CAN0 message data byte 23 register 16	C0MDATA2316				√	Undefined
03FEC302H	CAN0 message data byte 2 register 16	C0MDATA216			√		Undefined
03FEC303H	CAN0 message data byte 3 register 16	C0MDATA316			√		Undefined
03FEC304H	CAN0 message data byte 45 register 16	C0MDATA4516				√	Undefined
03FEC304H	CAN0 message data byte 4 register 16	C0MDATA416			√		Undefined
03FEC305H	CAN0 message data byte 5 register 16	C0MDATA516			√		Undefined
03FEC306H	CAN0 message data byte 67 register 16	C0MDATA6716				√	Undefined
03FEC306H	CAN0 message data byte 6 register 16	C0MDATA616			√		Undefined
03FEC307H	CAN0 message data byte 7 register 16	C0MDATA716			√		Undefined
03FEC308H	CAN0 message data length register 16	C0MDLC16			√		0000xxxxB
03FEC309H	CAN0 message configuration register 16	C0MCONF16			√		Undefined
03FEC30AH	CAN0 message identifier register 16	C0MIDL16				√	Undefined
03FEC30CH		C0MIDH16				√	Undefined
03FEC30EH	CAN0 message control register 16	C0MCTRL16				√	00x00000 000xx000B
03FEC320H	CAN0 message data byte 01 register 17	C0MDATA0117				√	Undefined
03FEC320H	CAN0 message data byte 0 register 17	C0MDATA017			√		Undefined
03FEC321H	CAN0 message data byte 1 register 17	C0MDATA117			√		Undefined
03FEC322H	CAN0 message data byte 23 register 17	C0MDATA2317				√	Undefined
03FEC322H	CAN0 message data byte 2 register 17	C0MDATA217			√		Undefined
03FEC323H	CAN0 message data byte 3 register 17	C0MDATA317			√		Undefined
03FEC324H	CAN0 message data byte 45 register 17	C0MDATA4517				√	Undefined
03FEC324H	CAN0 message data byte 4 register 17	C0MDATA417			√		Undefined
03FEC325H	CAN0 message data byte 5 register 17	C0MDATA517			√		Undefined
03FEC326H	CAN0 message data byte 67 register 17	C0MDATA6717				√	Undefined
03FEC326H	CAN0 message data byte 6 register 17	C0MDATA617			√		Undefined
03FEC327H	CAN0 message data byte 7 register 17	C0MDATA717			√		Undefined
03FEC328H	CAN0 message data length register 17	C0MDLC17			√		0000xxxxB
03FEC329H	CAN0 message configuration register 17	C0MCONF17			√		Undefined
03FEC32AH	CAN0 message identifier register 17	C0MIDL17				√	Undefined
03FEC32CH		C0MIDH17				√	Undefined
03FEC32EH	CAN0 message control register 17	C0MCTRL17				√	00x00000 000xx000B

Table 17-16. Register Access Types (11/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC340H	CAN0 message data byte 01 register 18	C0MDATA0118	R/W			√	Undefined
03FEC340H	CAN0 message data byte 0 register 18	C0MDATA018			√		Undefined
03FEC341H	CAN0 message data byte 1 register 18	C0MDATA118			√		Undefined
03FEC342H	CAN0 message data byte 23 register 18	C0MDATA2318				√	Undefined
03FEC342H	CAN0 message data byte 2 register 18	C0MDATA218			√		Undefined
03FEC343H	CAN0 message data byte 3 register 18	C0MDATA318			√		Undefined
03FEC344H	CAN0 message data byte 45 register 18	C0MDATA4518				√	Undefined
03FEC344H	CAN0 message data byte 4 register 18	C0MDATA418			√		Undefined
03FEC345H	CAN0 message data byte 5 register 18	C0MDATA518			√		Undefined
03FEC346H	CAN0 message data byte 67 register 18	C0MDATA6718				√	Undefined
03FEC346H	CAN0 message data byte 6 register 18	C0MDATA618			√		Undefined
03FEC347H	CAN0 message data byte 7 register 18	C0MDATA718			√		Undefined
03FEC348H	CAN0 message data length register 18	C0MDLC18			√		0000xxxxB
03FEC349H	CAN0 message configuration register 18	C0MCONF18			√		Undefined
03FEC34AH	CAN0 message identifier register 18	C0MIDL18				√	Undefined
03FEC34CH		C0MIDH18				√	Undefined
03FEC34EH	CAN0 message control register 18	C0MCTRL18				√	00x00000 000xx000B
03FEC360H	CAN0 message data byte 01 register 19	C0MDATA0119				√	Undefined
03FEC360H	CAN0 message data byte 0 register 19	C0MDATA019			√		Undefined
03FEC361H	CAN0 message data byte 1 register 19	C0MDATA119			√		Undefined
03FEC362H	CAN0 message data byte 23 register 19	C0MDATA2319				√	Undefined
03FEC362H	CAN0 message data byte 2 register 19	C0MDATA219			√		Undefined
03FEC363H	CAN0 message data byte 3 register 19	C0MDATA319			√		Undefined
03FEC364H	CAN0 message data byte 45 register 19	C0MDATA4519				√	Undefined
03FEC364H	CAN0 message data byte 4 register 19	C0MDATA419			√		Undefined
03FEC365H	CAN0 message data byte 5 register 19	C0MDATA519			√		Undefined
03FEC366H	CAN0 message data byte 67 register 19	C0MDATA6719				√	Undefined
03FEC366H	CAN0 message data byte 6 register 19	C0MDATA619			√		Undefined
03FEC367H	CAN0 message data byte 7 register 19	C0MDATA719			√		Undefined
03FEC368H	CAN0 message data length register 19	C0MDLC19			√		0000xxxxB
03FEC369H	CAN0 message configuration register 19	C0MCONF19			√		Undefined
03FEC36AH	CAN0 message identifier register 19	C0MIDL19				√	Undefined
03FEC36CH		C0MIDH19				√	Undefined
03FEC36EH	CAN0 message control register 19	C0MCTRL19				√	00x00000 000xx000B

Table 17-16. Register Access Types (12/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC380H	CAN0 message data byte 01 register 20	C0MDATA0120	R/W			√	Undefined
03FEC380H	CAN0 message data byte 0 register 20	C0MDATA020			√		Undefined
03FEC381H	CAN0 message data byte 1 register 20	C0MDATA120			√		Undefined
03FEC382H	CAN0 message data byte 23 register 20	C0MDATA2320				√	Undefined
03FEC382H	CAN0 message data byte 2 register 20	C0MDATA220			√		Undefined
03FEC383H	CAN0 message data byte 3 register 20	C0MDATA320			√		Undefined
03FEC384H	CAN0 message data byte 45 register 20	C0MDATA4520				√	Undefined
03FEC384H	CAN0 message data byte 4 register 20	C0MDATA420			√		Undefined
03FEC385H	CAN0 message data byte 5 register 20	C0MDATA520			√		Undefined
03FEC386H	CAN0 message data byte 67 register 20	C0MDATA6720				√	Undefined
03FEC386H	CAN0 message data byte 6 register 20	C0MDATA620			√		Undefined
03FEC387H	CAN0 message data byte 7 register 20	C0MDATA720			√		Undefined
03FEC388H	CAN0 message data length register 20	C0MDLC20			√		0000xxxxB
03FEC389H	CAN0 message configuration register 20	C0MCONF20			√		Undefined
03FEC38AH	CAN0 message identifier register 20	C0MIDL20				√	Undefined
03FEC38CH		C0MIDH20				√	Undefined
03FEC38EH	CAN0 message control register 20	C0MCTRL20				√	00x00000 000xx000B
03FEC3A0H	CAN0 message data byte 01 register 21	C0MDATA0121				√	Undefined
03FEC3A0H	CAN0 message data byte 0 register 21	C0MDATA021			√		Undefined
03FEC3A1H	CAN0 message data byte 1 register 21	C0MDATA121			√		Undefined
03FEC3A2H	CAN0 message data byte 23 register 21	C0MDATA2321				√	Undefined
03FEC3A2H	CAN0 message data byte 2 register 21	C0MDATA221			√		Undefined
03FEC3A3H	CAN0 message data byte 3 register 21	C0MDATA321			√		Undefined
03FEC3A4H	CAN0 message data byte 45 register 21	C0MDATA4521				√	Undefined
03FEC3A4H	CAN0 message data byte 4 register 21	C0MDATA421			√		Undefined
03FEC3A5H	CAN0 message data byte 5 register 21	C0MDATA521			√		Undefined
03FEC3A6H	CAN0 message data byte 67 register 21	C0MDATA6721				√	Undefined
03FEC3A6H	CAN0 message data byte 6 register 21	C0MDATA621			√		Undefined
03FEC3A7H	CAN0 message data byte 7 register 21	C0MDATA721			√		Undefined
03FEC3A8H	CAN0 message data length register 21	C0MDLC21			√		0000xxxxB
03FEC3A9H	CAN0 message configuration register 21	C0MCONF21			√		Undefined
03FEC3AAH	CAN0 message identifier register 21	C0MIDL21				√	Undefined
03FEC3ACH		C0MIDH21				√	Undefined
03FEC3AEH	CAN0 message control register 21	C0MCTRL21				√	00x00000 000xx000B

Table 17-16. Register Access Types (13/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC3C0H	CAN0 message data byte 01 register 22	C0MDATA0122	R/W			√	Undefined
03FEC3C0H	CAN0 message data byte 0 register 22	C0MDATA022			√		Undefined
03FEC3C1H	CAN0 message data byte 1 register 22	C0MDATA122			√		Undefined
03FEC3C2H	CAN0 message data byte 23 register 22	C0MDATA2322				√	Undefined
03FEC3C2H	CAN0 message data byte 2 register 22	C0MDATA222			√		Undefined
03FEC3C3H	CAN0 message data byte 3 register 22	C0MDATA322			√		Undefined
03FEC3C4H	CAN0 message data byte 45 register 22	C0MDATA4522				√	Undefined
03FEC3C4H	CAN0 message data byte 4 register 22	C0MDATA422			√		Undefined
03FEC3C5H	CAN0 message data byte 5 register 22	C0MDATA522			√		Undefined
03FEC3C6H	CAN0 message data byte 67 register 22	C0MDATA6722				√	Undefined
03FEC3C6H	CAN0 message data byte 6 register 22	C0MDATA622			√		Undefined
03FEC3C7H	CAN0 message data byte 7 register 22	C0MDATA722			√		Undefined
03FEC3C8H	CAN0 message data length register 22	C0MDLC22			√		0000xxxxB
03FEC3C9H	CAN0 message configuration register 22	C0MCONF22			√		Undefined
03FEC3CAH	CAN0 message identifier register 22	C0MIDL22				√	Undefined
03FEC3CCH		C0MIDH22				√	Undefined
03FEC3CEH	CAN0 message control register 22	C0MCTRL22				√	00x00000 000xx000B
03FEC3E0H	CAN0 message data byte 01 register 23	C0MDATA0123				√	Undefined
03FEC3E0H	CAN0 message data byte 0 register 23	C0MDATA023			√		Undefined
03FEC3E1H	CAN0 message data byte 1 register 23	C0MDATA123			√		Undefined
03FEC3E2H	CAN0 message data byte 23 register 23	C0MDATA2323				√	Undefined
03FEC3E2H	CAN0 message data byte 2 register 23	C0MDATA223			√		Undefined
03FEC3E3H	CAN0 message data byte 3 register 23	C0MDATA323			√		Undefined
03FEC3E4H	CAN0 message data byte 45 register 23	C0MDATA4523				√	Undefined
03FEC3E4H	CAN0 message data byte 4 register 23	C0MDATA423			√		Undefined
03FEC3E5H	CAN0 message data byte 5 register 23	C0MDATA523			√		Undefined
03FEC3E6H	CAN0 message data byte 67 register 23	C0MDATA6723				√	Undefined
03FEC3E6H	CAN0 message data byte 6 register 23	C0MDATA623			√		Undefined
03FEC3E7H	CAN0 message data byte 7 register 23	C0MDATA723			√		Undefined
03FEC3E8H	CAN0 message data length register 23	C0MDLC23			√		0000xxxxB
03FEC3E9H	CAN0 message configuration register 23	C0MCONF23			√		Undefined
03FEC3EAH	CAN0 message identifier register 23	C0MIDL23				√	Undefined
03FEC3ECH		C0MIDH23				√	Undefined
03FEC3EEH	CAN0 message control register 23	C0MCTRL23				√	00x00000 000xx000B

Table 17-16. Register Access Types (14/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC400H	CAN0 message data byte 01 register 24	C0MDATA0124	R/W			√	Undefined
03FEC400H	CAN0 message data byte 0 register 24	C0MDATA024			√		Undefined
03FEC401H	CAN0 message data byte 1 register 24	C0MDATA124			√		Undefined
03FEC402H	CAN0 message data byte 23 register 24	C0MDATA2324				√	Undefined
03FEC402H	CAN0 message data byte 2 register 24	C0MDATA224			√		Undefined
03FEC403H	CAN0 message data byte 3 register 24	C0MDATA324			√		Undefined
03FEC404H	CAN0 message data byte 45 register 24	C0MDATA4524				√	Undefined
03FEC404H	CAN0 message data byte 4 register 24	C0MDATA424			√		Undefined
03FEC405H	CAN0 message data byte 5 register 24	C0MDATA524			√		Undefined
03FEC406H	CAN0 message data byte 67 register 24	C0MDATA6724				√	Undefined
03FEC406H	CAN0 message data byte 6 register 24	C0MDATA624			√		Undefined
03FEC407H	CAN0 message data byte 7 register 24	C0MDATA724			√		Undefined
03FEC408H	CAN0 message data length register 24	C0MDLC24			√		0000xxxxB
03FEC409H	CAN0 message configuration register 24	C0MCONF24			√		Undefined
03FEC40AH	CAN0 message identifier register 24	C0MIDL24				√	Undefined
03FEC40CH		C0MIDH24				√	Undefined
03FEC40EH	CAN0 message control register 24	C0MCTRL24				√	00x00000 000xx000B
03FEC420H	CAN0 message data byte 01 register 25	C0MDATA0125				√	Undefined
03FEC420H	CAN0 message data byte 0 register 25	C0MDATA025			√		Undefined
03FEC421H	CAN0 message data byte 1 register 25	C0MDATA125			√		Undefined
03FEC422H	CAN0 message data byte 23 register 25	C0MDATA2325				√	Undefined
03FEC422H	CAN0 message data byte 2 register 25	C0MDATA225			√		Undefined
03FEC423H	CAN0 message data byte 3 register 25	C0MDATA325			√		Undefined
03FEC424H	CAN0 message data byte 45 register 25	C0MDATA4525				√	Undefined
03FEC424H	CAN0 message data byte 4 register 25	C0MDATA425			√		Undefined
03FEC425H	CAN0 message data byte 5 register 25	C0MDATA525			√		Undefined
03FEC426H	CAN0 message data byte 67 register 25	C0MDATA6725				√	Undefined
03FEC426H	CAN0 message data byte 6 register 25	C0MDATA625			√		Undefined
03FEC427H	CAN0 message data byte 7 register 25	C0MDATA725			√		Undefined
03FEC428H	CAN0 message data length register 25	C0MDLC25			√		0000xxxxB
03FEC429H	CAN0 message configuration register 25	C0MCONF25			√		Undefined
03FEC42AH	CAN0 message identifier register 25	C0MIDL25				√	Undefined
03FEC42CH		C0MIDH25				√	Undefined
03FEC42EH	CAN0 message control register 25	C0MCTRL25				√	00x00000 000xx000B

Table 17-16. Register Access Types (15/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC440H	CAN0 message data byte 01 register 26	C0MDATA0126	R/W			√	Undefined
03FEC440H	CAN0 message data byte 0 register 26	C0MDATA026			√		Undefined
03FEC441H	CAN0 message data byte 1 register 26	C0MDATA126			√		Undefined
03FEC442H	CAN0 message data byte 23 register 26	C0MDATA2326				√	Undefined
03FEC442H	CAN0 message data byte 2 register 26	C0MDATA226			√		Undefined
03FEC443H	CAN0 message data byte 3 register 26	C0MDATA326			√		Undefined
03FEC444H	CAN0 message data byte 45 register 26	C0MDATA4526				√	Undefined
03FEC444H	CAN0 message data byte 4 register 26	C0MDATA426			√		Undefined
03FEC445H	CAN0 message data byte 5 register 26	C0MDATA526			√		Undefined
03FEC446H	CAN0 message data byte 67 register 26	C0MDATA6726				√	Undefined
03FEC446H	CAN0 message data byte 6 register 26	C0MDATA626			√		Undefined
03FEC447H	CAN0 message data byte 7 register 26	C0MDATA726			√		Undefined
03FEC448H	CAN0 message data length register 26	C0MDLC26			√		0000xxxxB
03FEC449H	CAN0 message configuration register 26	C0MCONF26			√		Undefined
03FEC44AH	CAN0 message identifier register 26	C0MIDL26				√	Undefined
03FEC44CH		C0MIDH26				√	Undefined
03FEC44EH	CAN0 message control register 26	C0MCTRL26				√	00x00000 000xx000B
03FEC460H	CAN0 message data byte 01 register 27	C0MDATA0127				√	Undefined
03FEC460H	CAN0 message data byte 0 register 27	C0MDATA027			√		Undefined
03FEC461H	CAN0 message data byte 1 register 27	C0MDATA127			√		Undefined
03FEC462H	CAN0 message data byte 23 register 27	C0MDATA2327				√	Undefined
03FEC462H	CAN0 message data byte 2 register 27	C0MDATA227			√		Undefined
03FEC463H	CAN0 message data byte 3 register 27	C0MDATA327			√		Undefined
03FEC464H	CAN0 message data byte 45 register 27	C0MDATA4527				√	Undefined
03FEC464H	CAN0 message data byte 4 register 27	C0MDATA427			√		Undefined
03FEC465H	CAN0 message data byte 5 register 27	C0MDATA527			√		Undefined
03FEC466H	CAN0 message data byte 67 register 27	C0MDATA6727				√	Undefined
03FEC466H	CAN0 message data byte 6 register 27	C0MDATA627			√		Undefined
03FEC467H	CAN0 message data byte 7 register 27	C0MDATA727			√		Undefined
03FEC468H	CAN0 message data length register 27	C0MDLC27			√		0000xxxxB
03FEC469H	CAN0 message configuration register 27	C0MCONF27			√		Undefined
03FEC46AH	CAN0 message identifier register 27	C0MIDL27				√	Undefined
03FEC46CH		C0MIDH27				√	Undefined
03FEC46EH	CAN0 message control register 27	C0MCTRL27				√	00x00000 000xx000B

Table 17-16. Register Access Types (16/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC480H	CAN0 message data byte 01 register 28	C0MDATA0128	R/W			√	Undefined
03FEC480H	CAN0 message data byte 0 register 28	C0MDATA028			√		Undefined
03FEC481H	CAN0 message data byte 1 register 28	C0MDATA128			√		Undefined
03FEC482H	CAN0 message data byte 23 register 28	C0MDATA2328				√	Undefined
03FEC482H	CAN0 message data byte 2 register 28	C0MDATA228			√		Undefined
03FEC483H	CAN0 message data byte 3 register 28	C0MDATA328			√		Undefined
03FEC484H	CAN0 message data byte 45 register 28	C0MDATA4528				√	Undefined
03FEC484H	CAN0 message data byte 4 register 28	C0MDATA428			√		Undefined
03FEC485H	CAN0 message data byte 5 register 28	C0MDATA528			√		Undefined
03FEC486H	CAN0 message data byte 67 register 28	C0MDATA6728				√	Undefined
03FEC486H	CAN0 message data byte 6 register 28	C0MDATA628			√		Undefined
03FEC487H	CAN0 message data byte 7 register 28	C0MDATA728			√		Undefined
03FEC488H	CAN0 message data length register 28	C0MDLC28			√		0000xxxxB
03FEC489H	CAN0 message configuration register 28	C0MCONF28			√		Undefined
03FEC48AH	CAN0 message identifier register 28	C0MIDL28				√	Undefined
03FEC48CH		C0MIDH28				√	Undefined
03FEC48EH	CAN0 message control register 28	C0MCTRL28				√	00x00000 000xx000B
03FEC4A0H	CAN0 message data byte 01 register 29	C0MDATA0129				√	Undefined
03FEC4A0H	CAN0 message data byte 0 register 29	C0MDATA029			√		Undefined
03FEC4A1H	CAN0 message data byte 1 register 29	C0MDATA129			√		Undefined
03FEC4A2H	CAN0 message data byte 23 register 29	C0MDATA2329				√	Undefined
03FEC4A2H	CAN0 message data byte 2 register 29	C0MDATA229			√		Undefined
03FEC4A3H	CAN0 message data byte 3 register 29	C0MDATA329			√		Undefined
03FEC4A4H	CAN0 message data byte 45 register 29	C0MDATA4529				√	Undefined
03FEC4A4H	CAN0 message data byte 4 register 29	C0MDATA429			√		Undefined
03FEC4A5H	CAN0 message data byte 5 register 29	C0MDATA529			√		Undefined
03FEC4A6H	CAN0 message data byte 67 register 29	C0MDATA6729				√	Undefined
03FEC4A6H	CAN0 message data byte 6 register 29	C0MDATA629			√		Undefined
03FEC4A7H	CAN0 message data byte 7 register 29	C0MDATA729			√		Undefined
03FEC4A8H	CAN0 message data length register 29	C0MDLC29			√		0000xxxxB
03FEC4A9H	CAN0 message configuration register 29	C0MCONF29			√		Undefined
03FEC4AAH	CAN0 message identifier register 29	C0MIDL29				√	Undefined
03FEC4ACH		C0MIDH29				√	Undefined
03FEC4AEH	CAN0 message control register 29	C0MCTRL29				√	00x00000 000xx000B

Table 17-16. Register Access Types (17/17)

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
				1 Bit	8 Bits	16 Bits	
03FEC4C0H	CAN0 message data byte 01 register 30	C0MDATA0130	R/W			√	Undefined
03FEC4C0H	CAN0 message data byte 0 register 30	C0MDATA030			√		Undefined
03FEC4C1H	CAN0 message data byte 1 register 30	C0MDATA130			√		Undefined
03FEC4C2H	CAN0 message data byte 23 register 30	C0MDATA2330				√	Undefined
03FEC4C2H	CAN0 message data byte 2 register 30	C0MDATA230			√		Undefined
03FEC4C3H	CAN0 message data byte 3 register 30	C0MDATA330			√		Undefined
03FEC4C4H	CAN0 message data byte 45 register 30	C0MDATA4530				√	Undefined
03FEC4C4H	CAN0 message data byte 4 register 30	C0MDATA430			√		Undefined
03FEC4C5H	CAN0 message data byte 5 register 30	C0MDATA530			√		Undefined
03FEC4C6H	CAN0 message data byte 67 register 30	C0MDATA6730				√	Undefined
03FEC4C6H	CAN0 message data byte 6 register 30	C0MDATA630			√		Undefined
03FEC4C7H	CAN0 message data byte 7 register 30	C0MDATA730			√		Undefined
03FEC4C8H	CAN0 message data length register 30	C0MDLC30			√		0000xxxxB
03FEC4C9H	CAN0 message configuration register 30	C0MCONF30			√		Undefined
03FEC4CAH	CAN0 message identifier register 30	C0MIDL30				√	Undefined
03FEC4CCH		C0MIDH30				√	Undefined
03FEC4CEH	CAN0 message control register 30	C0MCTRL30				√	00x00000 000xx000B
03FEC4E0H	CAN0 message data byte 01 register 31	C0MDATA0131				√	Undefined
03FEC4E0H	CAN0 message data byte 0 register 31	C0MDATA031			√		Undefined
03FEC4E1H	CAN0 message data byte 1 register 31	C0MDATA131			√		Undefined
03FEC4E2H	CAN0 message data byte 23 register 31	C0MDATA2331				√	Undefined
03FEC4E2H	CAN0 message data byte 2 register 31	C0MDATA231			√		Undefined
03FEC4E3H	CAN0 message data byte 3 register 31	C0MDATA331			√		Undefined
03FEC4E4H	CAN0 message data byte 45 register 31	C0MDATA4531				√	Undefined
03FEC4E4H	CAN0 message data byte 4 register 31	C0MDATA431			√		Undefined
03FEC4E5H	CAN0 message data byte 5 register 31	C0MDATA531			√		Undefined
03FEC4E6H	CAN0 message data byte 67 register 31	C0MDATA6731				√	Undefined
03FEC4E6H	CAN0 message data byte 6 register 31	C0MDATA631			√		Undefined
03FEC4E7H	CAN0 message data byte 7 register 31	C0MDATA731			√		Undefined
03FEC4E8H	CAN0 message data length register 31	C0MDLC31			√		0000xxxxB
03FEC4E9H	CAN0 message configuration register 31	C0MCONF31			√		Undefined
03FEC4EAH	CAN0 message identifier register 31	C0MIDL31				√	Undefined
03FEC4ECH		C0MIDH31				√	Undefined
03FEC4EEH	CAN0 message control register 31	C0MCTRL31				√	00x00000 000xx000B

17.5.3 Register bit configuration

Table 17-17. CAN Global Register Bit Configuration

Address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
03FEC000H	C0GMCTRL (W)	0	0	0	0	0	0	0	Clear GOM
03FEC001H		0	0	0	0	0	0	Set EFSD	Set GOM
03FEC000H	C0GMCTRL (R)	0	0	0	0	0	0	EFSD	GOM
03FEC001H		MBON	0	0	0	0	0	0	0
03FEC002H	C0GMCS	0	0	0	0	CCP3	CCP2	CCP1	CCP0
03FEC006H	C0GMABT (W)	0	0	0	0	0	0	0	Clear ABTTRG
03FEC007H		0	0	0	0	0	0	Set ABTCLR	Set ABTTRG
03FEC006H	C0GMABT (R)	0	0	0	0	0	0	ABTCLR	ABTTRG
03FEC007H		0	0	0	0	0	0	0	0
03FEC008H	C0GMABTD	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0

Table 17-18. CAN Module Register Bit Configuration (1/2)

Address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
03FEC040H	C0MASK1L	CMID7 to CMID0							
03FEC041H		CMID15 to CMID8							
03FEC042H	C0MASK1H	CMID23 to CMID16							
03FEC043H		0	0	0	CMID28 to CMID24				
03FEC044H	C0MASK2L	CMID7 to CMID0							
03FEC045H		CMID15 to CMID8							
03FEC046H	C0MASK2H	CMID23 to CMID16							
03FEC047H		0	0	0	CMID28 to CMID24				
03FEC048H	C0MASK3L	CMID7 to CMID0							
03FEC049H		CMID15 to CMID8							
03FEC04AH	C0MASK3H	CMID23 to CMID16							
03FEC04BH		0	0	0	CMID28 to CMID24				
03FEC04CH	C0MASK4L	CMID7 to CMID0							
03FEC04DH		CMID15 to CMID8							
03FEC04EH	C0MASK4H	CMID23 to CMID16							
03FEC04FH		0	0	0	CMID28 to CMID24				
03FEC050H	C0CTRL (W)	0	Clear AL	Clear VALID	Clear PSMODE1	Clear PSMODE0	Clear OPMODE2	Clear OPMODE1	Clear OPMODE0
03FEC051H		Set CCERC	Set AL	0	Set PSMODE1	Set PSMODE0	Set OPMODE2	Set OPMODE1	Set OPMODE0
03FEC050H	C0CTRL (R)	CCERC	AL	VALID	PS MODE1	PS MODE0	OP MODE2	OP MODE1	OP MODE0
03FEC051H		0	0	0	0	0	0	RSTAT	TSTAT
03FEC052H	C0LEC (W)	0	0	0	0	0	0	0	0
03FEC052H	C0LEC (R)	0	0	0	0	0	LEC2	LEC1	LEC0
03FEC053H	C0INFO	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0
03FEC054H	C0ERC	TEC7 to TEC0							
03FEC055H		REC7 to REC0							
03FEC056H	C0IE (W)	0	0	Clear CIE5	Clear CIE4	Clear CIE3	Clear CIE2	Clear CIE1	Clear CIE0
03FEC057H		0	0	Set CIE5	Set CIE4	Set CIE3	Set CIE2	Set CIE1	Set CIE0
03FEC056H	C0IE (R)	0	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0
03FEC057H		0	0	0	0	0	0	0	0
03FEC058H	C0INTS (W)	0	0	Clear CINTS5	Clear CINTS4	Clear CINTS3	Clear CINTS2	Clear CINTS1	Clear CINTS0
03FEC059H		0	0	0	0	0	0	0	0
03FEC058H	C0INTS (R)	0	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0
03FEC059H		0	0	0	0	0	0	0	0

Table 17-18. CAN Module Register Bit Configuration (2/2)

Address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
03FEC05AH	C0BRP	TQPRS7 to TQPRS0							
03FEC05CH	C0BTR	0	0	0	0	TSEG13 to TSEG10			
03FEC05DH		0	0	SJW1, SJW0		0	TSEG22 to TSEG20		
03FEC05EH	C0LIPT	LIPT7 to LIPT0							
03FEC060H	C0RGPT (W)	0	0	0	0	0	0	0	Clear ROVF
03FEC061H		0	0	0	0	0	0	0	0
03FEC060H	C0RGPT (R)	0	0	0	0	0	0	RHPM	ROVF
03FEC061H		RGPT7 to RGPT0							
03FEC062H	C0LOPT	LOPT7 to LOPT0							
03FEC064H	C0TGPT (W)	0	0	0	0	0	0	0	Clear TOVF
03FEC065H		0	0	0	0	0	0	0	0
03FEC064H	C0TGPT (R)	0	0	0	0	0	0	THPM	TOVF
03FEC065H		TGPT7 to TGPT0							
03FEC068H to 03FEC0FFH	–	Access prohibited (reserved for future use)							

Table 17-19. Message Buffer Register Bit Configuration

Address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
03FECxx0H	C0MDATA01m	Message data (byte 0)							
03FECxx1H		Message data (byte 1)							
03FECxx0H	C0MDATA0m	Message data (byte 0)							
03FECxx1H	C0MDATA1m	Message data (byte 1)							
03FECxx2H	C0MDATA23m	Message data (byte 2)							
03FECxx3H		Message data (byte 3)							
03FECxx2H	C0MDATA2m	Message data (byte 2)							
03FECxx3H	C0MDATA3m	Message data (byte 3)							
03FECxx4H	C0MDATA45m	Message data (byte 4)							
03FECxx5H		Message data (byte 5)							
03FECxx4H	C0MDATA4m	Message data (byte 4)							
03FECxx5H	C0MDATA5m	Message data (byte 5)							
03FECxx6H	C0MDATA67m	Message data (byte 6)							
03FECxx7H		Message data (byte 7)							
03FECxx6H	C0MDATA6m	Message data (byte 6)							
03FECxx7H	C0MDATA7m	Message data (byte 7)							
03FECxx8H	C0MDLCm	0				MDLC3	MDLC2	MDLC1	MDLC0
03FECxx9H	C0MCONFm	OWS	RTR	MT2	MT1	MT0	0	0	MA0
03FECxxAH	C0MIDLm	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
03FECxxBH		ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
03FECxxCH	C0MIDHm	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
03FECxxDH		IDE	0	0	ID28	ID27	ID26	ID25	ID24
03FECxxEH	C0MCTRLm (W)	0	0	0	Clear MOW	Clear IE	Clear DN	Clear TRQ	Clear RDY
03FECxxFH		0	0	0	0	Set IE	0	Set TRQ	Set RDY
03FECxxEH	C0MCTRLm (R)	0	0	0	MOW	IE	DN	TRQ	RDY
03FECxxFH		0	0	MUC	0	0	0	0	0
03FECxx0 to 03FECxxFH	–	Access prohibited (reserved for future use)							

Remark m = 00 to 31

xx = 10, 12, 14, 16, 18, 1A, 1C, 1E, 20, 22, 24, 26, 28, 2A, 2C, 2E, 30, 32, 34, 36, 38, 3A, 3C, 3E, 40, 42, 44, 46, 48, 4A, 4C, 4E

17.6 Registers

Caution Accessing the CAN controller registers is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.

- **When the CPU operates with the subclock and the main clock oscillation is stopped**
- **When the CPU operates with the internal oscillation clock**

Remark m = 00 to 31

(1) CAN0 global control register (C0GMCTRL)

The C0GMCTRL register is used to control the operation of the CAN module.

(1/2)

After reset: 0000H R/W Address: 03FEC000H

(a) Read

	15	14	13	12	11	10	9	8
C0GMCTRL	MBON	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	EFSD	GOM

(b) Write

	15	14	13	12	11	10	9	8
C0GMCTRL	0	0	0	0	0	0	Set EFSD	Set GOM
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	Clear GOM

(a) Read

MBON	Bit enabling access to message buffer register, transmit/receive history registers
0	Write access and read access to the message buffer register and the transmit/receive history list registers is disabled.
1	Write access and read access to the message buffer register and the transmit/receive history list registers is enabled.

Cautions 1. While the MBON bit is cleared (to 0), software access to the message buffers (C0MDATA0m, C0MDATA1m, C0MDATA01m, C0MDATA2m, C0MDATA3m, C0MDATA23m, C0MDATA4m, C0MDATA5m, C0MDATA45m, C0MDATA6m, C0MDATA7m, C0MDATA67m, C0MDLCm, C0MCONFm, C0MIDLm, C0MIDHm, and C0MCTRLm), or registers related to transmit history or receive history (C0LOPT, C0TGPT, C0LIPT, and C0RGPT) is disabled.

2. This bit is read-only. Even if 1 is written to the MBON bit while it is 0, the value of the MBON bit does not change, and access to the message buffer registers, or registers related to transmit history or receive history remains disabled.

Remark When the CAN sleep mode/CAN stop mode is entered, or when the GOM bit is cleared to 0, the MBON bit is cleared to 0. When the CAN sleep mode/CAN stop mode is released, or when the GOM bit is set to 1, the MBON bit is set to 1.

(2/2)

EFSD	Bit enabling forced shut down
0	Forced shut down by GOM bit = 0 disabled.
1	Forced shut down by GOM bit = 0 enabled.

Caution To request forced shut down, clear the GOM bit to 0 immediately after the EFSD bit has been set to 1. If access to another register (including reading the C0GMCTRL register) is executed without clearing the GOM bit immediately after the EFSD bit has been set to 1, the EFSD bit is forcibly cleared to 0, and the forced shut down request is invalid.

GOM	Global operation mode bit
0	CAN module is disabled from operating.
1	CAN module is enabled to operate.

Caution The GOM bit is cleared to 0 only in the initialization mode or immediately after the EFSD bit is set to 1.

(b) Write

Set EFSD	EFSD bit setting
0	No change in EFSD bit.
1	EFSD bit is set to 1.

Set GOM	Clear GOM	GOM bit setting
0	1	GOM bit is cleared to 0.
1	0	GOM bit is set to 1.
Other than above		No change in GOM bit.

Caution Be sure to set the GOM bit and EFSD bit separately.

(2) CAN0 global clock selection register (C0GMCS)

The C0GMCS register is used to select the CAN module system clock.

After reset: 0FH R/W Address: 03FEC002H

	7	6	5	4	3	2	1	0
C0GMCS	0	0	0	0	CCP3	CCP2	CCP1	CCP0

CCP3	CCP2	CCP1	CCP1	CAN module system clock (f_{CANMOD})
0	0	0	0	f_{CAN}/1
0	0	0	1	f_{CAN}/2
0	0	1	0	f_{CAN}/3
0	0	1	1	f_{CAN}/4
0	1	0	0	f_{CAN}/5
0	1	0	1	f_{CAN}/6
0	1	1	0	f_{CAN}/7
0	1	1	1	f_{CAN}/8
1	0	0	0	f_{CAN}/9
1	0	0	1	f_{CAN}/10
1	0	1	0	f_{CAN}/11
1	0	1	1	f_{CAN}/12
1	1	0	0	f_{CAN}/13
1	1	0	1	f_{CAN}/14
1	1	1	0	f_{CAN}/15
1	1	1	1	f_{CAN}/16 (default value)

Remark f_{CAN} = Clock supplied to CAN = f_{XX}

(3) CAN0 global automatic block transmission control register (C0GMABT)

The C0GMABT register is used to control the automatic block transmission (ABT) operation.

(1/2)

After reset: 0000H R/W Address: 03FEC006H

(a) Read

	15	14	13	12	11	10	9	8
C0GMABT	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	ABTCLR	ABTTRG

(a) Write

	15	14	13	12	11	10	9	8
C0GMABT	0	0	0	0	0	0	Set ABTCLR	Set ABTTRG
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	Clear ABTTRG

Caution Before changing the normal operation mode with ABT to the initialization mode, be sure to set the C0GMABT register to the default value (0000H). After setting, confirm that the C0GMABT register is initialized to 0000H.

(a) Read

ABTCLR	Automatic block transmission engine clear status bit
0	Clearing the automatic transmission engine is completed.
1	The automatic transmission engine is being cleared.

Remarks 1. Set the ABTCLR bit to 1 while the ABTTRG bit is cleared to 0.
The operation is not guaranteed if the ABTCLR bit is set to 1 while the ABTTRG bit is set to 1.

2. When the automatic block transmission engine is cleared by setting the ABTCLR bit to 1, the ABTCLR bit is automatically cleared to 0 as soon as the requested clearing processing is complete.

ABTTRG	Automatic block transmission status bit
0	Automatic block transmission is stopped.
1	Automatic block transmission is under execution.

Cautions 1. Do not set the ABTTRG bit to 1 in the initialization mode. If the ABTTRG bit is set to 1 in the initialization mode, the operation is not guaranteed after the CAN module has entered the normal operation mode with ABT.

2. Do not set the ABTTRG bit to 1 while the C0CTRL.TSTAT bit is set to 1. Directly confirm that the TSTAT bit = 0 before setting the ABTTRG bit to 1.

(b) Write

Set ABTCLR	Automatic block transmission engine clear request bit
0	The automatic block transmission engine is in idle status or under operation.
1	Request to clear the automatic block transmission engine. After the automatic block transmission engine has been cleared, automatic block transmission is started from message buffer 0 by setting the ABTTRG bit to 1.

Set ABTTRG	Clear ABTTRG	Automatic block transmission start bit
0	1	Request to stop automatic block transmission.
1	0	Request to start automatic block transmission.
Other than above		No change in ABTTRG bit.

Caution Even if the ABTTRG bit is set (1), transmission is not immediately executed, depending on the situation such as when a message is received from another node or when a message other than the ABT message (message buffers 8 to 31) is transmitted.
Even if the ABTTRG bit is cleared (0), transmission is not terminated midway. If transmission is under execution, it is continued until completed (regardless of whether transmission is successful or fails).

(4) CAN0 global automatic block transmission delay setting register (C0GMABTD)

The C0GMABTD register is used to set the interval at which the data of the message buffer assigned to ABT is to be transmitted in the normal operation mode with ABT.

After reset: 00H R/W Address: 03FEC008H

	7	6	5	4	3	2	1	0
C0GMABTD	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0

ABTD3	ABTD2	ABTD1	ABTD0	Data frame interval during automatic block transmission (Unit: Data bit time (DBT))
0	0	0	0	0 DBT (default value)
0	0	0	1	2^5 DBT
0	0	1	0	2^6 DBT
0	0	1	1	2^7 DBT
0	1	0	0	2^8 DBT
0	1	0	1	2^9 DBT
0	1	1	0	2^{10} DBT
0	1	1	1	2^{11} DBT
1	0	0	0	2^{12} DBT
Other than above				Setting prohibited

Cautions 1. Do not change the contents of the C0GMABTD register while the ABTTRG bit is set to 1.

2. The timing at which the ABT message is actually transmitted onto the CAN bus differs depending on the status of transmission from the other station or how a request to transmit a message other than an ABT message (message buffers 8 to 31) is made.

(5) CAN0 module mask control register (C0MASKaL, C0MASKaH) (a = 1, 2, 3, or 4)

The C0MASKaL and C0MASKaH registers are used to extend the number of receivable messages in the same message buffer by masking part of the identifier (ID) of a message and invalidating the ID comparison of the masked part.

(1/2)

- CAN0 module mask 1 register (C0MASK1L, C0MASK1H)

After reset: Undefined R/W Address: C0MASK1L 03FEC040H, C0MASK1H 03FEC042H

	15	14	13	12	11	10	9	8
C0MASK1L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
	7	6	5	4	3	2	1	0
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	15	14	13	12	11	10	9	8
C0MASK1H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	7	6	5	4	3	2	1	0
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16

- CAN0 module mask 2 register (C0MASK2L, C0MASK2H)

After reset: Undefined R/W Address: C0MASK2L 03FEC044H, C0MASK2H 03FEC046H

	15	14	13	12	11	10	9	8
C0MASK2L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
	7	6	5	4	3	2	1	0
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	15	14	13	12	11	10	9	8
C0MASK2H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	7	6	5	4	3	2	1	0
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16

(2/2)

- CAN0 module mask 3 register (C0MASK3L, C0MASK3H)

After reset: Undefined R/W Address: C0MASK3L 03FEC048H, C0MASK3H 03FEC04AH

	15	14	13	12	11	10	9	8
C0MASK3L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
	7	6	5	4	3	2	1	0
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	15	14	13	12	11	10	9	8
C0MASK3H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	7	6	5	4	3	2	1	0
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16

- CAN0 module mask 4 register (C0MASK4L, C0MASK4H)

After reset: Undefined R/W Address: C0MASK4L 03FEC04CH, C0MASK4H 03FEC04EH

	15	14	13	12	11	10	9	8
C0MASK4L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
	7	6	5	4	3	2	1	0
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
	15	14	13	12	11	10	9	8
C0MASK4H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	7	6	5	4	3	2	1	0
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16

CMID28 to CMID0	Mask pattern setting of ID bit
0	The ID bits of the message buffer set by the CMID28 to CMID0 bits are compared with the ID bits of the received message frame.
1	The ID bits of the message buffer set by the CMID28 to CMID0 bits are not compared with the ID bits of the received message frame (they are masked).

Remark Masking is always defined by an ID length of 29 bits. If a mask is assigned to a message with a standard ID, the CMID17 to CMID0 bits are ignored. Therefore, only the CMID28 to CMID18 bits of the received ID are masked. The same mask can be used for both the standard and extended IDs.

(6) CAN0 module control register (C0CTRL)

The C0CTRL register is used to control the operation mode of the CAN module.

(1/4)

After reset: 0000H R/W Address: 03FEC050H

(a) Read

	15	14	13	12	11	10	9	8
C0CTRL	0	0	0	0	0	0	RSTAT	TSTAT
	7	6	5	4	3	2	1	0
	CCERC	AL	VALID	PSMODE 1	PSMODE 0	OPMODE 2	OPMODE 1	OPMODE 0

(a) Write

	15	14	13	12	11	10	9	8
C0CTRL	Set CCERC	Set AL	0	Set PSMODE 1	Set PSMODE 0	Set OPMODE 2	Set OPMODE 1	Set OPMODE 0
	7	6	5	4	3	2	1	0
	0	Clear AL	Clear VALID	Clear PSMODE 1	Clear PSMODE 0	Clear OPMODE 2	Clear OPMODE 1	Clear OPMODE 0

(a) Read

RSTAT	Reception status bit
0	Reception is stopped.
1	Reception is in progress.

Remark
- The RSTAT bit is set to 1 under the following conditions (timing)
  - The SOF bit of a receive frame is detected
  - On occurrence of arbitration loss during a transmit frame
- The RSTAT bit is cleared to 0 under the following conditions (timing)
  - When a recessive level is detected at the second bit of the interframe space
  - On transition to the initialization mode at the first bit of the interframe space

(2/4)

TSTAT	Transmission status bit
0	Transmission is stopped.
1	Transmission is in progress.

Remark
- The TSTAT bit is set to 1 under the following conditions (timing)
  - The SOF bit of a transmit frame is detected
- The TSTAT bit is cleared to 0 under the following conditions (timing)
  - During transition to bus-off status
  - On occurrence of arbitration loss in transmit frame
  - On detection of recessive level at the second bit of the interframe space
  - On transition to the initialization mode at the first bit of the interframe space

CCERC	Error counter clear bit
0	The C0ERC and C0INFO registers are not cleared in the initialization mode.
1	The C0ERC and C0INFO registers are cleared in the initialization mode.

Remarks 1. The CCERC bit is used to clear the C0ERC and C0INFO registers for re-initialization or forced recovery from the bus-off status. This bit can be set to 1 only in the initialization mode.

2. When the C0ERC and C0INFO registers have been cleared, the CCERC bit is also cleared to 0 automatically.

3. The CCERC bit can be set to 1 at the same time as a request to change the initialization mode to an operation mode is made.

4. If the CCERC bit is set to 1 immediately after the INIT mode is entered in the self test mode, the receive data may be corrupted.

AL	Bit to set operation in case of arbitration loss
0	Re-transmission is not executed in case of an arbitration loss in the single-shot mode.
1	Re-transmission is executed in case of an arbitration loss in the single-shot mode.

Remark The AL bit is valid only in the single-shot mode.

VALID	Valid receive message frame detection bit
0	A valid message frame has not been received since the VALID bit was last cleared to 0.
1	A valid message frame has been received since the VALID bit was last cleared to 0.

Remarks 1. Detection of a valid receive message frame is not dependent upon storage in the receive message buffer (data frame) or transmit message buffer (remote frame).

2. Clear the VALID bit (0) before changing the initialization mode to an operation mode.

3. If only two CAN nodes are connected to the CAN bus with one transmitting a message frame in the normal mode and the other in the receive-only mode, since no ACK is generated in the receive-only mode, the VALID bit is not set to 1 before the transmitting node enters the error passive status.

4. To clear the VALID bit, set the Clear VALID bit to 1 first and confirm that the VALID bit is cleared. If it is not cleared, perform clearing processing again.

PSMODE1	PSMODE0	Power save mode
0	0	No power save mode is selected.
0	1	CAN sleep mode
1	0	Setting prohibited
1	1	CAN stop mode

Cautions 1. Transition to and from the CAN stop mode must be made via CAN sleep mode. A request for direct transition to and from the CAN stop mode is ignored.

2. After releasing the power save mode, the C0GMCTRL.MBON flag must be checked before accessing the message buffer again.

3. A request for transition to the CAN sleep mode is held pending until it is canceled by software or until the CAN bus enters the bus idle state. The software can check transition to the CAN sleep mode by reading the PSMODE0 and PSMODE1 bits.

OPMODE2	OPMODE1	OPMODE0	Operation mode
0	0	0	No operation mode is selected (CAN module is in the initialization mode).
0	0	1	Normal operation mode
0	1	0	Normal operation mode with automatic block transmission function (normal operation mode with ABT)
0	1	1	Receive-only mode
1	0	0	Single-shot mode
1	0	1	Self-test mode
Other than above			Setting prohibited

Caution It may take time to change the mode to the initialization mode or power save mode. Therefore, be sure to check if the mode has been successfully changed, by reading the register value before executing the processing.

Remark The OPMODE0 to OPMODE2 bits are read-only in the CAN sleep mode or CAN stop mode.

(4/4)

(b) Write

Set CCERC	Setting of CCERC bit
1	CCERC bit is set to 1.
Other than above	CCERC bit is not changed.

Set AL	Clear AL	Setting of AL bit
0	1	AL bit is cleared to 0.
1	0	AL bit is set to 1.
Other than above		AL bit is not changed.

Clear VALID	Setting of VALID bit
0	VALID bit is not changed.
1	VALID bit is cleared to 0.

Set PSMODE0	Clear PSMODE0	Setting of PSMODE0 bit
0	1	PSMODE0 bit is cleared to 0.
1	0	PSMODE bit is set to 1.
Other than above		PSMODE0 bit is not changed.

Set PSMODE1	Clear PSMODE1	Setting of PSMODE1 bit
0	1	PSMODE1 bit is cleared to 0.
1	0	PSMODE1 bit is set to 1.
Other than above		PSMODE1 bit is not changed.

Set OPMODE0	Clear OPMODE0	Setting of OPMODE0 bit
0	1	OPMODE0 bit is cleared to 0.
1	0	OPMODE0 bit is set to 1.
Other than above		OPMODE0 bit is not changed.

Set OPMODE1	Clear OPMODE1	Setting of OPMODE1 bit
0	1	OPMODE1 bit is cleared to 0.
1	0	OPMODE1 bit is set to 1.
Other than above		OPMODE1 bit is not changed.

Set OPMODE2	Clear OPMODE2	Setting of OPMODE2 bit
0	1	OPMODE2 bit is cleared to 0.
1	0	OPMODE2 bit is set to 1.
Other than above		OPMODE2 bit is not changed.

(7) CAN0 module last error information register (C0LEC)

The C0LEC register provides the error information of the CAN protocol.

After reset: 00H R/W Address: 03FEC052H

	7	6	5	4	3	2	1	0
C0LEC	0	0	0	0	0	LEC2	LEC1	LEC0

Remarks 1. The contents of the C0LEC register are not cleared when the CAN module changes from an operation mode to the initialization mode.

2. If an attempt is made to write a value other than 00H to the C0LEC register by software, the access is ignored.

LEC2	LEC1	LEC0	Last CAN protocol error information
0	0	0	No error
0	0	1	Stuff error
0	1	0	Form error
0	1	1	ACK error
1	0	0	Bit error. (The CAN module tried to transmit a recessive-level bit as part of a transmit message (except the arbitration field), but the value on the CAN bus is a dominant-level bit.)
1	0	1	Bit error. (The CAN module tried to transmit a dominant-level bit as part of a transmit message, ACK bit, error frame, or overload frame, but the value on the CAN bus is a recessive-level bit.)
1	1	0	CRC error
1	1	1	Undefined

(8) CAN0 module information register (C0INFO)

The C0INFO register indicates the status of the CAN module.

After reset: 00H R Address: 03FEC053H

	7	6	5	4	3	2	1	0
C0INFO	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0

BOFF	Bus-off status bit
0	Not bus-off status (transmit error counter ≤ 255). (The value of the transmit counter is less than 256.)
1	Bus-off status (transmit error counter > 255). (The value of the transmit error counter is 256 or more.)

TECS1	TECS0	Transmission error counter status bit
0	0	The value of the transmission error counter is less than that of the warning level (< 96).
0	1	The value of the transmission error counter is in the range of the warning level (96 to 127).
1	0	Undefined
1	1	The value of the transmission error counter is in the range of the error passive or bus-off status (≥ 128).

RECS1	RECS0	Reception error counter status bit
0	0	The value of the reception error counter is less than that of the warning level (< 96).
0	1	The value of the reception error counter is in the range of the warning level (96 to 127).
1	0	Undefined
1	1	The value of the reception error counter is in the error passive range (≥ 128).

(9) CAN0 module error counter register (C0ERC)

The C0ERC register indicates the count value of the transmission/reception error counter.

After reset: 0000H R Address: 03FEC054H

	15	14	13	12	11	10	9	8
C0ERC	REPS	REC6	REC5	REC4	REC3	REC2	REC1	REC0
	7	6	5	4	3	2	1	0
	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0

REPS	Reception error passive status bit
0	The value of the reception error counter is not error passive (< 128)
1	The value of the reception error counter is in the error passive range (≥ 128)

REC6 to REC0	Reception error counter bit
0 to 127	Number of reception errors. These bits reflect the status of the reception error counter. The number of errors is defined by the CAN protocol.

Remark The REC6 to REC0 bits of the reception error counter are invalid in the reception error passive status (C0INFO.RECS1, C0INFO.RECS0 bit = 11B).

TEC7 to TEC0	Transmission error counter bit
0 to 255	Number of transmission errors. These bits reflect the status of the transmission error counter. The number of errors is defined by the CAN protocol.

Remark The TEC7 to TEC0 bits of the transmission error counter are invalid in the bus-off status (C0INFO.BOFF bit = 1).

(10) CAN0 module interrupt enable register (C0IE)

The C0IE register is used to enable or disable the interrupts of the CAN module.

(1/2)

After reset: 0000H R/W Address: 03FEC056H

(a) Read

	15	14	13	12	11	10	9	8
C0IE	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0

(b) Write

	15	14	13	12	11	10	9	8
C0IE	0	0	Set CIE5	Set CIE4	Set CIE3	Set CIE2	Set CIE1	Set CIE0
	7	6	5	4	3	2	1	0
	0	0	Clear CIE5	Clear CIE4	Clear CIE3	Clear CIE2	Clear CIE1	Clear CIE0

(a) Read

CIE5 to CIE0	CAN module interrupt enable bit
0	Output of the interrupt corresponding to interrupt status register CINTSx is disabled.
1	Output of the interrupt corresponding to interrupt status register CINTSx is enabled.

(2/2)

(b) Write

Set CIE5	Clear CIE5	Setting of CIE5 bit
0	1	CIE5 bit is cleared to 0.
1	0	CIE5 bit is set to 1.
Other than above		CIE5 bit is not changed.

Set CIE4	Clear CIE4	Setting of CIE4 bit
0	1	CIE4 bit is cleared to 0.
1	0	CIE4 bit is set to 1.
Other than above		CIE4 bit is not changed.

Set CIE3	Clear CIE3	Setting of CIE3 bit
0	1	CIE3 bit is cleared to 0.
1	0	CIE3 bit is set to 1.
Other than above		CIE3 bit is not changed.

Set CIE2	Clear CIE2	Setting of CIE2 bit
0	1	CIE2 bit is cleared to 0.
1	0	CIE2 bit is set to 1.
Other than above		CIE2 bit is not changed.

Set CIE1	Clear CIE1	Setting of CIE1 bit
0	1	CIE1 bit is cleared to 0.
1	0	CIE1 bit is set to 1.
Other than above		CIE1 bit is not changed.

Set CIE0	Clear CIE0	Setting of CIE0 bit
0	1	CIE0 bit is cleared to 0.
1	0	CIE0 bit is set to 1.
Other than above		CIE0 bit is not changed.

(11) CAN0 module interrupt status register (C0INTS)

The C0INTS register indicates the interrupt status of the CAN module.

After reset: 0000H R/W Address: 03FEC058H

(a) Read

	15	14	13	12	11	10	9	8
C0INTS	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0

(b) Write

	15	14	13	12	11	10	9	8
C0INTS	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	Clear CINTS5	Clear CINTS4	Clear CINTS3	Clear CINTS2	Clear CINTS1	Clear CINTS0

(a) Read

CINTS5 to CINTS0	CAN interrupt status bit
0	No related interrupt source event is generated.
1	A related interrupt source event is generated.

Interrupt status bit	Related interrupt source event
CINTS5	Wakeup interrupt from CAN sleep mode[Note]
CINTS4	Arbitration loss interrupt
CINTS3	CAN protocol error interrupt
CINTS2	CAN error status interrupt
CINTS1	Interrupt on completion of reception of valid message frame to message buffer m
CINTS0	Interrupt on normal completion of transmission of message frame from message buffer m

Note The CINTS5 bit is set (1) only when the CAN module is woken up from the CAN sleep mode by a CAN bus operation. The CINTS5 bit is not set (1) when the CAN sleep mode has been released by software.

(b) Write

Clear CINTS5 to CINTS0	Setting of CINTS5 to CINTS0 bits
0	CINTS5 to CINTS0 bits are not changed.
1	CINTS5 to CINTS0 bits are cleared to 0.

Caution The status bit of this register is not automatically cleared. Clear (0) it by software if each status must be checked in the interrupt servicing.

(12) CAN0 module bit rate prescaler register (C0BRP)

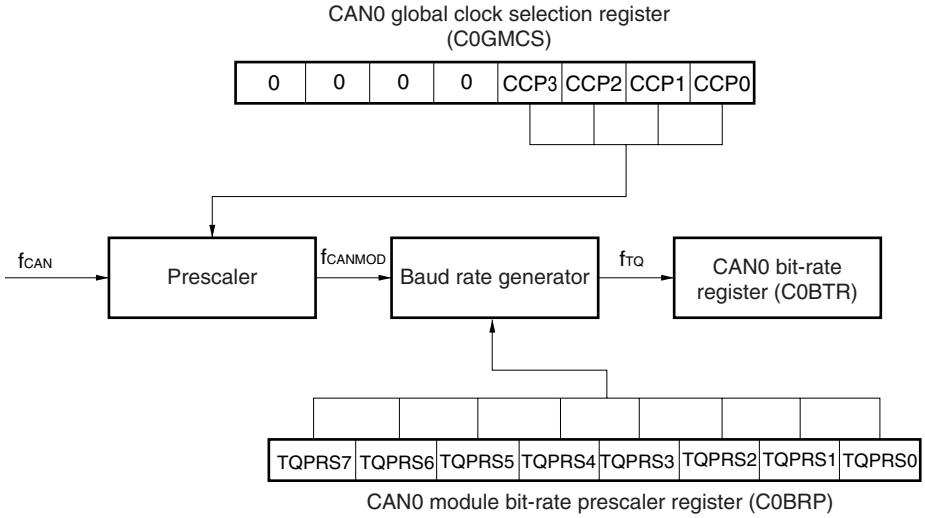
The C0BRP register is used to select the CAN protocol layer base clock (f_{TQ}). The communication baud rate is set to the C0BTR register.

After reset: FFH R/W Address: 03FEC05AH

	7	6	5	4	3	2	1	0
C0BRP	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0

TQPRS7 to TQPRS0	CAN protocol layer base system clock (f_{TQ})
0	f_{CANMOD}/1
1	f_{CANMOD}/2
n	f_{CANMOD}/(n+1)
:	:
255	f_{CANMOD}/256 (default value)

Figure 17-23. CAN Module Clock



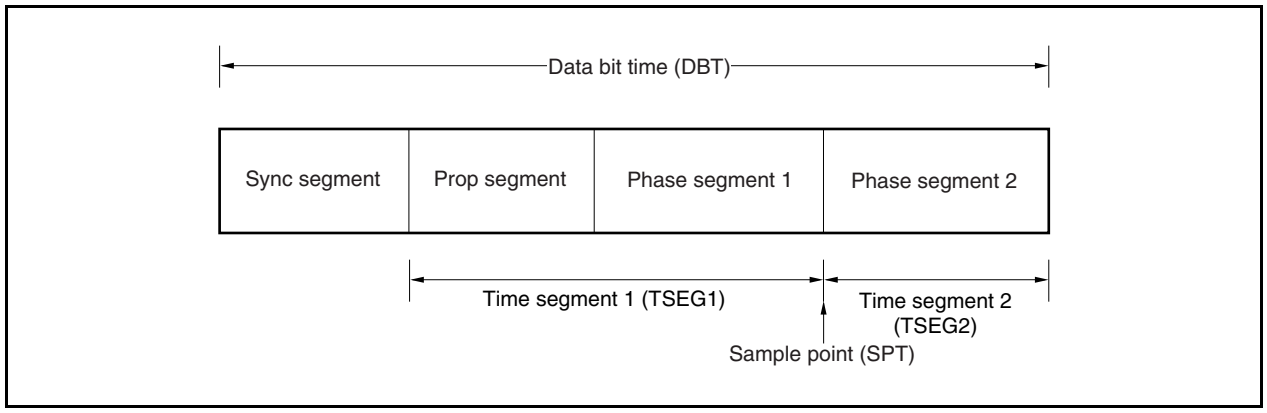
Remark f_{CAN}: Clock supplied to CAN = f_{XX}
f_{CANMOD}: CAN module system clock
f_{TQ}: CAN protocol layer base system clock

Caution The C0BRP register can be write-accessed only in the initialization mode.

(13) CAN0 module bit rate register (C0BTR)

The C0BTR register is used to control the data bit time of the communication baud rate.

Figure 17-24. Data Bit Time



After reset: 370FH R/W Address: 03FEC05CH

	15	14	13	12	11	10	9	8
C0BTR	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20
	7	6	5	4	3	2	1	0
	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10

SJW1	SJW0	Length of synchronization jump width
0	0	1TQ
0	1	2TQ
1	0	3TQ
1	1	4TQ (default value)

TSEG22	TSEG21	TSEG20	Length of time segment 2
0	0	0	1TQ
0	0	1	2TQ
0	1	0	3TQ
0	1	1	4TQ
1	0	0	5TQ
1	0	1	6TQ
1	1	0	7TQ
1	1	1	8TQ (default value)

TSEG13	TSEG12	TSEG11	TSEG10	Length of time segment 1
0	0	0	0	Setting prohibited
0	0	0	1	2TQ[Note]
0	0	1	0	3TQ[Note]
0	0	1	1	4TQ
0	1	0	0	5TQ
0	1	0	1	6TQ
0	1	1	0	7TQ
0	1	1	1	8TQ
1	0	0	0	9TQ
1	0	0	1	10TQ
1	0	1	0	11TQ
1	0	1	1	12TQ
1	1	0	0	13TQ
1	1	0	1	14TQ
1	1	1	0	15TQ
1	1	1	1	16TQ (default value)

Note This setting must not be made when the C0BRP register = 00H.

Remark TQ = 1/f_{TQ} (f_{TQ}: CAN protocol layer base system clock)

(14) CAN0 module last in-pointer register (C0LIPT)

The C0LIPT register indicates the number of the message buffer in which a data frame or a remote frame was last stored.

After reset: Undefined R Address: 03FEC05EH

	7	6	5	4	3	2	1	0
C0LIPT	LIPT7	LIPT6	LIPT5	LIPT4	LIPT3	LIPT2	LIPT1	LIPT0

LIPT7 to LIPT0	Last in-pointer register (C0LIPT)
0 to 31	When the C0LIPT register is read, the contents of the element indexed by the last in-pointer (LIPT) of the receive history list are read. These contents indicate the number of the message buffer in which a data frame or a remote frame was last stored.

Remark The read value of the C0LIPT register is undefined if a data frame or a remote frame has never been stored in the message buffer. If the C0RGPT.RHPM bit is set to 1 after the CAN module has changed from the initialization mode to an operation mode, therefore, the read value of the C0LIPT register is undefined.

(15) CAN0 module receive history list register (C0RGPT)

The C0RGPT register is used to read the receive history list.

(1/2)

After reset: xx02H R/W Address: 03FEC060H

(a) Read

	15	14	13	12	11	10	9	8
C0RGPT	RGPT7	RGPT6	RGPT5	RGPT4	RGPT3	RGPT2	RGPT1	RGPT0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	RHPM	ROVF

(b) Write

	15	14	13	12	11	10	9	8
C0RGPT	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	Clear ROVF

(a) Read

RGPT7 to RGPT0	Receive history list read pointer
0 to 31	When the C0RGPT register is read, the contents of the element indexed by the receive history list get pointer (RGPT) of the receive history list are read. These contents indicate the number of the message buffer in which a data frame or a remote frame has been stored.

RHPM[Note 1]	Receive history list pointer match
0	The receive history list has at least one message buffer number that has not been read.
1	The receive history list has no message buffer numbers that have not been read.

ROVF[Note 2]	Receive history list overflow bit
0	All the message buffer numbers that have not been read are preserved. All the numbers of the message buffers in which a new data frame or remote frame has been received and stored are recorded to the receive history list (the receive history list has a vacant element).
1	At least 23 entries have been stored since the host processor serviced the RHL last time (i.e. read C0RGPT). The first 22 entries are sequentially stored whereas the last entry might have been overwritten by newly received messages a number of times because all buffer numbers are stored at position LIPT-1 when the ROVF bit is set to 1. As a consequence receptions cannot be completely recovered in the order that they were received.

Notes 1. The read value of the RGPT0 to RGPT7 bits is invalid when the RHPM bit = 1.

2. If all the receive history is read by the C0RGPT register while the ROVF bit is set (1), the RHPM bit is not cleared (0) but kept set (1) even if newly received data is stored.

(2/2)

(b) Write

Clear ROVF	Setting of ROVF bit
0	ROVF bit is not changed.
1	ROVF bit is cleared to 0.

(16) CAN0 module last out-pointer register (C0LOPT)

The C0LOPT register indicates the number of the message buffer to which a data frame or a remote frame was transmitted last.

After reset: Undefined R Address: 03FEC062H

	7	6	5	4	3	2	1	0
C0LOPT	LOPT7	LOPT6	LOPT5	LOPT4	LOPT3	LOPT2	LOPT1	LOPT0

LOPT7 to LOPT0	Last out-pointer of transmit history list (LOPT)
0 to 31	When the C0LOPT register is read, the contents of the element indexed by the last out-pointer (LOPT) of the receive history list are read. These contents indicate the number of the message buffer to which a data frame or a remote frame was transmitted last.

Remark The value read from the C0LOPT register is undefined if a data frame or remote frame has never been transmitted from a message buffer. If the C0TGPT.THPM bit is set to 1 after the CAN module has changed from the initialization mode to an operation mode, therefore, the read value of the C0LOPT register is undefined.

(17) CAN0 module transmit history list register (C0TGPT)

The C0TGPT register is used to read the transmit history list.

After reset: xx02H R/W Address: 03FEC064H

(a) Read

	15	14	13	12	11	10	9	8
C0TGPT	TGPT7	TGPT6	TGPT5	TGPT4	TGPT3	TGPT2	TGPT1	TGPT0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	THPM	TOVF

(b) Write

	15	14	13	12	11	10	9	8
C0TGPT	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	Clear TOVF

(a) Read

TGPT7 to TGPT0	Transmit history list read pointer
0 to 31	When the C0TGPT register is read, the contents of the element indexed by the read pointer (TGPT) of the transmit history list are read. These contents indicate the number of the message buffer to which a data frame or a remote frame was transmitted last.

THPM[Note 1]	Transmit history pointer match
0	The transmit history list has at least one message buffer number that has not been read.
1	The transmit history list has no message buffer numbers that have not been read.

TOVF[Note 2]	Transmit history list overflow bit
0	All the message buffer numbers that have not been read are preserved. All the numbers of the message buffers to which a new data frame or remote frame has been transmitted are recorded to the transmit history list (the transmit history list has a vacant element).
1	At least 7 entries have been stored since the host processor serviced the THL last time (i.e. read C0TGPT). The first 6 entries are sequentially stored whereas the last entry might have been overwritten by newly transmitted messages a number of times because all buffer numbers are stored at position LOPT-1 when TOVF bit is set to 1. As a consequence receptions cannot be completely recovered in the order that they were received.

Notes 1. The read value of the TGPT0 to TGPT7 bits is invalid when the THPM bit = 1.

2. If all the transmit history is read by the C0TGPT register while the TOVF bit is set (1), the THPM bit is not cleared (0) but kept set (1), even if transmission of new data has been completed.

Remark Transmission from message buffers 0 to 7 is not recorded to the transmit history list in the normal operation mode with ABT.

(2/2)

(b) Write

Clear TOVF	Setting of TOVF bit
0	TOVF bit is not changed.
1	TOVF bit is cleared to 0.

(18) CAN0 message data byte register (C0MDATAxm, C0MDATAym) (x = 0 to 7, y = 01, 23, 45, 67)

The C0MDATAxm register is used to store the data of a transmit/receive message, and can be accessed in 8-bit unit.

The C0MDATAxm register can be accessed in 16-bit units by the C0MDATAym register.

(1/2)

After reset: Undefined R/W Address: See **Table 17-16**.

C0MDATA01m

15	14	13	12	11	10	9	8
MDATA01 15	MDATA01 14	MDATA01 13	MDATA01 12	MDATA01 11	MDATA01 10	MDATA01 9	MDATA01 8
7	6	5	4	3	2	1	0
MDATA01 7	MDATA01 6	MDATA01 5	MDATA01 4	MDATA01 3	MDATA01 2	MDATA01 1	MDATA01 0

C0MDATA0m

7	6	5	4	3	2	1	0
MDATA0 7	MDATA0 6	MDATA0 5	MDATA0 4	MDATA0 3	MDATA0 2	MDATA0 1	MDATA0 0

C0MDATA1m

7	6	5	4	3	2	1	0
MDATA1 7	MDATA1 6	MDATA1 5	MDATA1 4	MDATA1 3	MDATA1 2	MDATA1 1	MDATA1 0

C0MDATA23m

15	14	13	12	11	10	9	8
MDATA23 15	MDATA23 14	MDATA23 13	MDATA23 12	MDATA23 11	MDATA23 10	MDATA23 9	MDATA23 8
7	6	5	4	3	2	1	0
MDATA23 7	MDATA23 6	MDATA23 5	MDATA23 4	MDATA23 3	MDATA23 2	MDATA23 1	MDATA23 0

C0MDATA2m

7	6	5	4	3	2	1	0
MDATA2 7	MDATA2 6	MDATA2 5	MDATA2 4	MDATA2 3	MDATA2 2	MDATA2 1	MDATA2 0

(2/2)

C0MDATA3m

7	6	5	4	3	2	1	0
MDATA37	MDATA36	MDATA35	MDATA34	MDATA33	MDATA32	MDATA31	MDATA30

C0MDATA45m

15	14	13	12	11	10	9	8
MDATA4515	MDATA4514	MDATA4513	MDATA4512	MDATA4511	MDATA4510	MDATA459	MDATA458

7	6	5	4	3	2	1	0
MDATA457	MDATA456	MDATA455	MDATA454	MDATA453	MDATA452	MDATA451	MDATA450

C0MDATA4m

7	6	5	4	3	2	1	0
MDATA47	MDATA46	MDATA45	MDATA44	MDATA43	MDATA42	MDATA41	MDATA40

C0MDATA5m

7	6	5	4	3	2	1	0
MDATA57	MDATA56	MDATA55	MDATA54	MDATA53	MDATA52	MDATA51	MDATA50

C0MDATA67m

15	14	13	12	11	10	9	8
MDATA6715	MDATA6714	MDATA6713	MDATA6712	MDATA6711	MDATA6710	MDATA679	MDATA678

7	6	5	4	3	2	1	0
MDATA677	MDATA676	MDATA675	MDATA674	MDATA673	MDATA672	MDATA671	MDATA670

C0MDATA6m

7	6	5	4	3	2	1	0
MDATA67	MDATA66	MDATA65	MDATA64	MDATA63	MDATA62	MDATA61	MDATA60

C0MDATA7m

7	6	5	4	3	2	1	0
MDATA77	MDATA76	MDATA75	MDATA74	MDATA73	MDATA72	MDATA71	MDATA70

(19) CAN0 message data length register m (C0MDLCm)

The C0MDLCm register is used to set the number of bytes of the data field of a message buffer.

After reset: 0000xxxxB R/W Address: See **Table 17-16**.

	7	6	5	4	3	2	1	0
C0MDLCm	0	0	0	0	MDLC3	MDLC2	MDLC1	MDLC0

MDLC3	MDLC2	MDLC1	MDLC0	Data length of transmit/receive message
0	0	0	0	0 bytes
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	0	0	0	8 bytes
1	0	0	1	Setting prohibited (If these bits are set during transmission, 8-byte data is transmitted regardless of the set DLC value when a data frame is transmitted. However, the DLC actually transmitted to the CAN bus is the DLC value set to this register.)[Note]
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Note The data and DLC value actually transmitted to CAN bus are as follows.

Type of transmit frame	Length of transmit data	DLC transmitted
Data frame	Number of bytes specified by DLC (However, 8 bytes if DLC ≥ 8)	MDLC3 to MDLC0 bits
Remote frame	0 bytes	

Cautions 1. Be sure to set bits 7 to 4 to 0000B.

2. Receive data is stored in as many C0MDATAxm registers as the number of bytes (however, the upper limit is 8) corresponding to the DLC of a receive frame. The C0MDATAxm register in which no data is stored is undefined.

(20) CAN0 message configuration register m (C0MCONFm)

The C0MCONFm register is used to specify the type of the message buffer and to set a mask.

(1/2)

After reset: Undefined R/W Address: See **Table 17-16**.

	7	6	5	4	3	2	1	0
C0MCONFm	OWS	RTR	MT2	MT1	MT0	0	0	MA0

OWS	Overwrite control bit
0	The message buffer[Note] that has already received a data frame is not overwritten by a newly received data frame. The newly received data frame is discarded.
1	The message buffer that has already received a data frame is overwritten by a newly received data frame.

Note The “message buffer that has already received a data frame” is a receive message buffer whose the C0MCTRLm.DN bit has been set to 1.

Remark A remote frame is received and stored, regardless of the setting of the OWS and DN bits. A remote frame that satisfies the other conditions (ID matches, the RTR bit = 0, the C0MCTRLm.TRQ bit = 0) is always received and stored in the corresponding message buffer (interrupt generated, DN flag set, the C0MDLCm.MDLC0 to C0MDLCm.MDLC3 bits updated, and recorded to the receive history list).

RTR	Remote frame request bit[Note]
0	Transmit a data frame.
1	Transmit a remote frame.

Note The RTR bit specifies the type of message frame that is transmitted from a message buffer defined as a transmit message buffer. Even if a valid remote frame has been received, the RTR bit of the transmit message buffer that has received the frame remains cleared to 0. Even if a remote frame whose ID matches has been received from the CAN bus with the RTR bit of the transmit message buffer set to 1 to transmit a remote frame, that remote frame is not received or stored (interrupt generated, DN flag set, the MDLC0 to MDLC3 bits updated, and recorded to the receive history list).

MT2	MT1	MT0	Message buffer type setting bit
0	0	0	Transmit message buffer
0	0	1	Receive message buffer (no mask setting)
0	1	0	Receive message buffer (mask 1 set)
0	1	1	Receive message buffer (mask 2 set)
1	0	0	Receive message buffer (mask 3 set)
1	0	1	Receive message buffer (mask 4 set)
Other than above			Setting prohibited

(2/2)

MA0	Message buffer assignment bit
0	Message buffer not used.
1	Message buffer used.

Caution Be sure to write 0 to bits 2 and 1.

(21) CAN0 message ID register m (C0MIDLm, C0MIDHm)

The C0MIDLm and C0MIDHm registers are used to set an identifier (ID).

After reset: Undefined R/W Address: See **Table 17-16**.

	15	14	13	12	11	10	9	8
C0MIDLm	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
	7	6	5	4	3	2	1	0
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0

	15	14	13	12	11	10	9	8
C0MIDHm	IDE	0	0	ID28	ID27	ID26	ID25	ID24
	7	6	5	4	3	2	1	0
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16

IDE	Format mode specification bit
0	Standard format mode (ID28 to ID18: 11 bits)[Note]
1	Extended format mode (ID28 to ID0: 29 bits)

Note The ID17 to ID0 bits are not used.

ID28 to ID0	Message ID
ID28 to ID18	Standard ID value of 11 bits (when IDE = 0)
ID28 to ID0	Extended ID value of 29 bits (when IDE = 1)

Cautions 1 Be sure to write 0 to bits 14 and 13 of the C0MIDHm register.

2. Be sure to arrange the ID values to be registered in accordance with the bit positions of this register. For the standard ID, shift the bit positions of ID28 to ID18 of the ID value.

(22) CAN0 message control register m (C0MCTRLm)

The C0MCTRLm register is used to control the operation of the message buffer.

(1/3)

After reset: 00x000000 R/W Address: See **Table 17-16**.
000xx000B

(a) Read

	15	14	13	12	11	10	9	8
C0MCTRLm	0	0	MUC	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	MOW	IE	DN	TRQ	RDY

(b) Write

	15	14	13	12	11	10	9	8
C0MCTRLm	0	0	0	0	Set IE	0	Set TRQ	Set RDY
	7	6	5	4	3	2	1	0
	0	0	0	Clear MOW	Clear IE	Clear DN	Clear TRQ	Clear RDY

(a) Read

MUC[Note]	Bit indicating that message buffer data is being updated
0	The CAN module is not updating the message buffer (reception and storage).
1	The CAN module is updating the message buffer (reception and storage).

Note The MUC bit is undefined until the first reception and storage is performed.

MOW	Message buffer overwrite status bit
0	The message buffer is not overwritten by a newly received data frame.
1	The message buffer is overwritten by a newly received data frame.

Remark The MOW bit is not set to 1 even if a remote frame is received and stored in the transmit message buffer with the DN bit = 1.

IE	Message buffer interrupt request enable bit
0	Receive message buffer: Valid message reception completion interrupt disabled. Transmit message buffer: Normal message transmission completion interrupt disabled.
1	Receive message buffer: Valid message reception completion interrupt enabled. Transmit message buffer: Normal message transmission completion interrupt enabled.

DN	Message buffer data update bit
0	A data frame or remote frame is not stored in the message buffer.
1	A data frame or remote frame is stored in the message buffer.

(2/3)

TRQ	Message buffer transmission request bit
0	No message frame transmitting request that is pending or being transmitted is in the message buffer.
1	The message buffer is holding transmission of a message frame pending or is transmitting a message frame.

Caution Do not set the TRQ bit and RDY bit to 1 at the same time. Be sure to set the RDY bit to 1 before setting the TRQ bit to 1.

RDY	Message buffer ready bit
0	The message buffer can be written by software. The CAN module cannot write to the message buffer.
1	Writing the message buffer by software is ignored (except a write access to the RDY, TRQ, DN, and MOW bits). The CAN module can write to the message buffer.

Cautions 1. Do not clear (0) the RDY bit during message transmission. Follow transmission abort procedures in order to clear the RDY bit for redefinition.

2. If the RDY bit is not cleared (0) even when the processing to clear it is executed, execute the clearing processing again.

3. Confirm, by reading the RDY bit again, that the RDY bit has been cleared (0) before writing data to the message buffer.
However, it is unnecessary to confirm that the TRQ or RDY bit has been set (1) or that the DN or MOW bit has been cleared (0).

(b) Write

Clear MOW	Setting of MOW bit
0	MOW bit is not changed.
1	MOW bit is cleared to 0.

Set IE	Clear IE	Setting of IE bit
0	1	IE bit is cleared to 0.
1	0	IE bit is set to 1.
Other than above		IE bit is not changed.

Caution Be sure to set the IE and RDY bits separately.

Clear DN	Setting of DN bit
1	DN bit is cleared to 0.
0	DN bit is not changed.

Caution Do not set the DN bit to 1 by software. Be sure to write 0 to bit 10.

(3/3)

Set TRQ	Clear TRQ	Setting of TRQ bit
0	1	TRQ bit is cleared to 0.
1	0	TRQ bit is set to 1.
Other than above		TRQ bit is not changed.

Caution Even if the TRQ bit is set (1), transmission may not be immediately executed depending on the situation such as when a message is received from another node or when a message is transmitted from the message buffer.

Transmission under execution is not terminated midway even if the TRQ bit is cleared. Transmission is continued until it is completed (regardless of whether it is executed successfully or fails).

Set RDY	Clear RDY	Setting of RDY bit
0	1	RDY bit is cleared to 0.
1	0	RDY bit is set to 1.
Other than above		RDY bit is not changed.

Caution Be sure to set the TRQ and RDY bits separately.

17.7 Bit Set/Clear Function

The CAN control registers include registers whose bits can be set or cleared via the CPU and via the CAN interface. An operation error occurs if the following registers are written directly. Do not write any values directly via bit manipulation, read/modify/write, or direct writing of target values.

- CAN0 global control register (C0GMCTRL)
- CAN0 global automatic block transmission control register (C0GMABT)
- CAN0 module control register (C0CTRL)
- CAN0 module interrupt enable register (C0IE)
- CAN0 module interrupt status register (C0INTS)
- CAN0 module receive history list register (C0RGPT)
- CAN0 module transmit history list register (C0TGPT)
- CAN0 message control register (C0MCTRLm)

Remark m = 00 to 31

All the 16 bits in the above registers can be read via the usual method. Use the procedure described in Figure 17-25 below to set or clear the lower 8 bits in these registers.

Setting or clearing of lower 8 bits in the above registers is performed in combination with the higher 8 bits (refer to the bit status after set/clear operation is specified in Figure 17-26). Figure 17-25 shows how the values of set bits or clear bits relate to set/clear/no change operations in the corresponding register.

Figure 17-25. Example of Bit Setting/Clearing Operations

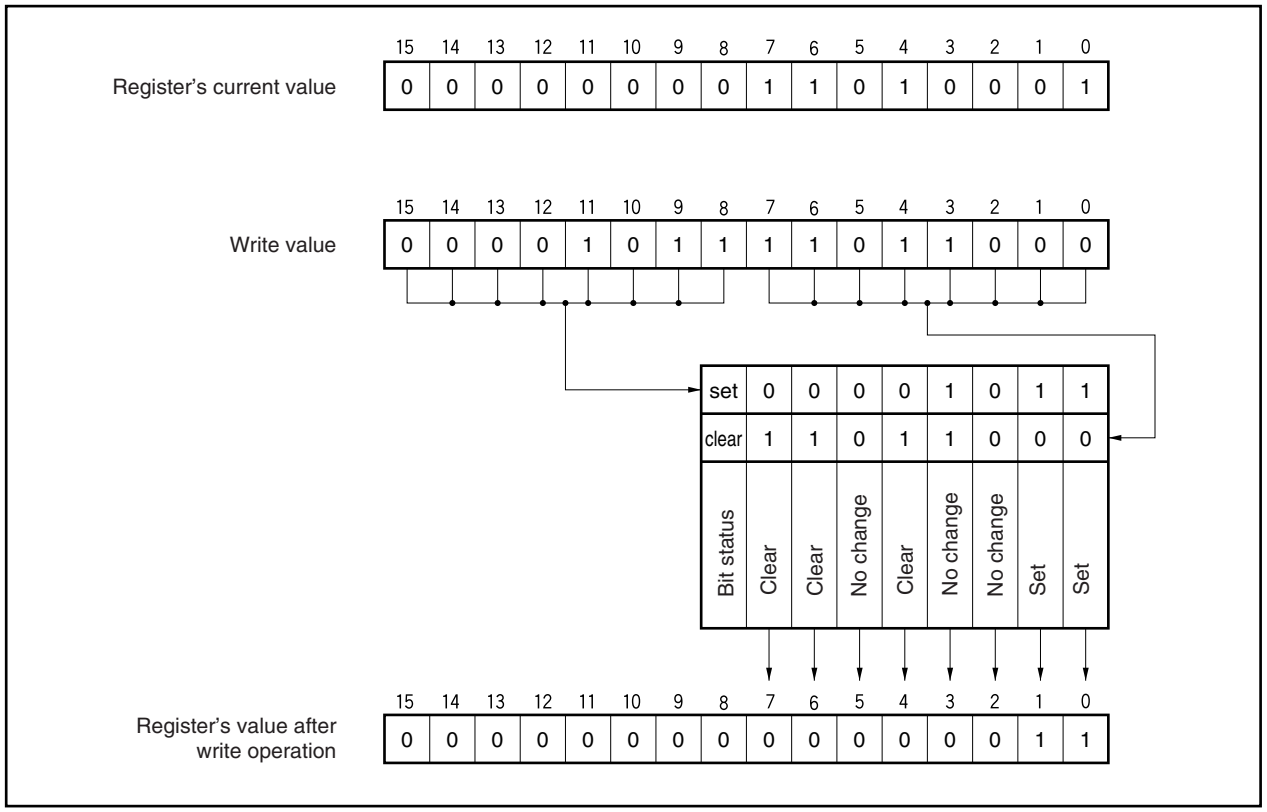


Figure 17-26. Bit Status After Bit Setting/Clearing Operations

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Set 7	Set 6	Set 5	Set 4	Set 3	Set 2	Set 1	Set 0	Clear 7	Clear 6	Clear 5	Clear 4	Clear 3	Clear 2	Clear 1	Clear 0

Set n	Clear n	Status of bit n after bit set/clear operation
0	0	No change
0	1	0
1	0	1
1	1	No change

Remark n = 0 to 7

17.8 CAN Controller Initialization

17.8.1 Initialization of CAN module

Before CAN module operation is enabled, the CAN module system clock needs to be determined by setting the C0GMCS.CCP0 to C0GMCS.CCP3 bits by software. Do not change the setting of the CAN module system clock after CAN module operation is enabled.

The CAN module is enabled by setting the C0GMCTRL.GOM bit.

For the procedure of initializing the CAN module, see **17.15 Operation of CAN Controller.**

17.8.2 Initialization of message buffer

After the CAN module is enabled, the message buffers contain undefined values. A minimum initialization for all the message buffers, even for those not used in the application, is necessary before switching the CAN module from the initialization mode to one of the operation modes.

- Clear the C0MCTRLm.RDY, C0MCTRLm.TRQ, and C0MCTRLm.DN bits to 0.
- Clear the C0MCONFm.MA0 bit to 0.

Remark m = 00 to 31

17.8.3 Redefinition of message buffer

Redefining a message buffer means changing the ID and control information of the message buffer while a message is being received or transmitted, without affecting other transmission/reception operations.

(1) To redefine message buffer in initialization mode

Place the CAN module in the initialization mode once and then change the ID and control information of the message buffer in the initialization mode. After changing the ID and control information, set the CAN module to an operation mode.

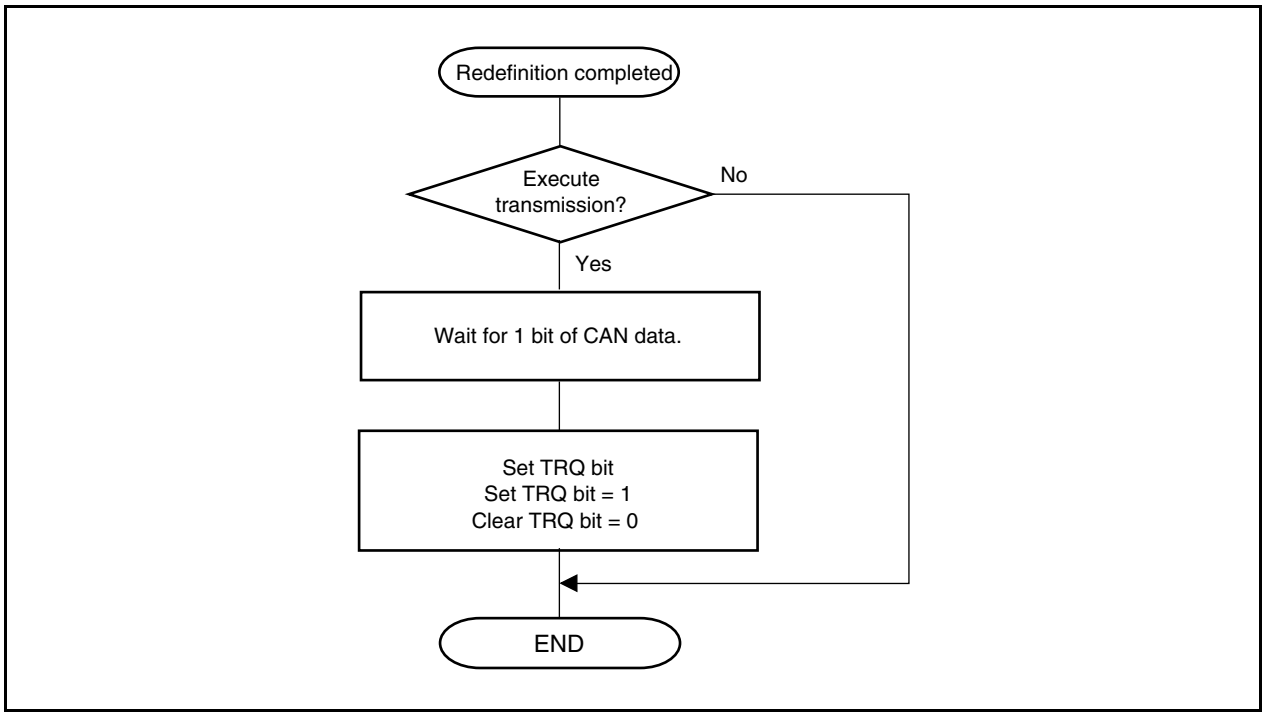
(2) To redefine message buffer during reception

Perform redefinition as shown in Figure 17-38.

(3) To redefine message buffer during transmission

To rewrite the contents of a transmit message buffer to which a transmission request has been set, perform transmission abort processing (see **17.10.4 (1) Transmission abort process other than in normal operation mode with automatic block transmission (ABT), 17.10.4 (2) Transmission abort process except for ABT transmission in normal operation mode with automatic block transmission (ABT)**). Confirm that transmission has been aborted or completed, and then redefine the message buffer. After redefining the transmit message buffer, set a transmission request using the procedure described below. When setting a transmission request to a message buffer that has been redefined without aborting the transmission in progress, however, the 1-bit wait time is not necessary.

Figure 17-27. Setting Transmission Request (TRQ) to Transmit Message Buffer After Redefinition



Cautions 1. When a message is received, reception filtering is performed in accordance with the ID and mask set to each receive message buffer. If the procedure in Figure 17-39 is not observed, the contents of the message buffer after it has been redefined may contradict the result of reception (result of reception filtering). If this happens, check that the ID and IDE received first and stored in the message buffer following redefinition are those stored after the message buffer has been redefined. If no ID and IDE are stored after redefinition, redefine the message buffer again.

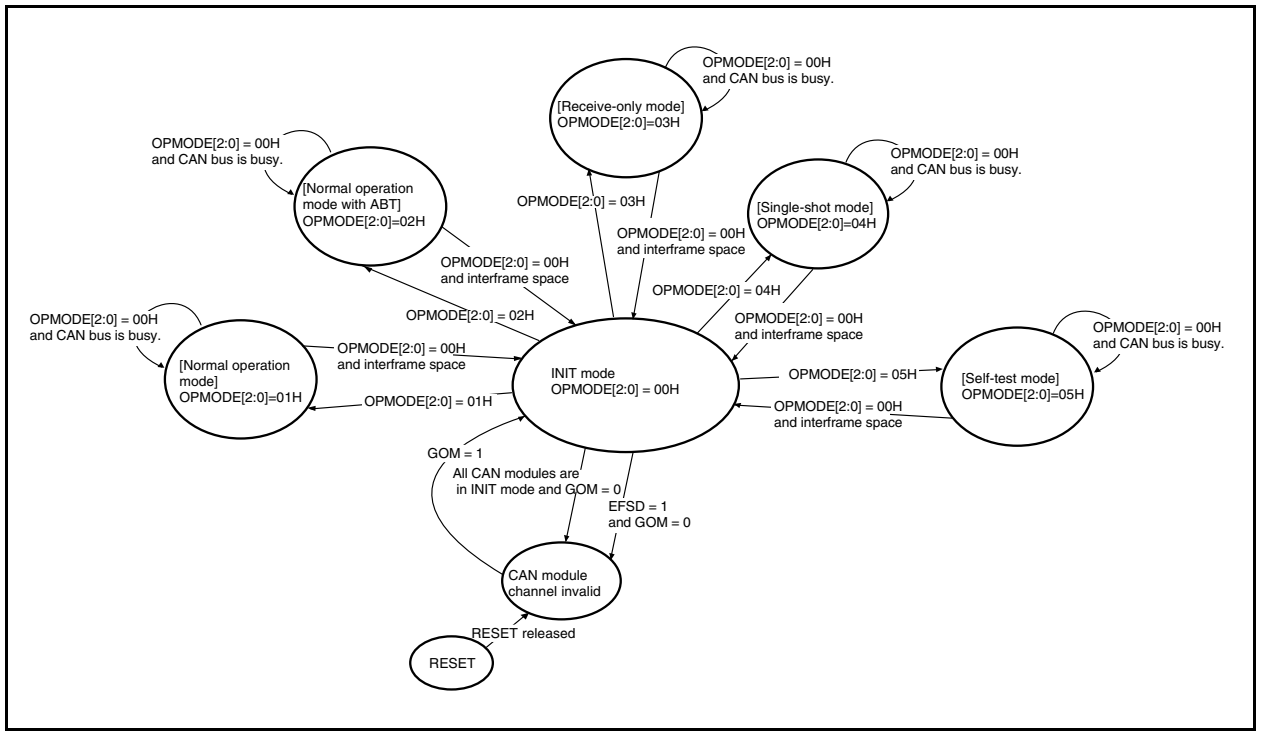
2. When a message is transmitted, the transmission priority is checked in accordance with the ID, IDE, and RTR bits set to each transmit message buffer to which a transmission request was set. The transmit message buffer having the highest priority is selected for transmission. If the procedure in Figure 17-27 is not observed, a message with an ID not having the highest priority may be transmitted after redefinition.

17.8.4 Transition from initialization mode to operation mode

The CAN module can be switched to the following operation modes.

- Normal operation mode
- Normal operation mode with ABT
- Receive-only mode
- Single-shot mode
- Self-test mode

Figure 17-28. Transition to Operation Modes



The transition from the initialization mode to an operation mode is controlled by the C0CTRL.OPMODE2 to C0CTRL.OPMODE0 bits.

Changing from one operation mode into another requires shifting to the initialization mode in between. Do not change one operation mode to another directly; otherwise the operation will not be guaranteed.

Requests for transition from an operation mode to the initialization mode are held pending when the CAN bus is not in the interframe space (i.e., frame reception or transmission is in progress), and the CAN module enters the initialization mode at the first bit in the interframe space (the values of the OPMODE2 to OPMODE0 bits are changed to 000B). After issuing a request to change the mode to the initialization mode, read the OPMODE2 to OPMODE0 bits until their values become 000B to confirm that the module has entered the initialization mode (see **Figure 17-36**).

17.8.5 Resetting error counter C0ERC of CAN module

If it is necessary to reset the C0ERC and C0INFO registers when re-initialization or forced recovery from the bus-off status is made, set the C0CTRL.CCERC bit to 1 in the initialization mode. When this bit is set to 1, the C0ERC and C0INFO registers are cleared to their default values.

17.9 Message Reception

17.9.1 Message reception

All buffers satisfying the following conditions are searched in all the message buffer areas in all the operation modes in order to store newly received messages.

- Used as a message buffer
  (C0MCONFm.MA0 bit is set to 1.)
- Set as a receive message buffer
  (C0MCONFm.MT2 to C0MCONFm.MT0 bits are set to 001B, 010B, 011B, 100B, or 101B.)
- Ready for reception
  (C0MCTRLm.RDY bit is set to 1.)

Remark m = 00 to 31

When two or more message buffers of the CAN module receive a message, the message is stored according to the priority explained below. The message is always stored in the message buffer with the highest priority, not in a message buffer with a low priority. For example, when an unmasked receive message buffer and a receive message buffer linked to mask 1 have the same ID, the received message is not stored in the message buffer linked to mask 1 that has not received a message, even if a message has already been received in the unmasked receive message buffer. In other words, when a condition has been set to store a message in two or more message buffers with different priorities, the message buffer with the highest priority always stores the message; the message is not stored in message buffers with a lower priority. This also applies when the message buffer with the highest priority is unable to receive and store a message (i.e., when the DN bit = 1 indicating that a message has already been received, but rewriting is disabled because the OWS bit = 0). In this case, the message is not actually received and stored in the candidate message buffer with the highest priority, but neither is it stored in a message buffer with a lower priority.

Priority	Storing Condition If Same ID Is Set	
1 (high)	Unmasked message buffer	DN bit = 0
		DN bit = 1 and OWS bit = 1
2	Message buffer linked to mask 1	DN bit = 0
		DN bit = 1 and OWS bit = 1
3	Message buffer linked to mask 2	DN bit = 0
		DN bit = 1 and OWS bit = 1
4	Message buffer linked to mask 3	DN bit = 0
		DN bit = 1 and OWS bit = 1
5 (low)	Message buffer linked to mask 4	DN bit = 0
		DN bit = 1 and OWS bit = 1

17.9.2 Reading reception data

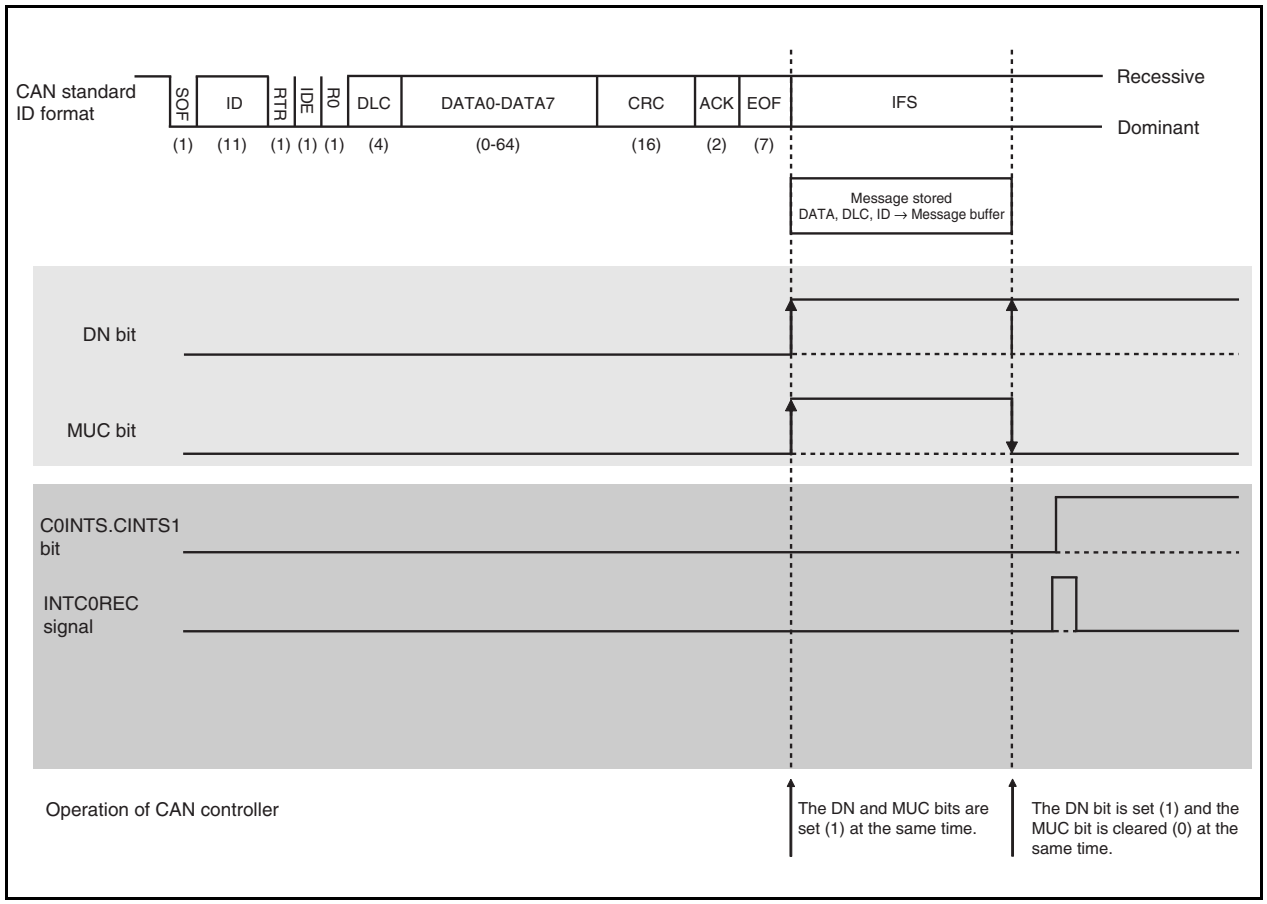
If it is necessary to consistently read data from the CAN message buffer by software, follow the recommended procedures shown in Figures 17-48 and 17-49.

While receiving a message, the CAN module sets the C0MCTRLm.DN bit two times, at the beginning of the processing to store data in the message buffer and at the end of this storing processing. During this storing processing, the C0MCTRLm.MUC bit of the message buffer is set (1) (refer to **Figure 17-29**).

Before the data is completely stored, the receive history list is written. During this data storing period (MUC bit = 1), the CPU is prohibited from rewriting the C0MCTRLm.RDY bit of the message buffer in which the data is to be stored. Completion of this data storing processing may be delayed by a CPU's access to any message buffer.

Remark m = 0 to 31

Figure 17-29. DN and MUC Bit Setting Period (in Standard ID Format)



17.9.3 Receive history list function

The receive history list (RHL) function records in the receive history list the number of the receive message buffer in which each data frame or remote frame was received and stored. The RHL consists of storage elements equivalent to up to 23 messages, the last in-message pointer (LIPT) with the corresponding C0LIPT register and the receive history list get pointer (RGPT) with the corresponding C0RGPT register.

The RHL is undefined immediately after the transition of the CAN module from the initialization mode to one of the operation modes.

The C0LIPT register holds the contents of the RHL element indicated by the value of the LIPT pointer minus 1. By reading the C0LIPT register, therefore, the number of the message buffer that received and stored a data frame or remote frame first can be checked. The LIPT pointer is utilized as a write pointer that indicates to what part of the RHL a message buffer number is recorded. Any time a data frame or remote frame is received and stored, the corresponding message buffer number is recorded to the RHL element indicated by the LIPT pointer. Each time recording to the RHL has been completed, the LIPT pointer is automatically incremented. In this way, the number of the message buffer that has received and stored a frame will be recorded chronologically.

The RGPT pointer is utilized as a read pointer that reads a recorded message buffer number from the RHL. This pointer indicates the first RHL element that the CPU has not read yet. By reading the C0RGPT register by software, the number of a message buffer that has received and stored a data frame or remote frame can be read. Each time a message buffer number is read from the C0RGPT register, the RGPT pointer is automatically incremented.

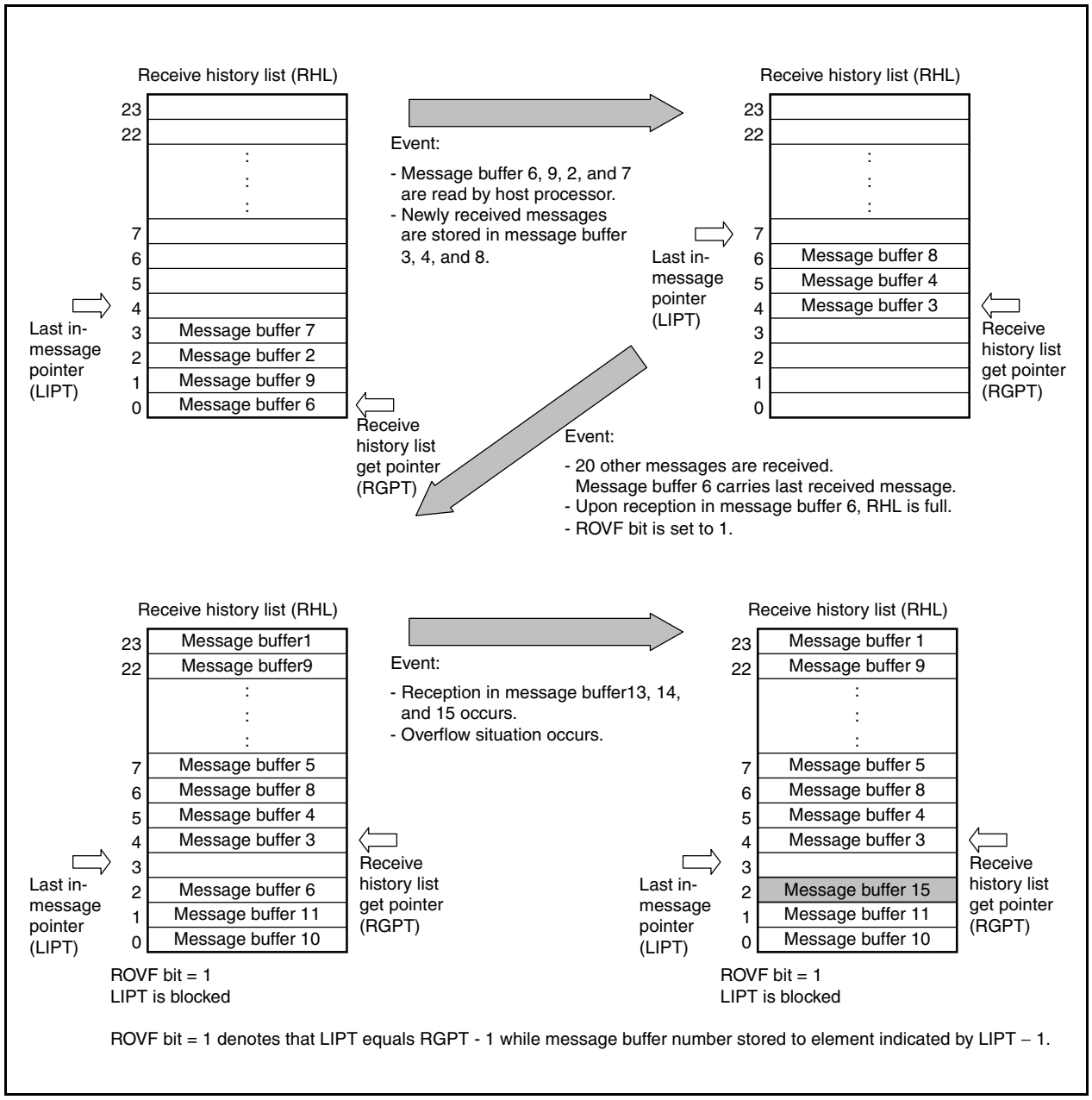
If the value of the RGPT pointer matches the value of the LIPT pointer, the C0RGPT.RHPM bit (receive history list pointer match) is set to 1. This indicates that no message buffer number that has not been read remains in the RHL. If a new message buffer number is recorded, the LIPT pointer is incremented and because its value no longer matches the value of the RGPT pointer, the RHPM bit is cleared. In other words, the numbers of the unread message buffers exist in the RHL.

If the LIPT pointer is incremented and matches the value of the RGPT pointer minus 1, the C0RGPT.ROVF bit (receive history list overflow) is set to 1. This indicates that the RHL is full of numbers of message buffers that have not been read. When further message reception and storing occur, the last recorded message buffer number is overwritten by the number of the message buffer that received and stored the new message. After the ROVF bit has been set to 1, the recorded message buffer numbers in the RHL do not completely reflect chronological order.
However the messages themselves are not lost and can be located by a CPU search in the message buffer memory with the help of the DN bit.

Caution Even if the receive history list overflows (C0RGPT.ROVF bit = 1), the receive history can be read until no more history is left unread and the C0RGPT.RHPM bit is set (1). However, the ROVF bit is kept set (1) (= overflow occurs) until cleared (0) by software. In this status, the RHPM bit is not cleared (0), unless the ROVF bit is cleared (0), even if a new receive history is stored and written to the list. If ROVF bit = 1 and RHPM bit = 1 and the receive history list overflows, therefore, the RHPM bit indicates that no more history is left unread even if new receive history is stored.

As long as the RHL contains 23 or less entries the sequence of occurrence is maintained. If more receptions occur without the RHL being read by the host processor, a complete sequence of receptions can not be recovered.

Figure 17-30. Receive History List



17.9.4 Mask function

It can be defined whether masking of the identifier that is set to a message buffer is linked with another message buffer.

By using the mask function, the identifier of a message received from the CAN bus can be compared with the identifier set to a message buffer in advance. Regardless of whether the masked ID is set to 0 or 1, the received message can be stored in the defined message buffer.

While the mask function is in effect, an identifier bit that is defined to be 1 by a mask in the received message is not compared with the corresponding identifier bit in the message buffer.

However, this comparison is performed for any bit whose value is defined as 0 by the mask.

For example, let us assume that all messages that have a standard-format ID, in which bits ID27 to ID25 are 0 and bits ID24 and ID22 are 1, are to be stored in message buffer 14. The procedure for this example is shown below.

<1> Identifier to be stored in message buffer

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x

Remark x = don't care

<2> Identifier to be configured in message buffer 14 (example)
(Using C0MIDL14 and C0MIDH14 registers)

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
x	x	x	x	x	x	x	x	x	x	x
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
x	x	x	x	x	x	x				

ID with the ID27 to ID25 bits cleared to 0 and the ID24 and ID22 bits set to 1 is registered (initialized) to message buffer 14.

Remark x = don't care

Remark Message buffer 14 is set as a standard format identifier that is linked to mask 1 (C0MCONF14.MT2 to C0MCONF14.MT0 bits are set to 010B).

<3> Mask setting for CAN module 0 (mask 0) (Example)
(Using CAN0 module mask 1 registers L and H (C0MASK1L and C0MASK1H))

CMID28	CMID27	CMID26	CMID25	CMID24	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18
1	0	0	0	0	1	0	1	1	1	1

CMID17	CMID16	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	CMID7
1	1	1	1	1	1	1	1	1	1	1

CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
1	1	1	1	1	1	1

1: Not compared (masked)
0: Compared

The CMID27 to CMID24 and CMID22 bits are cleared to 0, and the CMID28, CMID23, and CMID21 to CMID0 bits are set to 1.

17.9.5 Multi buffer receive block function

The multi buffer receive block (MBRB) function is used to store a block of data in two or more message buffers sequentially with no CPU interaction, by setting the same ID to two or more message buffers with the same message buffer type. These message buffers can be allocated in any area in the message buffer memory, and they are not necessarily to be allocated adjacent to each other.

Suppose, for example, the same message buffer type is set to 10 message buffers, message buffers 10 to 19, and the same ID is set to each message buffer. If the first message whose ID matches an ID of the message buffers is received, it is stored in message buffer 10. At this point, the DN bit of message buffer 10 is set, prohibiting overwriting the message buffer.

When the next message with a matching ID is received, it is received and stored in message buffer 11. Each time a message with a matching ID is received, it is sequentially (in the ascending order) stored in message buffers 12, 13, and so on. Even when a data block consisting of multiple messages is received, the messages can be stored and received without overwriting the previously received matching-ID data.

Whether a data block has been received and stored can be checked by setting the C0MCTRLm.IE bit of each message buffer. For example, if a data block consists of k messages, k message buffers are initialized for reception of the data block. The IE bit in message buffers 0 to (k-2) is cleared to 0 (interrupts disabled), and the IE bit in message buffer k-1 is set to 1 (interrupts enabled). In this case, a reception completion interrupt occurs when a message has been received and stored in message buffer k-1, indicating that MBRB has become full. Alternatively, by clearing the IE bit of message buffers 0 to (k-3) and setting the IE bit of message buffer k-2, a warning that MBRB is about to overflow can be issued.

The basic conditions of storing receive data in each message buffer for the MBRB are the same as the conditions of storing data in a single message buffer.

Cautions 1. MBRB can be configured for each of the same message buffer types. Therefore, even if a message buffer of another MBRB whose ID matches but whose message buffer type is different has a vacancy, the received message is not stored in that message buffer, but instead discarded.

2. MBRB does not have a ring buffer structure. Therefore, after a message is stored in the message buffer having the highest number in the MBRB configuration, a newly received message will no longer be stored in the message buffer in the order from the lowest message buffer number.

3. MBRB operates based on the reception and storage conditions; there are no settings dedicated to MBRB, such as function enable bits. By setting the same message buffer type and ID to two or more message buffers, MBRB is automatically configured.

4. With MBRB, "matching ID" means "matching ID after mask". Even if the ID set to each message buffer is not the same, if the ID that is masked by the mask register matches, it is considered a matching ID and the buffer that has this ID is treated as the storage destination of a message.

5. Priority among each MBRB conforms to the priority shown in 17.9.1 Message reception.

Remark m = 00 to 31

17.9.6 Remote frame reception

In all the operation modes, when a remote frame is received, the message buffer that is to store the remote frame is searched from all the message buffers satisfying the following conditions.

- Used as a message buffer
  (C0MCONFm.MA0 bit set to 1.)
- Set as a transmit message buffer
  (C0MCONFm.MT2 to C0MCONFm.MT0 bits set to 000B)
- Ready for reception
  (C0MCTRLm.RDY bit set to 1.)
- Set to transmit message
  (C0MCONFm.RTR bit is cleared to 0.)
- Transmission request is not set.
  (C0MCTRLm.TRQ bit is set to 0.)

Upon acceptance of a remote frame, the following actions are executed if the ID of the received remote frame matches the ID of a message buffer that satisfies the above conditions.

- The C0MDLCm.DLC3 to C0MDLCm.DLC0 bits store the received DLC value.
- The C0MDATA0m to C0MDATA7m registers in the data area are not updated (data before reception is saved).
- The C0MCTRLm.DN bit is set to 1.
- The C0INTS.CINTS1 bit is set to 1 (if the C0MCTRLm.IE bit of the message buffer that receives and stores the frame is set to 1).
- The receive completion interrupt (INTC0REC) is output (if the IE bit of the message buffer that receives and stores the frame is set to 1 and if the C0IE.CIE1 bit is set to 1).
- The message buffer number is recorded in the receive history list.

Caution When a message buffer is searched for receiving and storing a remote frame, overwrite control by the C0MCONFm.OWS bit of the message buffer and the DN bit are not affected. The setting of the OWS bit is ignored and the DN bit is set to 1 in every case.
If more than one transmit message buffer has the same ID and the ID of the received remote frame matches that ID, the remote frame is stored in the transmit message buffer with the lowest message buffer number.

Remark m = 00 to 31

17.10 Message Transmission

17.10.1 Message transmission

In all the operation modes, if the C0MCTRLm.TRQ bit is set to 1 in a message buffer that satisfies the following conditions, the message buffer that is to transmit a message is searched.

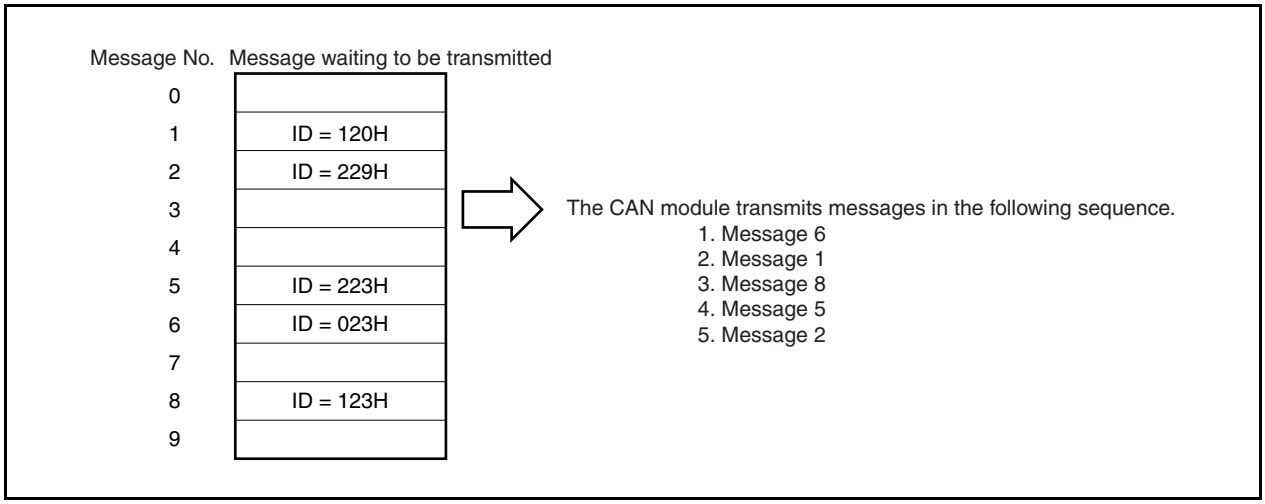
- Used as a message buffer
  (C0MCONFm.MA0 bit set to 1.)
- Set as a transmit message buffer
  (C0MCONFm.MT2 to C0MCONFm.MT0 bits set to 000B.)
- Ready for transmission
  (C0MCTRLm.RDY bit is set to 1.)

Remark m = 00 to 31

The CAN system is a multi-master communication system. In a system like this, the priority of message transmission is determined based on message identifiers (IDs). To facilitate transmission processing by software when there are several messages awaiting transmission, the CAN module uses hardware to check the ID of the message with the highest priority and automatically identifies that message. This eliminates the need for software-based priority control.

Transmission priority is controlled by the identifier (ID).

Figure 17-31. Message Processing Example



After the transmit message search, the transmit message with the highest priority of the transmit message buffers that have a pending transmission request (message buffers with the TRQ bit set to 1 in advance) is transmitted.

If a new transmission request is set, the transmit message buffer with the new transmission request is compared with the transmit message buffer with a pending transmission request. If the new transmission request has a higher priority, it is transmitted, unless transmission of a message with a low priority has already started. To solve this reversal of priorities, software can request that transmission of a message of low priority be stopped. The highest priority is determined according to the following rules.

Priority	Conditions	Description
1 (high)	Value of first 11 bits of ID [ID28 to ID18]:	The message frame with the lowest value represented by the first 11 bits of the ID is transmitted first. If the value of an 11-bit standard ID is equal to or smaller than the first 11 bits of a 29-bit extended ID, the 11-bit standard ID has a higher priority than a message frame with a 29-bit extended ID.
2	Frame type	A data frame with an 11-bit standard ID (C0MCONFm.RTR bit is cleared to 0) has a higher priority than a remote frame with a standard ID and a message frame with an extended ID.
3	ID type	A message frame with a standard ID (C0MIDHm.IDE bit is cleared to 0) has a higher priority than a message frame with an extended ID.
4	Value of lower 18 bits of ID [ID17 to ID0]:	If one or more transmission-pending extended ID message frame has equal values in the first 11 bits of the ID and the same frame type (equal RTR bit values), the message frame with the lowest value in the lower 18 bits of its extended ID is transmitted first.
5 (low)	Message buffer number	If two or more message buffers request transmission of message frames with the same ID, the message from the message buffer with the lowest message buffer number is transmitted first.

Remarks 1. If the automatic block transmission request bit C0GMABT.ABTTRG bit is set to 1 in the normal operation mode with ABT, the TRQ bit is set to 1 only for one message buffer in the ABT message buffer group.

If the ABT mode was triggered by the ABTTRG bit (1), one TRQ bit is set to 1 in the ABT area (buffers 0 to 7). In addition to this TRQ bit, the application can request transmissions (set TRQ bit to 1) for other TX-message buffers that do not belong to the ABT area. In that case an internal arbitration process (TX-search) evaluates all of the TX-message buffers with the TRQ bit set to 1 and chooses the message buffer that contains the highest prioritized identifier for the next transmission. If there are 2 or more identifiers that have the highest priority (i.e. identical identifiers), the message located at the lowest message buffer number is transmitted first.

Upon successful transmission of a message frame, the following operations are performed.

- The TRQ bit of the corresponding transmit message buffer is automatically cleared to 0.
- The transmission completion status bit CINTS0 of the C0INTS register is set to 1 (if the interrupt enable bit (IE) of the corresponding transmit message buffer is set to 1).
- An interrupt request signal INTC0TRX is output (if the C0IE.CIE0 bit is set to 1 and if the interrupt enable bit (IE) of the corresponding transmit message buffer is set to 1).

2. Before changing the contents of the transmit message buffer, the RDY flag of this buffer must be cleared. Since the RDY flag may be temporarily locked while the internal processing is changed, it is necessary to check the status of the RDY flag after changing the buffer contents.

17.10.2 Transmit history list function

The transmit history list (THL) function records in the transmit history list the number of the transmit message buffer in which each data frame or remote frame was received and stored. The THL consists of storage elements equivalent to up to seven messages, the last out-message pointer (LOPT) with the corresponding C0LOPT register, and the transmit history list get pointer (TGPT) with the corresponding C0TGPT register.

The THL is undefined immediately after the transition of the CAN module from the initialization mode to one of the operation modes.

The C0LOPT register holds the contents of the THL element indicated by the value of the LOPT pointer minus 1. By reading the C0LOPT register, therefore, the number of the message buffer that transmitted a data frame or remote frame first can be checked. The LOPT pointer is utilized as a write pointer that indicates to what part of the THL a message buffer number is recorded. Any time a data frame or remote frame is transmitted, the corresponding message buffer number is recorded to the THL element indicated by the LOPT pointer. Each time recording to the THL has been completed, the LOPT pointer is automatically incremented. In this way, the number of the message buffer that has received and stored a frame will be recorded chronologically.

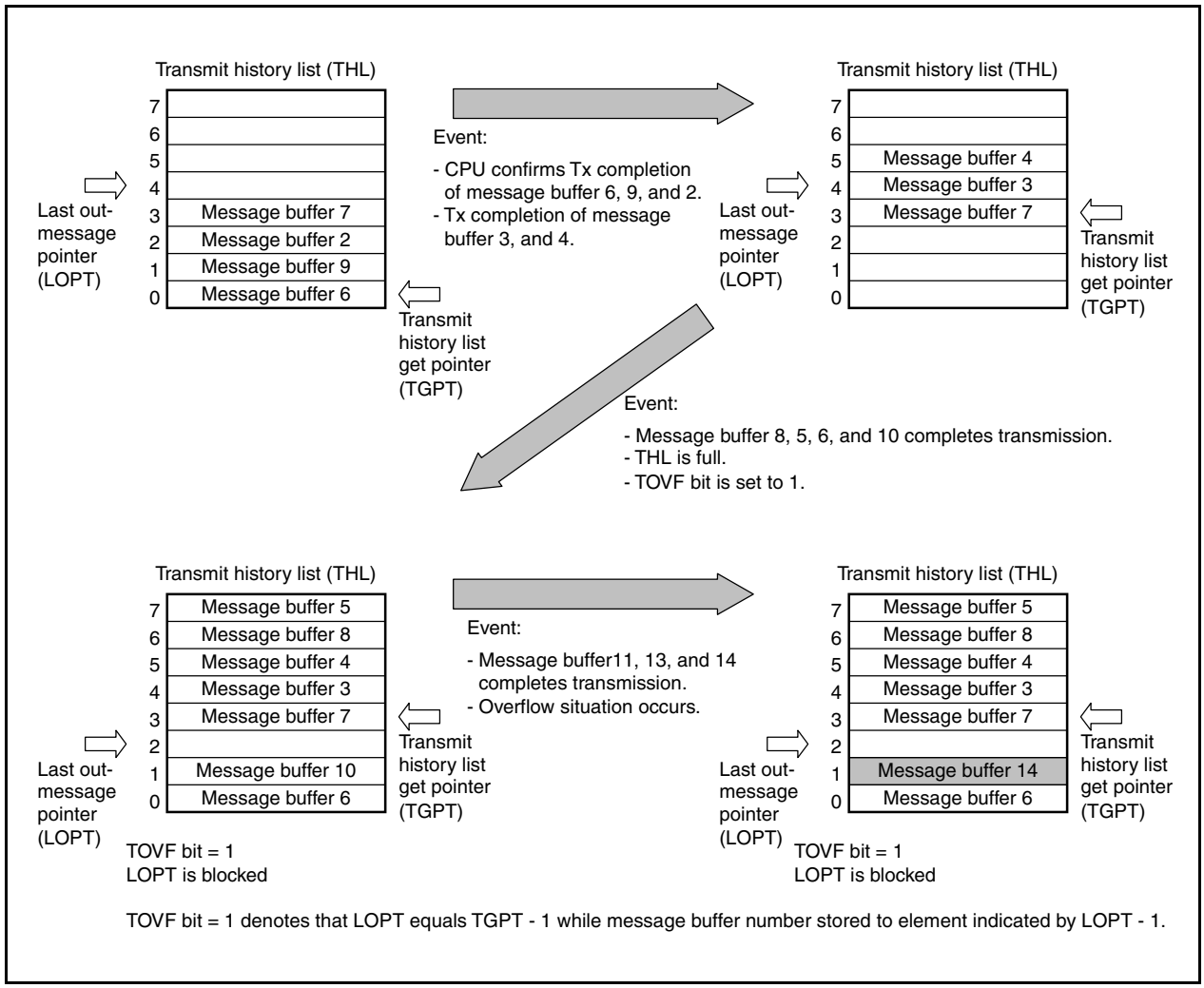
The TGPT pointer is utilized as a read pointer that reads a recorded message buffer number from the THL. This pointer indicates the first THL element that the CPU has not yet read. By reading the C0TGPT register by software, the number of a message buffer that has completed transmission can be read. Each time a message buffer number is read from the C0TGPT register, the TGPT pointer is automatically incremented.

If the value of the TGPT pointer matches the value of the LOPT pointer, the C0TGPT.THPM bit (transmit history list pointer match) is set to 1. This indicates that no message buffer numbers that have not been read remain in the THL. If a new message buffer number is recorded, the LOPT pointer is incremented and because its value no longer matches the value of the TGPT pointer, the THPM bit is cleared. In other words, the numbers of the unread message buffers exist in the THL.

If the LOPT pointer is incremented and matches the value of the TGPT pointer minus 1, the TOVF bit (transmit history list overflow) of the C0TGPT register is set to 1. This indicates that the THL is full of message buffer numbers that have not been read. If a new message is received and stored, the message buffer number recorded last is overwritten by the number of the message buffer that received and stored the new message. After the TOVF bit has been set (1), therefore, the recorded message buffer numbers in the THL do not completely reflect chronological order. However the transmitted messages can be found by a CPU search applied to all transmit message buffers unless the CPU has not overwritten a transmit object in one of these buffers beforehand. In total up to six transmission completions can occur without overflowing the THL.

Caution Even if the transmit history list overflows (C0TGPT.TOVF bit = 1), the transmit history can be read until no more history is left unread and the C0TGPT.THPM bit is set (1). However, the TOVF bit is kept set (1) (= overflow occurs) until cleared (0) by software. In this status, the THPM bit is not cleared (0), unless the TOVF bit is cleared (0), even if a new transmit history is stored and written to the list. If the TOVF bit = 1 and the THPM bit = 1 and the transmit history list overflows, therefore, the THPM bit indicates that no more history is left unread even if new transmit history is stored.

Figure 17-32. Transmit History List



TOVF bit = 1 denotes that LOPT equals TGPT - 1 while message buffer number stored to element indicated by LOPT - 1.

17.10.3 Automatic block transmission (ABT)

The automatic block transmission (ABT) function is used to transmit two or more data frames successively with no CPU interaction. The maximum number of transmit message buffers assigned to the ABT function is eight (message buffer numbers 0 to 7).

By setting the C0CTRL.OPMODE2 to C0CTRL.OPMODE0 bits to 010B, “normal operation mode with automatic block transmission function” (hereafter referred to as ABT mode) can be selected.

To issue an ABT transmission request, define the message buffers by software first. Set the C0MCONFm.MA0 bit (1) in all the message buffers used for ABT, and define all the buffers as transmit message buffers by setting the C0MCONFm.MA2 to C0MCONFm.MA0 bits to 000B. Be sure to set the ID used in the message buffer for ABT for each message buffer, even when that ID is being used for all the message buffers. To use two or more IDs, set the ID of each message buffer by using the C0MIDLm and C0MIDHm registers. Set the C0MDLCm and C0MDATA0m to C0MDATA7m registers before issuing a transmission request for the ABT function.

After initialization of message buffers for ABT is finished, the C0MCTRLm.RDY bit needs to be set (1). In the ABT mode, the C0MCTRLm.TRQ bit does not have to be manipulated by software.

After the data for the ABT message buffers has been prepared, set the C0GMABT.ABTTRG bit to 1. Automatic block transmission is then started. When ABT is started, the TRQ bit in the first message buffer (message buffer 0) is automatically set to 1. After transmission of the data of message buffer 0 is finished, the TRQ bit of the next message buffer, message buffer 1, is set automatically. In this way, transmission is executed successively.

A delay time can be inserted by program in the interval in which the transmission request (TRQ bit) is automatically set while successive transmission is being executed. The delay time to be inserted is defined by the C0GMABTD register. The unit of the delay time is DBT (data bit time). DBT depends on the setting of the C0BRP and C0BTR registers.

During ABT, the priority of the transmission ID is not searched in the ABT transmit message buffer. The data of message buffers 0 to 7 is sequentially transmitted. When transmission of the data frame from message buffer 7 has been completed, the ABTTRG bit is automatically cleared to 0 and the ABT operation is finished.

If the RDY bit of an ABT message buffer is cleared during ABT, no data frame is transmitted from that buffer, ABT is stopped, and the ABTTRG bit is cleared. After that, transmission can be resumed from the message buffer where ABT stopped, by setting the RDY and ABTTRG bits to 1 by software. To not resume transmission from the message buffer where ABT stopped, the internal ABT engine can be reset by setting the C0GMABT.ABTCLR bit to 1 while ABT mode is stopped and the ABTTRG bit is cleared to 0. In this case, transmission is started from message buffer 0 if the ABTCLR bit is cleared to 0 and then the ABTTRG bit is set to 1.

An interrupt can be used to check if data frames have been transmitted from all the message buffers for ABT. To do so, the C0MCTRLm.IE bit of each message buffer except the last message buffer needs to be cleared (0).

If a transmit message buffer other than those used by the ABT function (message buffers 8 to 31) is assigned to a transmit message buffer, the priority of the message to be transmitted is determined by the priority of the transmission ID of the ABT message buffer whose transmission is currently held pending and the transmission message buffer of the message buffers other than those used by the ABT function.

Transmission of a data frame from an ABT message buffer is not recorded in the transmit history list (THL).

Cautions 1. To resume the normal operation mode with ABT from the message buffer 0, set the ABTCLR bit to 1 while the ABTTRG bit is cleared to 0. If the ABTCLR bit is set to 1 while the ABTTRG bit is set to 1, the subsequent operation is not guaranteed.

2. Whether the automatic block transmission engine is cleared by setting the ABTCLR bit to 1 can be confirmed if the ABTCLR bit is automatically cleared immediately after the processing of the clearing request is completed.

3. Do not set the ABTTRG bit in the initialization mode. If the ABTTRG bit is set in the initialization mode, the proper operation is not guaranteed after the mode is changed from the initialization mode to the ABT mode.

4. Do not set the TRQ bit of the ABT message buffers to 1 by software in the normal operation mode with ABT. Otherwise, the operation is not guaranteed.

5. The C0GMABTD register is used to set the delay time that is inserted in the period from completion of the preceding ABT message to setting of the TRQ bit for the next ABT message when the transmission requests are set in the order of message numbers for each message for ABT that is successively transmitted in the ABT mode. The timing at which the messages are actually transmitted onto the CAN bus varies depending on the status of transmission from other stations and the status of the setting of the transmission request for messages other than the ABT messages (message buffers 8 to 31).

6. If a transmission request is made for a message other than an ABT message and if no delay time is inserted in the interval in which transmission requests for ABT are automatically set (C0GMABTD register = 00H), messages other than ABT messages may be transmitted regardless of their priority in regards to the ABT message.

7. Do not clear the RDY bit to 0 when the ABTTRG bit = 1.

8. If a message is received from another node in the normal operation mode with ABT, the message may be transmitted after the time of one frame has elapsed even when C0GMABTD register = 00H.

Remark m = 00 to 31

17.10.4 Transmission abort process

Remark m = 00 to 31

(1) Transmission abort process other than in normal operation mode with automatic block transmission (ABT)

The user can clear the C0MCTRLm.TRQ bit to 0 to abort a transmission request. The TRQ bit will be cleared immediately if the abort was successful. Whether the transmission was successfully aborted or not can be checked using the C0CTRL.TSTAT bit and the C0TGPT register, which indicate the transmission status on the CAN bus (for details, refer to the processing in **Figure 17-45**).

(2) Transmission abort process except for ABT transmission in normal operation mode with automatic block transmission (ABT)

The user can clear the C0GMABT.ABTTRG bit to 0 to abort a transmission request. After checking the ABTTRG bit of the C0GMABT register = 0, clear the C0MCTRLm.TRQ bit to 0. The TRQ bit will be cleared immediately if the abort was successful. Whether the transmission was successfully aborted or not can be checked by using the C0CTRL.TSTAT bit and the C0TGPT register, which indicate the transmission status on the CAN bus (for details, refer to the process in **Figure 17-46**).

(3) Transmission abort in normal operation mode with automatic block transmission (ABT)

To abort ABT that is already started, clear the C0GMABT.ABTTRG bit to 0. In this case, the ABTTRG bit remains 1 if an ABT message is currently being transmitted and until the transmission is completed (successfully or not), and is cleared to 0 as soon as transmission is finished. This aborts ABT.

If the last transmission (before ABT) was successful, the normal operation mode with ABT is left with the internal ABT pointer pointing to the next message buffer to be transmitted.

In the case of an erroneous transmission, the position of the internal ABT pointer depends on the status of the TRQ bit in the last transmitted message buffer. If the TRQ bit is set to 1 when clearing the ABTTRG bit is requested, the internal ABT pointer points to the last transmitted message buffer (for details, refer to the process in **Figure 17-47 (a)**). If the TRQ bit is cleared to 0 when clearing the ABTTRG bit is requested, the internal ABT pointer is increased in increments of 1 and indicates the next message buffer in the ABT area (for details, refer to the process in **Figure 17-47 (b)**).

Caution Be sure to abort ABT by clearing the ABTTRG bit to 0. The operation is not guaranteed if aborting transmission is requested by clearing RDY.

When the normal operation mode with ABT is resumed after ABT has been aborted and the ABTTRG bit is set to 1, the next ABT message buffer to be transmitted can be determined from the following table.

Status of TRQ of ABT Message Buffer	Abort After Successful Transmission	Abort After Erroneous Transmission
Set (1)	Next message buffer in the ABT area[Note]	Same message buffer in the ABT area
Cleared (0)	Next message buffer in the ABT area[Note]	Next message buffer in the ABT area[Note]

Note The above resumption operation can be performed only if a message buffer ready for ABT exists in the ABT area. For example, an abort request that is issued while ABT of message buffer 7 is in progress is regarded as completion of ABT, rather than abort, if transmission of message buffer 7 has been successfully completed, even if the ABTTRG bit is cleared to 0. If the C0MCTRLm.RDY bit in the next message buffer in the ABT area is cleared to 0, the internal ABT pointer is retained, but the resumption operation is not performed even if the ABTTRG bit is set to 1, and ABT ends immediately.

Remark m = 00 to 31

17.10.5 Remote frame transmission

Remote frames can be transmitted only from transmit message buffers. Set whether a data frame or remote frame is transmitted via the C0MCONFm.RTR bit. Setting (1) the RTR bit sets remote frame transmission.

Remark m = 00 to 31

17.11 Power Saving Modes

17.11.1 CAN sleep mode

The CAN sleep mode can be used to set the CAN controller to standby mode in order to reduce power consumption. The CAN module can enter the CAN sleep mode from all operation modes. Release of the CAN sleep mode returns the CAN module to exactly the same operation mode from which the CAN sleep mode was entered.

In the CAN sleep mode, the CAN module does not transmit messages, even when transmission requests are issued or pending.

(1) Entering CAN sleep mode

The CPU issues a CAN sleep mode transition request by writing 01B to the C0CTRL.PSMODE1 and C0CTRL.PSMODE0 bits.

This transition request is only acknowledged only under the following conditions.

(i) The CAN module is already in one of the following operation modes
- Normal operation mode
- Normal operation mode with ABT
- Receive-only mode
- Single-shot mode
- Self-test mode
- CAN stop mode in all the above operation modes

(ii) The CAN bus state is bus idle (the 4th bit in the interframe space is recessive)[Note]

Note If the CAN bus is fixed to dominant, the request for transition to the CAN sleep mode is held pending. Also the transition from CAN stop mode to CAN sleep mode is independent of the CAN bus state.

(iii) No transmission request is pending

If any one of the conditions mentioned above is not met, the CAN module will operate as follows.

- If the CAN sleep mode is requested from the initialization mode, the CAN sleep mode transition request is ignored and the CAN module remains in the initialization mode.
- If the CAN bus state is not bus idle (i.e., the CAN bus state is either transmitting or receiving) when the CAN sleep mode is requested in one of the operation modes, immediate transition to the CAN sleep mode is not possible. In this case, the CAN sleep mode transition request has to be held pending until the CAN bus state becomes bus idle (the 4th bit in the interframe space is recessive). In the time from the CAN sleep mode request to successful transition, the PSMODE1 and PSMODE0 bits remain 00B. When the module has entered the CAN sleep mode, the PSMODE1 and PSMODE0 bits are set to 01B.
- If a request for transition to the initialization mode and a request for transition to the CAN sleep mode are made at the same time while the CAN module is in one of the operation modes, the request for the initialization mode is enabled. The CAN module enters the initialization mode at a predetermined timing. At this time, the CAN sleep mode request is not held pending and is ignored.
- Even when the initialization mode and sleep mode are not requested simultaneously (i.e the first request was not granted when a second request was made), the request for initialization has priority over the CAN sleep mode request. The CAN sleep mode request is cancelled when the initialization mode is requested. When a pending request for the initialization mode is present, a subsequent request for the CAN sleep mode request is cancelled right at the point in time when it was submitted.

(2) Status in CAN sleep mode

The CAN module is in one of the following states after it enters the CAN sleep mode.

- The internal operating clock is stopped and the power consumption is minimized.
- The function to detect the falling edge of the CAN reception pin (CRXD0) remains in effect to wake up the CAN module from the CAN bus.
- To wake up the CAN module from the CPU, data can be written to the PSMODE1 and PSMODE0 bits, but nothing can be written to other CAN0 module registers or bits.
- The CAN0 module registers can be read, except for the C0LIPT, C0RGPT, C0LOPT, and C0TGPT registers.
- The CAN0 message buffer registers cannot be written or read.
- The C0GMCTRL.MBON bit is cleared to 0.
- A request for transition to the initialization mode is not acknowledged and is ignored.

(3) Releasing CAN sleep mode

The CAN sleep mode is released by the following events.

- When the CPU writes 00B to the PSMODE1 and PSMODE0 bits
- A falling edge at the CAN reception pin (CRXD0) (i.e. the CAN bus level shifts from recessive to dominant)

Cautions 1. Even if the falling edge belongs to the SOF of a receive message, this message will not be received and stored. If the CPU has turned off the clock to the CAN while the CAN was in sleep mode, later on the CAN sleep mode will not be released and PSMODE1 and PSMODE0 bits will continue to be 01B unless the clock for the CAN is provided again. In addition to this, the receive message will not be received afterwards.

2. If the falling edge is detected on the CAN reception pin (CRXD0) while the CAN clock is supplied, the PSMODE0 bit must be cleared by software (for details, refer to the processing in Figure 17-52).

After releasing the sleep mode, the CAN module returns to the operation mode from which the CAN sleep mode was requested and the PSMODE1 and PSMODE0 bits are reset to 00B. If the CAN sleep mode is released by a change in the CAN bus state, the C0INTS.CINTS5 bit is set to 1, regardless of the C0IE.CIE bit. After the CAN module is released from the CAN sleep mode, it participates in the CAN bus again by automatically detecting 11 consecutive recessive-level bits on the CAN bus. After releasing the sleep mode and before accessing the message buffer by application again, confirm that C0GMCTRL.MBON bit = 1.

When a request for transition to the initialization mode is made while the CAN module is in the CAN sleep mode, that request is ignored; the CPU has to be released from sleep mode by software first before entering the initialization mode.

Caution When the CAN sleep mode is released by an event of the CAN bus, a wakeup interrupt occurs even if the event of the CAN bus occurs immediately after the mode has been changed to the sleep mode. Note that the interrupt can occur at any time.

17.11.2 CAN stop mode

The CAN stop mode can be used to set the CAN controller to standby mode to reduce power consumption. The CAN module can enter the CAN stop mode only from the CAN sleep mode. Release of the CAN stop mode puts the CAN module in the CAN sleep mode.

The CAN stop mode can only be released (shifting to CAN sleep mode) by writing 01B to the C0CTRL.PSMODE1 and C0CTRL.PSMODE0 bits and not by a change in the CAN bus state. No message is transmitted even when transmission requests are issued or pending.

(1) Entering CAN stop mode

A CAN stop mode transition request is issued by writing 11B to the PSMODE1 and PSMODE0 bits.

A CAN stop mode request is only acknowledged when the CAN module is in the CAN sleep mode. In all other modes, the request is ignored.

Caution To set the CAN module to the CAN stop mode, the module must be in the CAN sleep mode. To confirm that the module is in the sleep mode, check that the PSMODE1 and PSMODE0 bits = 01B, and then request the CAN stop mode. If a bus change occurs at the CAN reception pin (CRXD0) while this process is being performed, the CAN sleep mode is automatically released. In this case, the CAN stop mode transition request cannot be acknowledged (when the CAN clock is supplied, however, the PSMODE0 bit must be cleared by software after a bus change occurs at the CAN reception pin (CRXD0)).

(2) Status in CAN stop mode

The CAN module is in one of the following states after it enters the CAN stop mode.

- The internal operating clock is stopped and the power consumption is minimized.
- To wake up the CAN module from the CPU, data can be written to the PSMODE1 and PSMODE0 bits, but nothing can be written to other CAN0 module registers or bits.
- The CAN0 module registers can be read, except for the C0LIPT, C0RGPT, C0LOPT, and C0TGPT registers.
- The CAN0 message buffer registers cannot be written or read.
- The C0GMCTRL.MBON bit is cleared to 0.
- An initialization mode transition request is not acknowledged and is ignored.

(3) Releasing CAN stop mode

The CAN stop mode can only be released by writing 01B to the PSMODE1 and PSMODE0 bits. After releasing the CAN stop mode, the CAN module enters the CAN sleep mode.

When the initialization mode is requested while the CAN module is in the CAN stop mode, that request is ignored; the CPU has to release the stop mode and subsequently the CAN sleep mode before entering into initialization mode. It is impossible to enter another operation mode directly from the CAN stop mode without entering the CAN sleep mode, the request will be ignored.

17.11.3 Example of using power saving modes

In some application systems, it may be necessary to place the CPU in a power saving mode to reduce the power consumption. By using the power saving mode specific to the CAN module and the power saving mode specific to the CPU in combination, the CPU can be woken up from the power saving status by the CAN bus.

Here is an example of using the power saving modes.

First, put the CAN module in the CAN sleep mode (PSMODE1, PSMODE0 bits = 01B). Next, put the CPU in the power saving mode. If an edge transition from recessive to dominant is detected at the CRXD0 signal in this status, the CINTS5 bit in the CAN module is set to 1. If the C0CTRL.CIE5 bit is set to 1, a wakeup interrupt (INTC0WUP) is generated. The CAN module is automatically released from the CAN sleep mode (PSMODE1, PSMODE0 bits = 00B) and returns to normal operation mode (when the CAN clock is supplied, however, the PSMODE0 bit must be cleared by software after a bus change occurs at the CAN reception pin (CRXD0)). The CPU, in response to INTC0WUP, can release its own power saving mode and return to normal operation mode.

To further reduce the power consumption of the CPU, the internal clocks, including that of the CAN module, may be tuned off. In this case, the operating clock supplied to the CAN module is turned off after the CAN module is put in the CAN sleep mode. Then the CPU enters a power saving mode in which the clock supplied to the CPU is turned off. If an edge transition from recessive to dominant is detected at the CRXD0 signal in this status, the CAN module can set the CINTS5 bit to 1 and generate a wakeup interrupt (INTC0WUP) even if it is not supplied with a clock. The other functions, however, do not operate because the clock supply to the CAN module is shut off, and the module remains in the CAN sleep mode. The CPU, in response to INTC0WUP, releases its power saving mode, resumes supply of the internal clocks, including the clock to the CAN module, after oscillation stabilization time has elapsed, and starts instruction execution. The CAN module is immediately released from the CAN sleep mode when the clock supply is resumed, and returns to normal operation mode (PSMODE1, PSMODE0 bits = 00B).

17.12 Interrupt Function

The CAN module provides 6 different interrupt sources.

The occurrence of these interrupt sources is stored in interrupt status registers. Four separate interrupt request signals are generated from the six interrupt sources. When an interrupt request signal that corresponds to two or more interrupt sources is generated, the interrupt sources can be identified by using an interrupt status register. After an interrupt source has occurred, the corresponding interrupt status bit must be cleared to 0 by software.

Table 17-20. List of CAN Module Interrupt Sources

No.	Interrupt Status Bit		Interrupt Enable Bit		Interrupt Request Signal	Interrupt Source Description
	Name	Register	Name	Register		
1	CINTS0[Note 1]	C0INTS	CIE0[Note 1]	C0IE	INTC0TRX	Message frame successfully transmitted from message buffer m
2	CINTS1[Note 1]	C0INTS	CIE1[Note 1]	C0IE	INTC0REC	Valid message frame reception in message buffer m
3	CINTS2	C0INTS	CIE2	C0IE	INTC0ERR	CAN module error state interrupt[Note 2]
4	CINTS3	C0INTS	CIE3	C0IE		CAN module protocol error interrupt[Note 3]
5	CINTS4	C0INTS	CIE4	C0IE		CAN module arbitration loss interrupt
6	CINTS5	C0INTS	CIE5	C0IE	INTC0WUP	CAN module wakeup interrupt from CAN sleep mode[Note 4]

Notes 1. The C0MCTRL.IE bit (message buffer interrupt enable bit) of the corresponding message buffer has to be set to 1 for that message buffer to participate in the interrupt generation process.

2. This interrupt is generated when the transmission/reception error counter is at the warning level, or in the error passive or bus-off state.

3. This interrupt is generated when a stuff error, form error, ACK error, bit error, or CRC error occurs.

4. This interrupt is generated when the CAN module is woken up from the CAN sleep mode because a falling edge is detected at the CAN reception pin (CAN bus transition from recessive to dominant).

Remark m = 00 to 31

17.13 Diagnosis Functions and Special Operational Modes

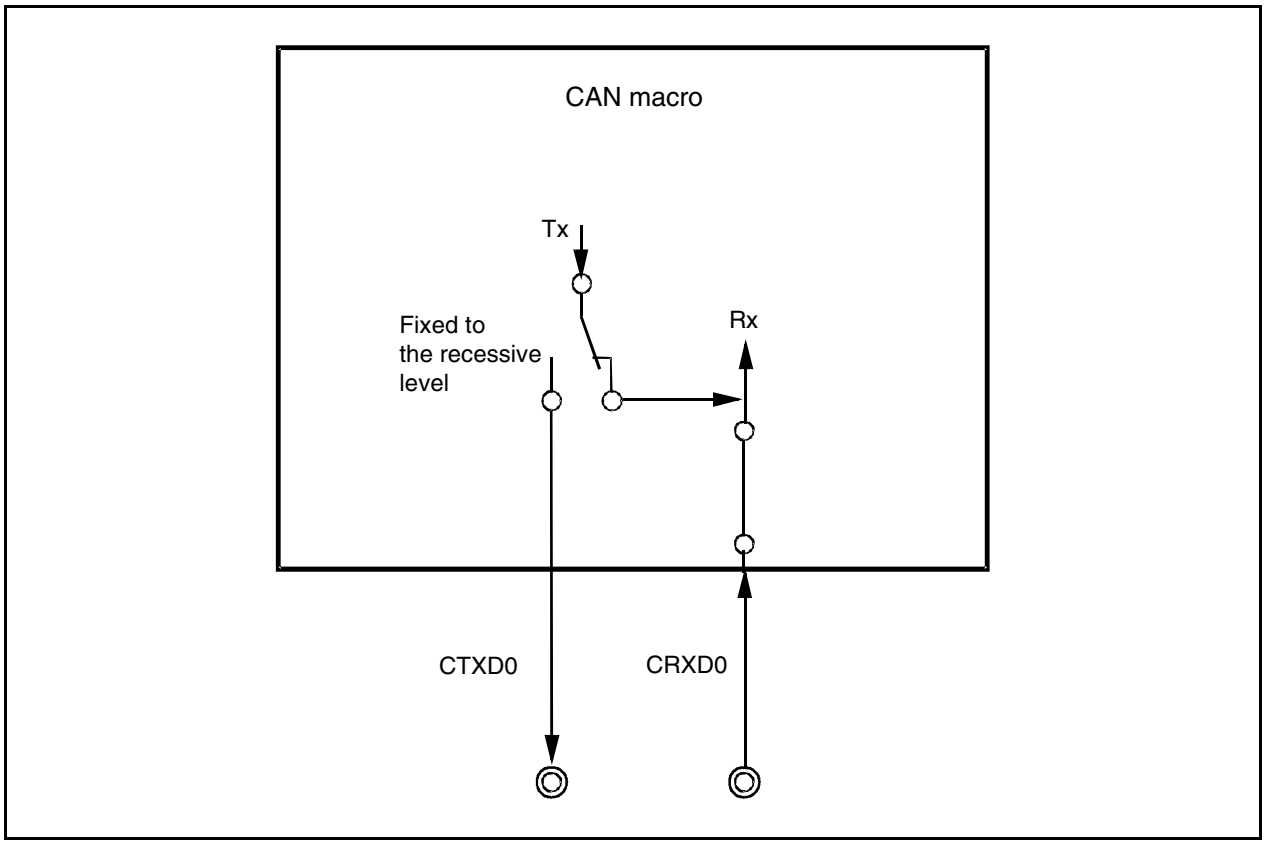
The CAN module provides a receive-only mode, single-shot mode, and self-test mode to support CAN bus diagnosis functions or the operation of special CAN communication methods.

17.13.1 Receive-only mode

The receive-only mode is used to monitor receive messages without causing any interference on the CAN bus and can be used for CAN bus analysis nodes.

For example, this mode can be used for automatic baud-rate detection. The baud rate in the CAN module is changed until “valid reception” is detected, so that the baud rates in the module match (“valid reception” means a message frame has been received in the CAN protocol layer without occurrence of an error and with an appropriate ACK between nodes connected to the CAN bus). A valid reception does not require message frames to be stored in a receive message buffer (data frames) or transmit message buffer (remote frames). The event of valid reception is indicated by setting the C0CTRL.VALID bit (1).

Figure 17-33. CAN Module Terminal Connection in Receive-Only Mode



In the receive-only mode, no message frames can be transmitted from the CAN module to the CAN bus. Transmit requests issued for message buffers defined as transmit message buffers are held pending.

In the receive-only mode, the CAN transmission pin (CTXD0) in the CAN module is fixed to the recessive level. Therefore, no active error flag can be transmitted from the CAN module to the CAN bus even when a CAN bus error is detected while receiving a message frame. Since no transmission can be issued from the CAN module, the transmission error counter the C0ERC.TEC7 to C0ERC.TEC0 bits are never updated. Therefore, a CAN module in the receive-only mode does not enter the bus-off state.

Furthermore, ACK is not returned to the CAN bus in this mode upon the valid reception of a message frame. Internally, the local node recognizes that it has transmitted ACK. An overload frame cannot be transmitted to the CAN bus.

Caution If only two CAN nodes are connected to the CAN bus and one of them is operating in the receive-only mode, there is no ACK on the CAN bus. Due to the missing ACK, the transmitting node will transmit an active error flag, and repeat transmitting a message frame. The transmitting node becomes error passive after transmitting the message frame 16 times (assuming that the error counter was 0 in the beginning and no other errors have occurred). When the message frame is transmitted for the 17th time, the transmitting node generates a passive error flag. The receiving node in the receive-only mode detects the first valid message frame at this point, and the VALID bit is set to 1 for the first time.

17.13.2 Single-shot mode

In the single-shot mode, automatic re-transmission as defined in the CAN protocol is switched off. (According to the CAN protocol, a message frame transmission that has been aborted by either arbitration loss or error occurrence has to be repeated without control by software.) All other behavior of single shot mode is identical to normal operation mode. Features of single shot mode can not be used in combination with normal mode with ABT.

The single-shot mode disables the re-transmission of an aborted message frame transmission according to the setting of the C0CTRL.AL bit. When the AL bit is cleared to 0, re-transmission upon arbitration loss and upon error occurrence is disabled. If the AL bit is set to 1, re-transmission upon error occurrence is disabled, but re-transmission upon arbitration loss is enabled. As a consequence, the C0MCTRLm.TRQ bit in a message buffer defined as a transmit message buffer is cleared to 0 by the following events.

- Successful transmission of the message frame
- Arbitration loss while sending the message frame (AL bit = 0)
- Error occurrence while sending the message frame

The events arbitration loss and error occurrence can be distinguished by checking the C0INTS.CINTS4 and C0INTS.CINTS3 bits, and the type of the error can be identified by reading the C0LEC.LEC2 to C0LEC.LEC0 bits of the register.

Upon successful transmission of the message frame, the transmit completion interrupt the CINTS0 bit of the C0INTS register is set to 1. If the C0IE.CIE0 bit is set to 1 at this time, an interrupt request signal is output.

The single-shot mode can be used when emulating time-triggered communication methods (e.g., TTCAN level 1).

Caution The AL bit is only valid in single-shot mode. It does not affect the operation of re-transmission upon arbitration loss in other operation modes.

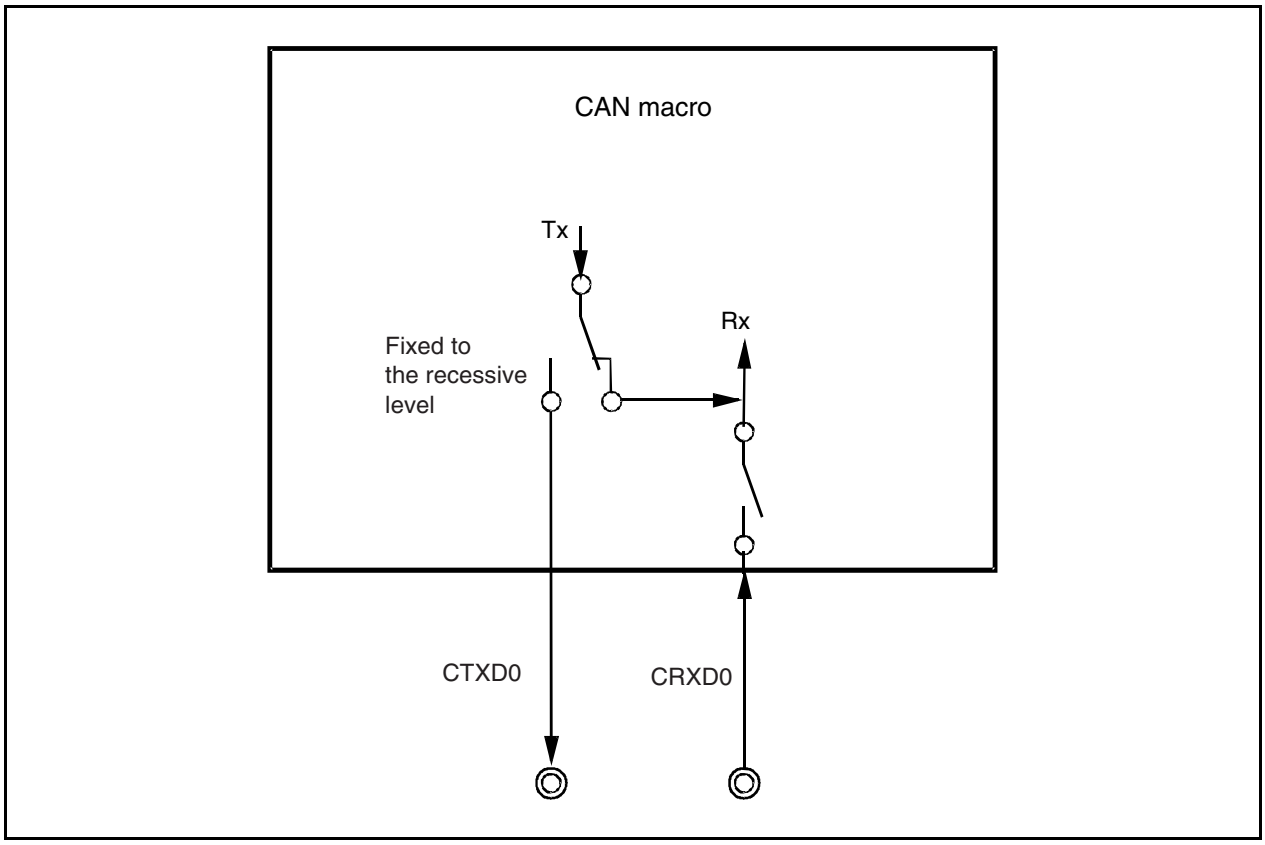
17.13.3 Self-test mode

In the self-test mode, message frame transmission and message frame reception can be tested without connecting the CAN node to the CAN bus or without affecting the CAN bus.

In the self-test mode, the CAN module is completely disconnected from the CAN bus, but transmission and reception are internally looped back. The CAN transmission pin (CTXD0) is fixed to the recessive level.

If the falling edge on the CAN reception pin (CRXD0) is detected after the CAN module has entered the CAN sleep mode from the self-test mode, however, the module is released from the CAN sleep mode in the same manner as the other operation modes (when the CAN clock is supplied, however, the PSMODE0 bit must be cleared by software after a bus change occurs at the CAN reception pin (CRXD0)). To keep the module in the CAN sleep mode, use the CAN reception pin (CRXD0) as a port pin.

Figure 17-34. CAN Module Terminal Connection in Self-Test Mode



17.13.4 Transmission/reception operation in each operation mode

Table 17-21 shows the transmission/reception operation in each operation mode.

Table 17-21. Overview of Transmission/Reception Operation in Each Operation Mode

Operation Mode	Data Frame/ Remote Frame Transmission	ACK Transmission	Error Frame/ Overload Frame Transmission	Retransmission	Automatic Block Transmission (ABT)	Setting of VALID Bit	Storing Data in Message Buffer
Initialization Mode	–	–	–	–	–	–	–
Normal operation mode	√	√	√	√	–	√	√
Normal operation mode with ABT	√	√	√	√	√	√	√
Receive-only mode	–	–	–	–	–	√	√
Single-shot mode	√	√	√	–Note 1	–	√	√
Self test mode	√Note 2	√Note 2	√Note 2	√Note 2	–	√Note 2	√Note 2

Notes 1. If arbitration is lost, retransmission can be selected by the C0CTRL.AL bit.

2. Each signal is not output to the external circuit but is internally generated by the CAN module.

17.14 Baud Rate Settings

17.14.1 Bit rate setting conditions

Make sure that the settings are within the range of limit values for ensuring correct operation of the CAN controller, as follows.

(a) 5TQ ≤ SPT (sampling point) ≤ 17TQ
SPT = TSEG1 + 1TQ

(b) 8TQ ≤ DBT (data bit time) ≤ 25TQ
DBT = TSEG1 + TSEG2 + 1TQ = TSEG2 + SPT

(c) 1TQ ≤ SJW (synchronization jump width) ≤ 4TQ
SJW ≤ DBT – SPT

(d) 4TQ ≤ TSEG1 ≤ 16TQ [3 ≤ Setting value of TSEG1[3:0] ≤ 15]

(e) 1TQ ≤ TSEG2 ≤ 8TQ [0 ≤ Setting value of TSEG2[2:0] ≤ 7]

Remark TQ = 1/f_{TQ} (f_{TQ}: CAN protocol layer base system clock)
TSEG1[3:0] (C0BTR.TSEG13 to C0BTR.TSEG10 bits)
TSEG2[2:0] (C0BTR.TSEG22 to C0BTR.TSEG20 bits)

Table 17-22 shows the combinations of bit rates that satisfy the above conditions.

Table 17-22. Settable Bit Rate Combinations (1/3)

Valid Bit Rate Setting					C0BTR Register Setting Value		Sampling Point (Unit %)
DBT Length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	TSEG13 to TSEG10	TSEG22 to TSEG20	
25	1	8	8	8	1111	111	68.0
24	1	7	8	8	1110	111	66.7
24	1	9	7	7	1111	110	70.8
23	1	6	8	8	1101	111	65.2
23	1	8	7	7	1110	110	69.6
23	1	10	6	6	1111	101	73.9
22	1	5	8	8	1100	111	63.6
22	1	7	7	7	1101	110	68.2
22	1	9	6	6	1110	101	72.7
22	1	11	5	5	1111	100	77.3
21	1	4	8	8	1011	111	61.9
21	1	6	7	7	1100	110	66.7
21	1	8	6	6	1101	101	71.4
21	1	10	5	5	1110	100	76.2
21	1	12	4	4	1111	011	81.0
20	1	3	8	8	1010	111	60.0
20	1	5	7	7	1011	110	65.0
20	1	7	6	6	1100	101	70.0
20	1	9	5	5	1101	100	75.0
20	1	11	4	4	1110	011	80.0
20	1	13	3	3	1111	010	85.0
19	1	2	8	8	1001	111	57.9
19	1	4	7	7	1010	110	63.2
19	1	6	6	6	1011	101	68.4
19	1	8	5	5	1100	100	73.7
19	1	10	4	4	1101	011	78.9
19	1	12	3	3	1110	010	84.2
19	1	14	2	2	1111	001	89.5
18	1	1	8	8	1000	111	55.6
18	1	3	7	7	1001	110	61.1
18	1	5	6	6	1010	101	66.7
18	1	7	5	5	1011	100	72.2
18	1	9	4	4	1100	011	77.8
18	1	11	3	3	1101	010	83.3
18	1	13	2	2	1110	001	88.9
18	1	15	1	1	1111	000	94.4

Table 17-22. Settable Bit Rate Combinations (2/3)

Valid Bit Rate Setting					C0BTR Register Setting Value		Sampling Point (Unit %)
DBT Length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	TSEG13 to TSEG10	TSEG22 to TSEG20	
17	1	2	7	7	1000	110	58.8
17	1	4	6	6	1001	101	64.7
17	1	6	5	5	1010	100	70.6
17	1	8	4	4	1011	011	76.5
17	1	10	3	3	1100	010	82.4
17	1	12	2	2	1101	001	88.2
17	1	14	1	1	1110	000	94.1
16	1	1	7	7	0111	110	56.3
16	1	3	6	6	1000	101	62.5
16	1	5	5	5	1001	100	68.8
16	1	7	4	4	1010	011	75.0
16	1	9	3	3	1011	010	81.3
16	1	11	2	2	1100	001	87.5
16	1	13	1	1	1101	000	93.8
15	1	2	6	6	0111	101	60.0
15	1	4	5	5	1000	100	66.7
15	1	6	4	4	1001	011	73.3
15	1	8	3	3	1010	010	80.0
15	1	10	2	2	1011	001	86.7
15	1	12	1	1	1100	000	93.3
14	1	1	6	6	0110	101	57.1
14	1	3	5	5	0111	100	64.3
14	1	5	4	4	1000	011	71.4
14	1	7	3	3	1001	010	78.6
14	1	9	2	2	1010	001	85.7
14	1	11	1	1	1011	000	92.9
13	1	2	5	5	0110	100	61.5
13	1	4	4	4	0111	011	69.2
13	1	6	3	3	1000	010	76.9
13	1	8	2	2	1001	001	84.6
13	1	10	1	1	1010	000	92.3
12	1	1	5	5	0101	100	58.3
12	1	3	4	4	0110	011	66.7
12	1	5	3	3	0111	010	75.0
12	1	7	2	2	1000	001	83.3
12	1	9	1	1	1001	000	91.7

Table 17-22. Settable Bit Rate Combinations (3/3)

Valid Bit Rate Setting					C0BTR Register Setting Value		Sampling Point (Unit %)
DBT Length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	TSEG13 to TSEG10	TSEG22 to TSEG20	
11	1	2	4	4	0101	011	63.6
11	1	4	3	3	0110	010	72.7
11	1	6	2	2	0111	001	81.8
11	1	8	1	1	1000	000	90.9
10	1	1	4	4	0100	011	60.0
10	1	3	3	3	0101	010	70.0
10	1	5	2	2	0110	001	80.0
10	1	7	1	1	0111	000	90.0
9	1	2	3	3	0100	010	66.7
9	1	4	2	2	0101	001	77.8
9	1	6	1	1	0110	000	88.9
8	1	1	3	3	0011	010	62.5
8	1	3	2	2	0100	001	75.0
8	1	5	1	1	0101	000	87.5
7[Note]	1	2	2	2	0011	001	71.4
7[Note]	1	4	1	1	0100	000	85.7
6[Note]	1	1	2	2	0010	001	66.7
6[Note]	1	3	1	1	0011	000	83.3
5[Note]	1	2	1	1	0010	000	80.0
4[Note]	1	1	1	1	0001	000	75.0

Note Setting with a DBT value of 7 or less is valid only when the value of the C0BRP register is other than 00H.

Caution The values in Table 17-22 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

17.14.2 Representative examples of baud rate settings

Tables 17-23 and 17-24 show representative examples of baud rate settings.

Table 17-23. Representative Examples of Baud Rate Settings (f_{CANMOD} = 8 MHz) (1/2)

Set Baud Rate Value (Unit: kbps)	Division Ratio of C0BRP Register	C0BRP Register Set Value	Valid Bit Rate Setting (Unit: kbps)					C0BTR Register Setting Value		Sampling Point (Unit: %)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	TSEG13 to TSEG10	TSEG22 to TSEG20	
1000	1	00000000	8	1	1	3	3	0011	010	62.5
1000	1	00000000	8	1	3	2	2	0100	001	75.0
1000	1	00000000	8	1	5	1	1	0101	000	87.5
500	1	00000000	16	1	1	7	7	0111	110	56.3
500	1	00000000	16	1	3	6	6	1000	101	62.5
500	1	00000000	16	1	5	5	5	1001	100	68.8
500	1	00000000	16	1	7	4	4	1010	011	75.0
500	1	00000000	16	1	9	3	3	1011	010	81.3
500	1	00000000	16	1	11	2	2	1100	001	87.5
500	1	00000000	16	1	13	1	1	1101	000	93.8
500	2	00000001	8	1	1	3	3	0011	010	62.5
500	2	00000001	8	1	3	2	2	0100	001	75.0
500	2	00000001	8	1	5	1	1	0101	000	87.5
250	2	00000001	16	1	1	7	7	0111	110	56.3
250	2	00000001	16	1	3	6	6	1000	101	62.5
250	2	00000001	16	1	5	5	5	1001	100	68.8
250	2	00000001	16	1	7	4	4	1010	011	75.0
250	2	00000001	16	1	9	3	3	1011	010	81.3
250	2	00000001	16	1	11	2	2	1100	001	87.5
250	2	00000001	16	1	13	1	1	1101	000	93.8
250	4	00000011	8	1	3	2	2	0100	001	75.0
250	4	00000011	8	1	5	1	1	0101	000	87.5
125	4	00000011	16	1	1	7	7	0111	110	56.3
125	4	00000011	16	1	3	6	6	1000	101	62.5
125	4	00000011	16	1	5	5	5	1001	100	68.8
125	4	00000011	16	1	7	4	4	1010	011	75.0
125	4	00000011	16	1	9	3	3	1011	010	81.3
125	4	00000011	16	1	11	2	2	1100	001	87.5
125	4	00000011	16	1	13	1	1	1101	000	93.8
125	8	00000111	8	1	3	2	2	0100	001	75.0
125	8	00000111	8	1	5	1	1	0101	000	87.5

Caution The values in Table 17-23 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

Table 17-23. Representative Examples of Baud Rate Settings (f_{CANMOD} = 8 MHz) (2/2)

Set Baud Rate Value (Unit: kbps)	Division Ratio of C0BRP Register	C0BRP Register Set Value	Valid Bit Rate Setting (Unit: kbps)					C0BTR Register Setting Value		Sampling Point (Unit: %)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	TSEG13 to TSEG10	TSEG22 to TSEG20	
100	4	00000011	20	1	7	6	6	1100	101	70.0
100	4	00000011	20	1	9	5	5	1101	100	75.0
100	5	00000100	16	1	7	4	4	1010	011	75.0
100	5	00000100	16	1	9	3	3	1011	010	81.3
100	8	00000111	10	1	3	3	3	0101	010	70.0
100	8	00000111	10	1	5	2	2	0110	001	80.0
100	10	00001001	8	1	3	2	2	0100	001	75.0
100	10	00001001	8	1	5	1	1	0101	000	87.5
83.3	4	00000011	24	1	7	8	8	1110	111	66.7
83.3	4	00000011	24	1	9	7	7	1111	110	70.8
83.3	6	00000101	16	1	5	5	5	1001	100	68.8
83.3	6	00000101	16	1	7	4	4	1010	011	75.0
83.3	6	00000101	16	1	9	3	3	1011	010	81.3
83.3	6	00000101	16	1	11	2	2	1100	001	87.5
83.3	8	00000111	12	1	5	3	3	0111	010	75.0
83.3	8	00000111	12	1	7	2	2	1000	001	83.3
83.3	12	00001011	8	1	3	2	2	0100	001	75.0
83.3	12	00001011	8	1	5	1	1	0101	000	87.5
33.3	10	00001001	24	1	7	8	8	1110	111	66.7
33.3	10	00001001	24	1	9	7	7	1111	110	70.8
33.3	12	00001011	20	1	7	6	6	1100	101	70.0
33.3	12	00001011	20	1	9	5	5	1101	100	75.0
33.3	15	00001110	16	1	7	4	4	1010	011	75.0
33.3	15	00001110	16	1	9	3	3	1011	010	81.3
33.3	16	00001111	15	1	6	4	4	1001	011	73.3
33.3	16	00001111	15	1	8	3	3	1010	010	80.0
33.3	20	00010011	12	1	5	3	3	0111	010	75.0
33.3	20	00010011	12	1	7	2	2	1000	001	83.3
33.3	24	00010111	10	1	3	3	3	0101	010	70.0
33.3	24	00010111	10	1	5	2	2	0110	001	80.0
33.3	30	00011101	8	1	3	2	2	0100	001	75.0
33.3	30	00011101	8	1	5	1	1	0101	000	87.5

Caution The values in Table 17-23 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

Table 17-24. Representative Examples of Baud Rate Settings (f_{CANMOD} = 16 MHz) (1/2)

Set Baud Rate Value (Unit: kbps)	Division Ratio of C0BRP Register	C0BRP Register Set Value	Valid Bit Rate Setting (Unit: kbps)					C0BTR Register Setting Value		Sampling Point (Unit: %)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	TSEG13 to TSEG10	TSEG22 to TSEG20	
1000	1	00000000	16	1	1	7	7	0111	110	56.3
1000	1	00000000	16	1	3	6	6	1000	101	62.5
1000	1	00000000	16	1	5	5	5	1001	100	68.8
1000	1	00000000	16	1	7	4	4	1010	011	75.0
1000	1	00000000	16	1	9	3	3	1011	010	81.3
1000	1	00000000	16	1	11	2	2	1100	001	87.5
1000	1	00000000	16	1	13	1	1	1101	000	93.8
1000	2	00000001	8	1	3	2	2	0100	001	75.0
1000	2	00000001	8	1	5	1	1	0101	000	87.5
500	2	00000001	16	1	1	7	7	0111	110	56.3
500	2	00000001	16	1	3	6	6	1000	101	62.5
500	2	00000001	16	1	5	5	5	1001	100	68.8
500	2	00000001	16	1	7	4	4	1010	011	75.0
500	2	00000001	16	1	9	3	3	1011	010	81.3
500	2	00000001	16	1	11	2	2	1100	001	87.5
500	2	00000001	16	1	13	1	1	1101	000	93.8
500	4	00000011	8	1	3	2	2	0100	001	75.0
500	4	00000011	8	1	5	1	1	0101	000	87.5
250	4	00000011	16	1	3	6	6	1000	101	62.5
250	4	00000011	16	1	5	5	5	1001	100	68.8
250	4	00000011	16	1	7	4	4	1010	011	75.0
250	4	00000011	16	1	9	3	3	1011	010	81.3
250	4	00000011	16	1	11	2	2	1100	001	87.5
250	8	00000111	8	1	3	2	2	0100	001	75.0
250	8	00000111	8	1	5	1	1	0101	000	87.5
125	8	00000111	16	1	3	6	6	1000	101	62.5
125	8	00000111	16	1	7	4	4	1010	011	75.0
125	8	00000111	16	1	9	3	3	1011	010	81.3
125	8	00000111	16	1	11	2	2	1100	001	87.5
125	16	00001111	8	1	3	2	2	0100	001	75.0
125	16	00001111	8	1	5	1	1	0101	000	87.5

Caution The values in Table 17-24 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

Table 17-24. Representative Examples of Baud Rate Settings (f_{CANMOD} = 16 MHz) (2/2)

Set Baud Rate Value (Unit: kbps)	Division Ratio of C0BRP Register	C0BRP Register Set Value	Valid Bit Rate Setting (Unit: kbps)					C0BTR Register Setting Value		Sampling Point (Unit: %)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	TSEG13 to TSEG10	TSEG22 to TSEG20	
100	8	00000111	20	1	9	5	5	1101	100	75.0
100	8	00000111	20	1	11	4	4	1110	011	80.0
100	10	00001001	16	1	7	4	4	1010	011	75.0
100	10	00001001	16	1	9	3	3	1011	010	81.3
100	16	00001111	10	1	3	3	3	0101	010	70.0
100	16	00001111	10	1	5	2	2	0110	001	80.0
100	20	00010011	8	1	3	2	2	0100	001	75.0
83.3	8	00000111	24	1	7	8	8	1110	111	66.7
83.3	8	00000111	24	1	9	7	7	1111	110	70.8
83.3	12	00001011	16	1	7	4	4	1010	011	75.0
83.3	12	00001011	16	1	9	3	3	1011	010	81.3
83.3	12	00001011	16	1	11	2	2	1100	001	87.5
83.3	16	00001111	12	1	5	3	3	0111	010	75.0
83.3	16	00001111	12	1	7	2	2	1000	001	83.3
83.3	24	00010111	8	1	3	2	2	0100	001	75.0
83.3	24	00010111	8	1	5	1	1	0101	000	87.5
33.3	30	00011101	24	1	7	8	8	1110	111	66.7
33.3	30	00011101	24	1	9	7	7	1111	110	70.8
33.3	24	00010111	20	1	9	5	5	1101	100	75.0
33.3	24	00010111	20	1	11	4	4	1110	011	80.0
33.3	30	00011101	16	1	7	4	4	1010	011	75.0
33.3	30	00011101	16	1	9	3	3	1011	010	81.3
33.3	32	00011111	15	1	8	3	3	1010	010	80.0
33.3	32	00011111	15	1	10	2	2	1011	001	86.7
33.3	37	00100100	13	1	6	3	3	1000	010	76.9
33.3	37	00100100	13	1	8	2	2	1001	001	84.6
33.3	40	00100111	12	1	5	3	3	0111	010	75.0
33.3	40	00100111	12	1	7	2	2	1000	001	83.3
33.3	48	00101111	10	1	3	3	3	0101	010	70.0
33.3	48	00101111	10	1	5	2	2	0110	001	80.0
33.3	60	00111011	8	1	3	2	2	0100	001	75.0
33.3	60	00111011	8	1	5	1	1	0101	000	87.5

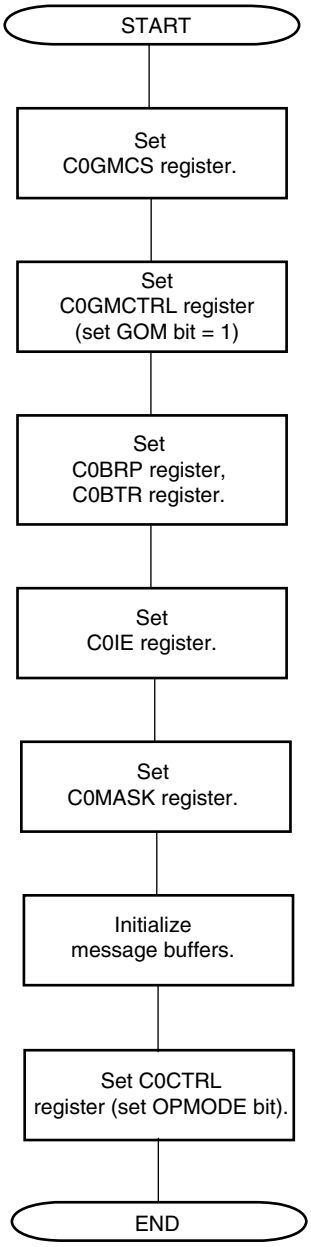
Caution The values in Table 17-24 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

17.15 Operation of CAN Controller

The processing procedure shown below is recommended to operate the CAN controller. Develop your program by referring to this recommended processing procedure.

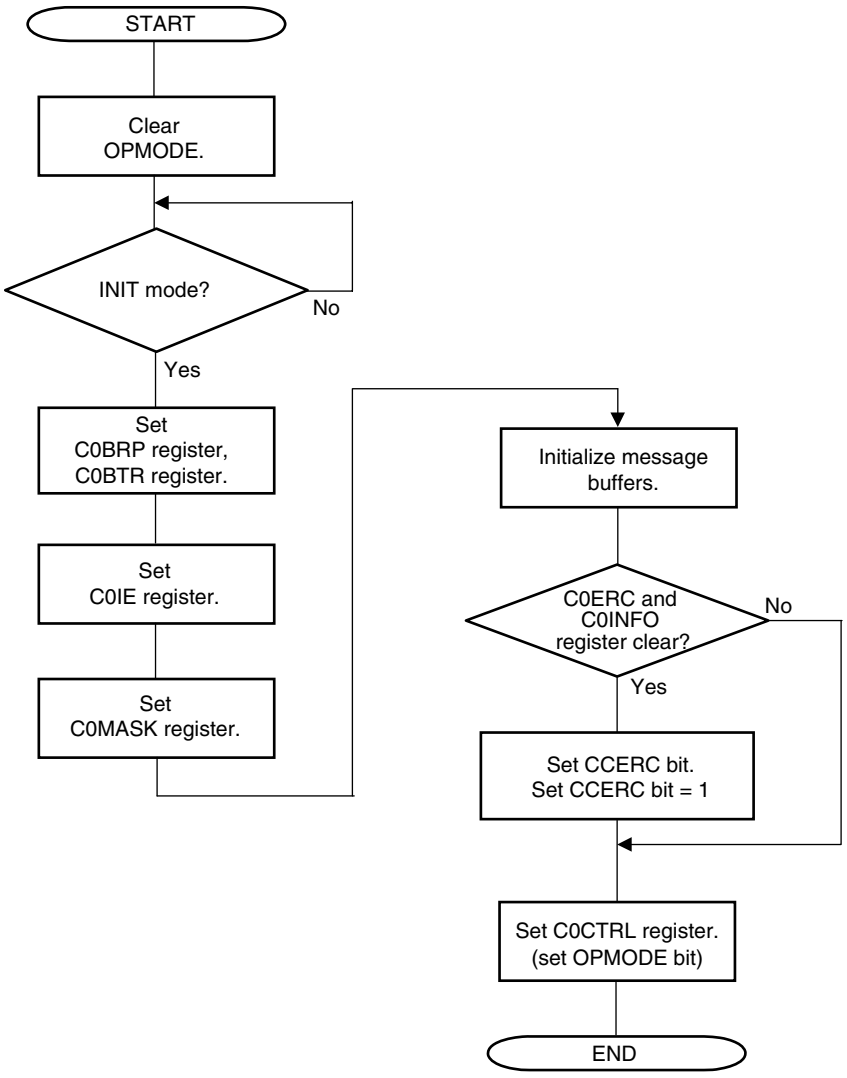
Remark m = 00 to 31

Figure 17-35. Initialization



Remark OPMODE: Normal operation mode, normal operation mode with ABT, receive-only mode, single-shot mode, self-test mode

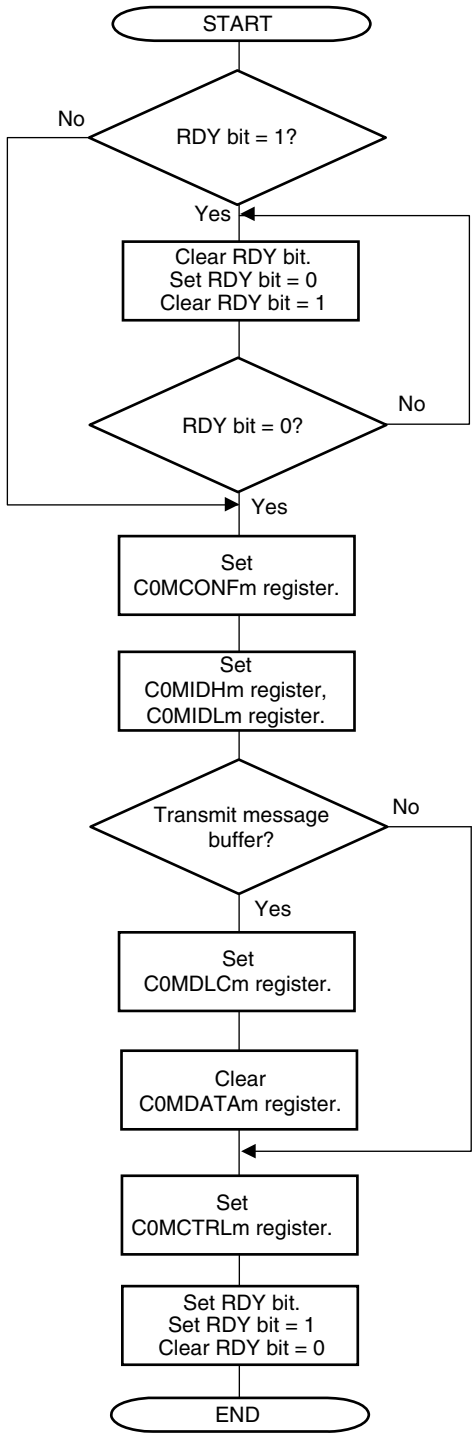
Figure 17-36. Re-initialization



Caution After setting the CAN module to the initialization mode, avoid setting the module to another operation mode immediately after. If it is necessary to immediately set the module to another operation mode, be sure to access registers other than the C0CTRL and C0GMCTRL registers (e.g., set a message buffer).

Remark OPMODE: Normal operation mode, normal operation mode with ABT, receive-only mode, single-shot mode, self-test mode

Figure 17-37. Message Buffer Initialization



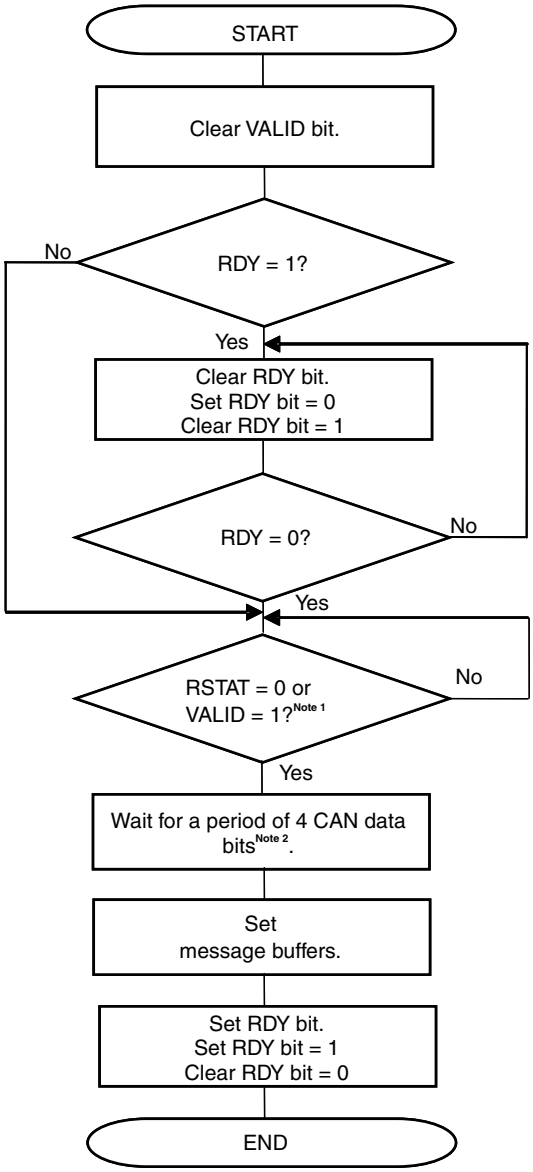
Cautions 1. Before a message buffer is initialized, the RDY bit must be cleared.

2. Make the following settings for message buffers not used by the application.

- **Clear the C0MCTRLm.RDY, C0MCTRLm.TRQ, and C0MCTRLm.DN bits to 0.**
- **Clear the C0MCONFm.MA0 bit to 0.**

Figure 17-38 shows the processing for a receive message buffer (C0MCONFm.MT2 to C0MCONFm.MT0 bits = 001B to 101B).

Figure 17-38. Message Buffer Redefinition



Notes 1. If redefinition is performed during a message reception, confirm that a message is being received because the RDY bit must be set after a message is completely received.

2. This 4-bit period may redefine the message buffer while a message is received and stored.

Figure 17-39 shows the processing for a transmit message buffer during transmission (MT2 to MT0 bits of C0MCONFm register = 000B).

Figure 17-39. Message Buffer Redefinition during Transmission

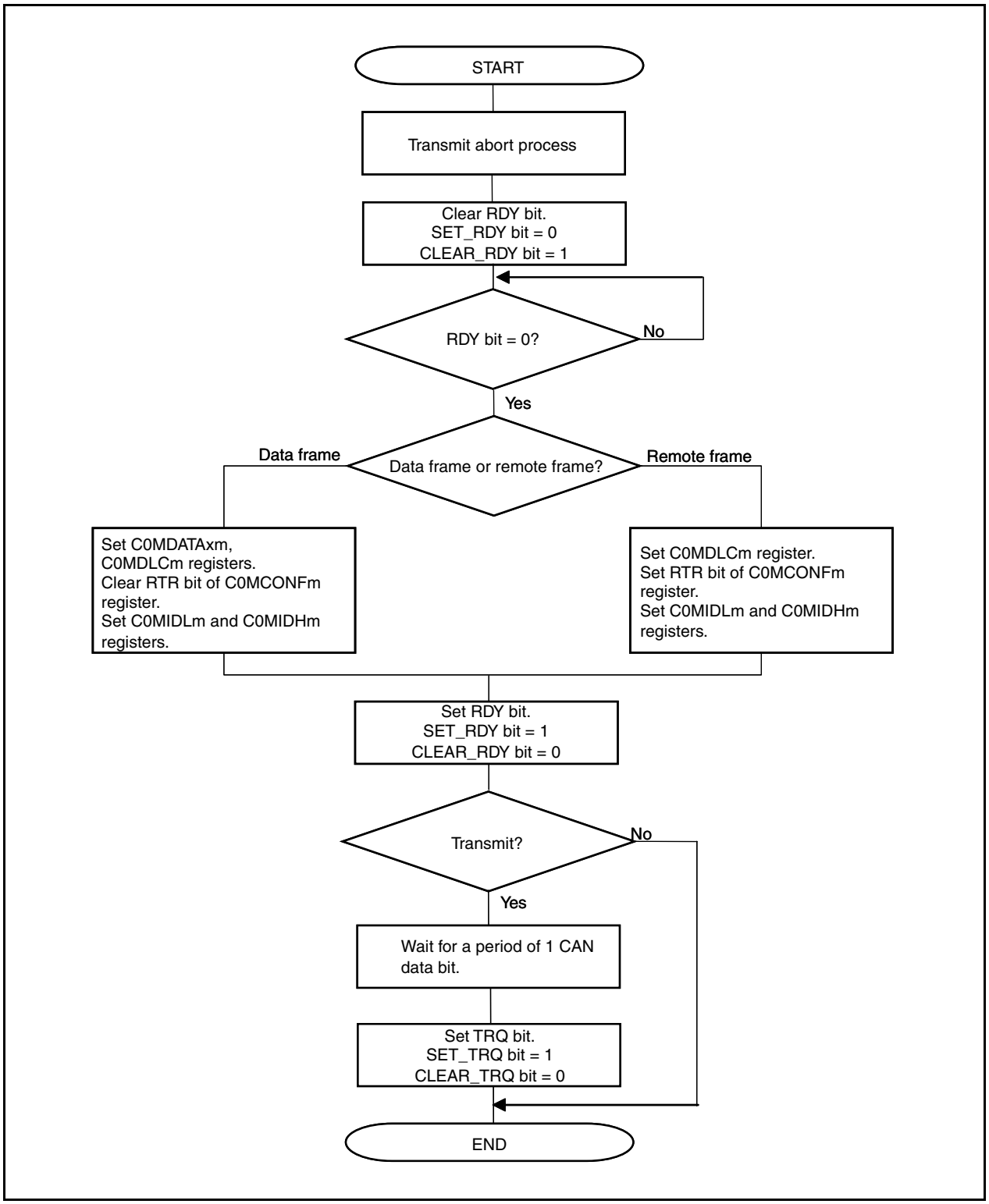
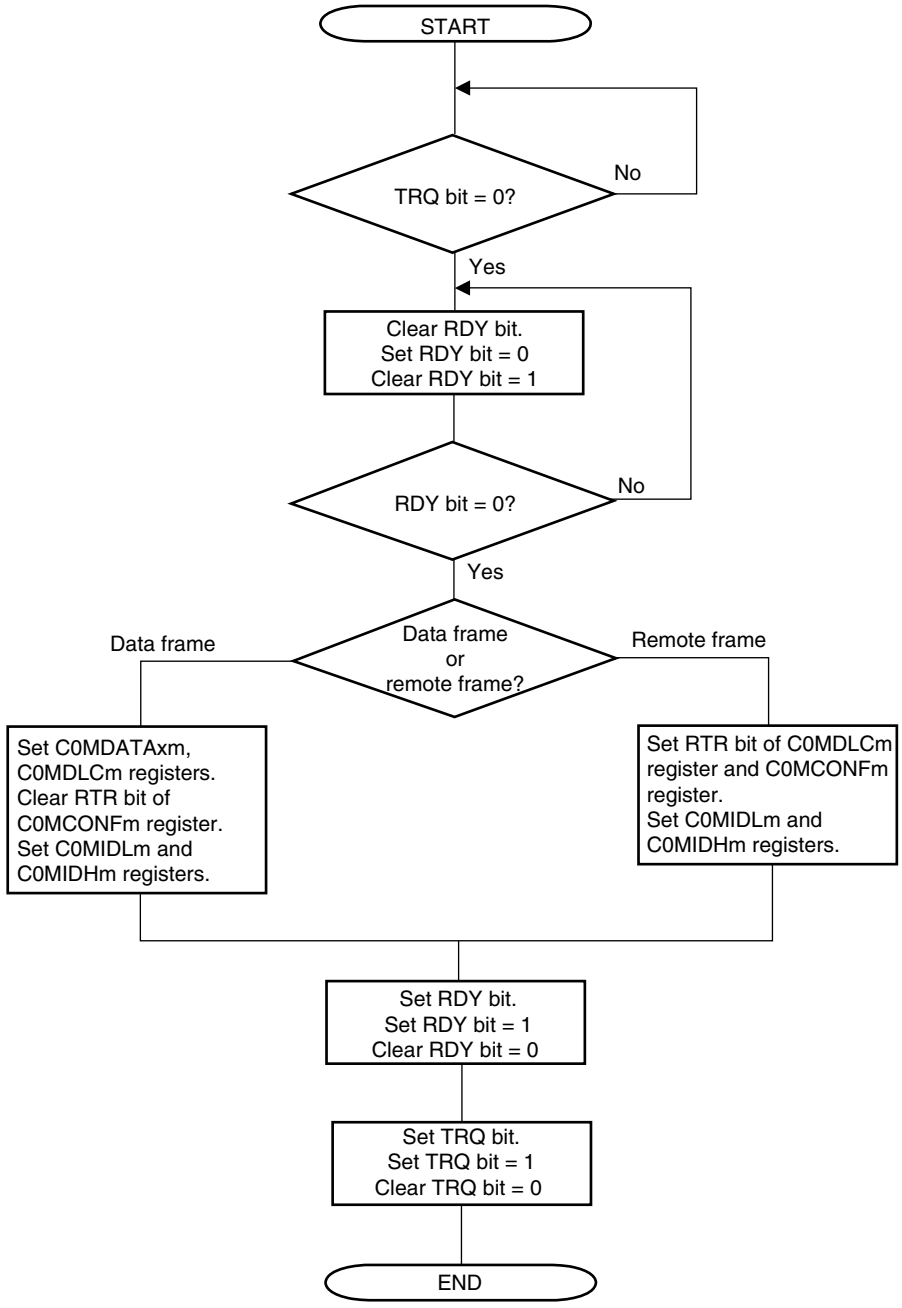


Figure 17-40 shows the processing for a transmit message buffer (C0MCONFm.MT2 to C0MCONFm.MT0 bits = 000B).

Figure 17-40. Message Transmit Processing

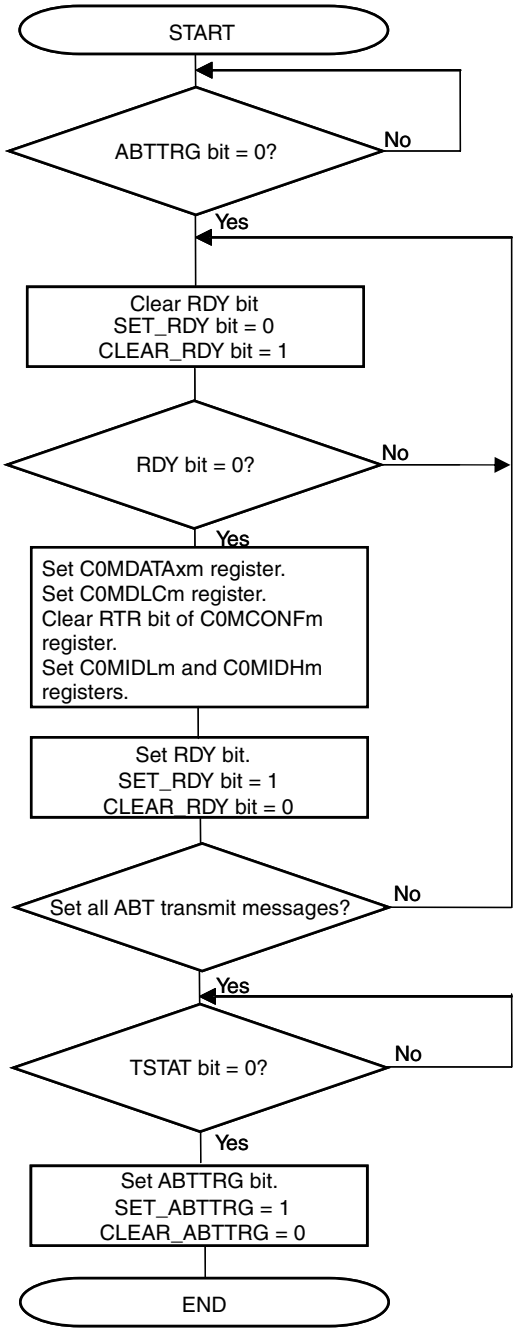


Cautions 1. The TRQ bit should be set after the RDY bit is set.

2. The RDY bit and TRQ bit should not be set at the same time.

Figure 17-41 shows the processing for a transmit message buffer (C0MCONFm.MT2 to C0MCONFm.MT0 bits = 000B).

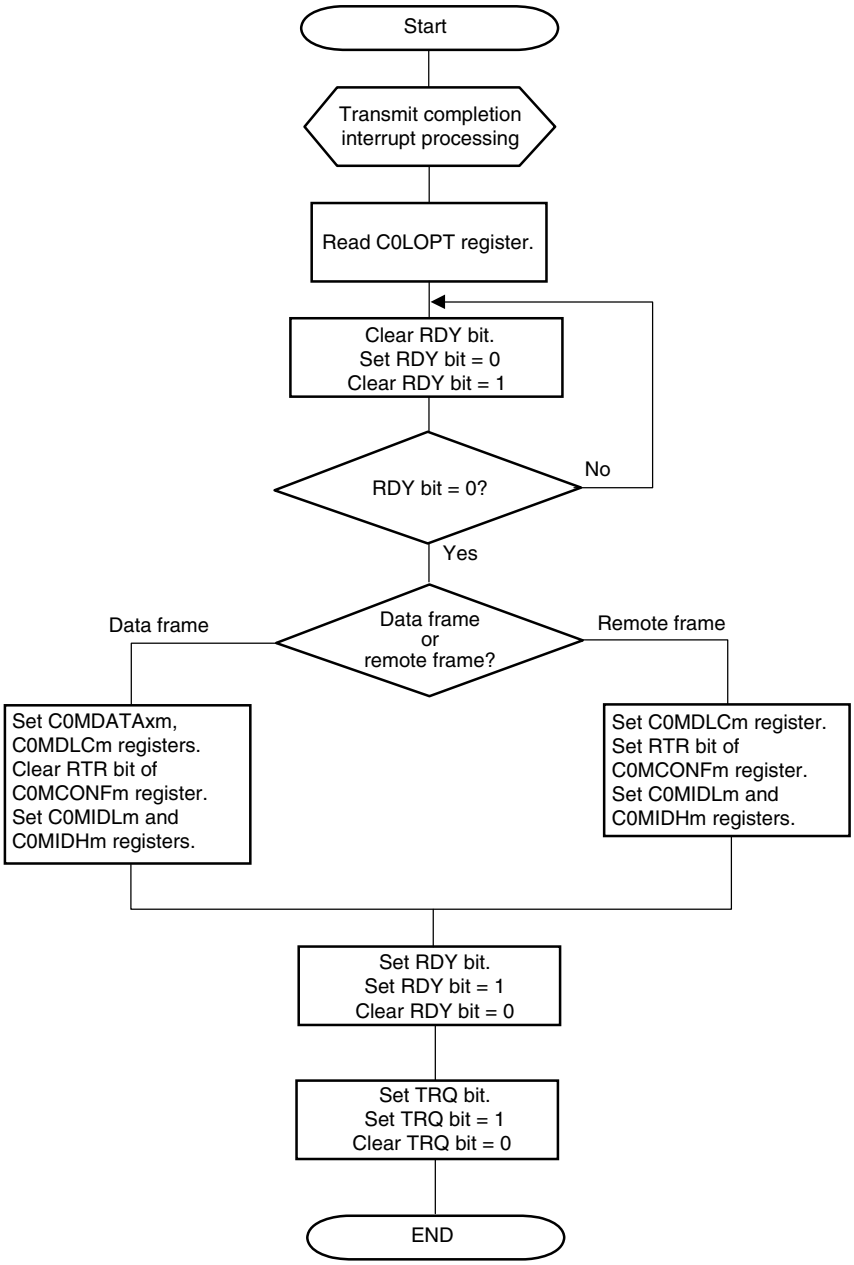
Figure 17-41. ABT Message Transmit Processing



Caution The ABTTRG bit should be set to 1 after the TSTAT bit is cleared to 0. The checking of the TSTAT bit and the setting for the ABTTRG bit to 1 must be continuous.

Remark This processing (message transmit processing with ABS) can only be applied to message buffers 0 to 7. For message buffers other than the ABT message buffers, see **Figure 17-40**.

Figure 17-42. Transmission via Interrupt (Using C0LOPT register)

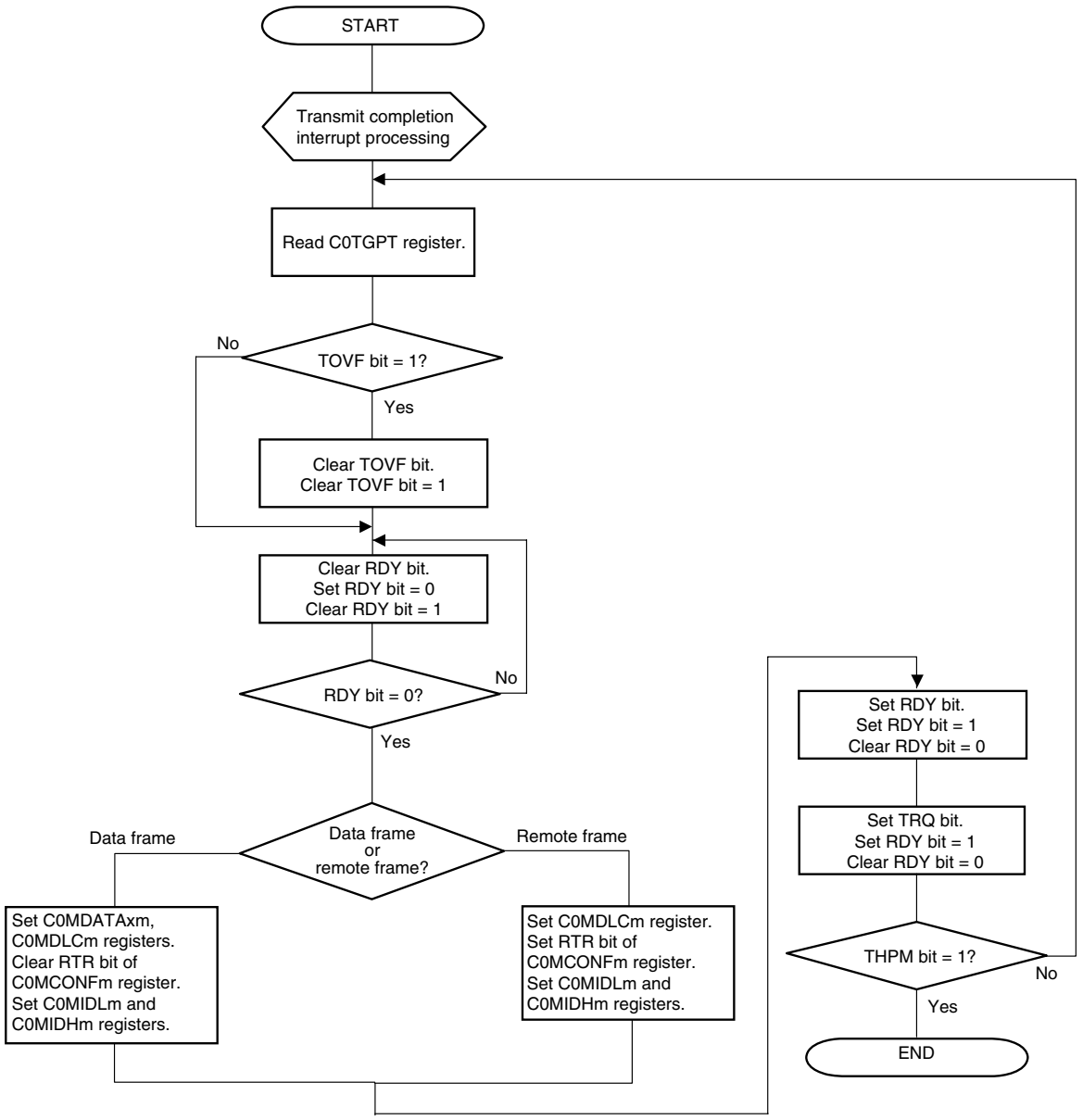


Cautions 1. The TRQ bit should be set after the RDY bit is set.

2. The RDY bit and TRQ bit should not be set at the same time.

Remark Check the MBON bit at the start and end of the interrupt routine to see if the message buffer and transmit history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again. It is therefore recommended to cancel the CAN sleep mode transition request before executing transmission interrupt servicing.

Figure 17-43. Transmission via Interrupt (Using C0TGPT Register)



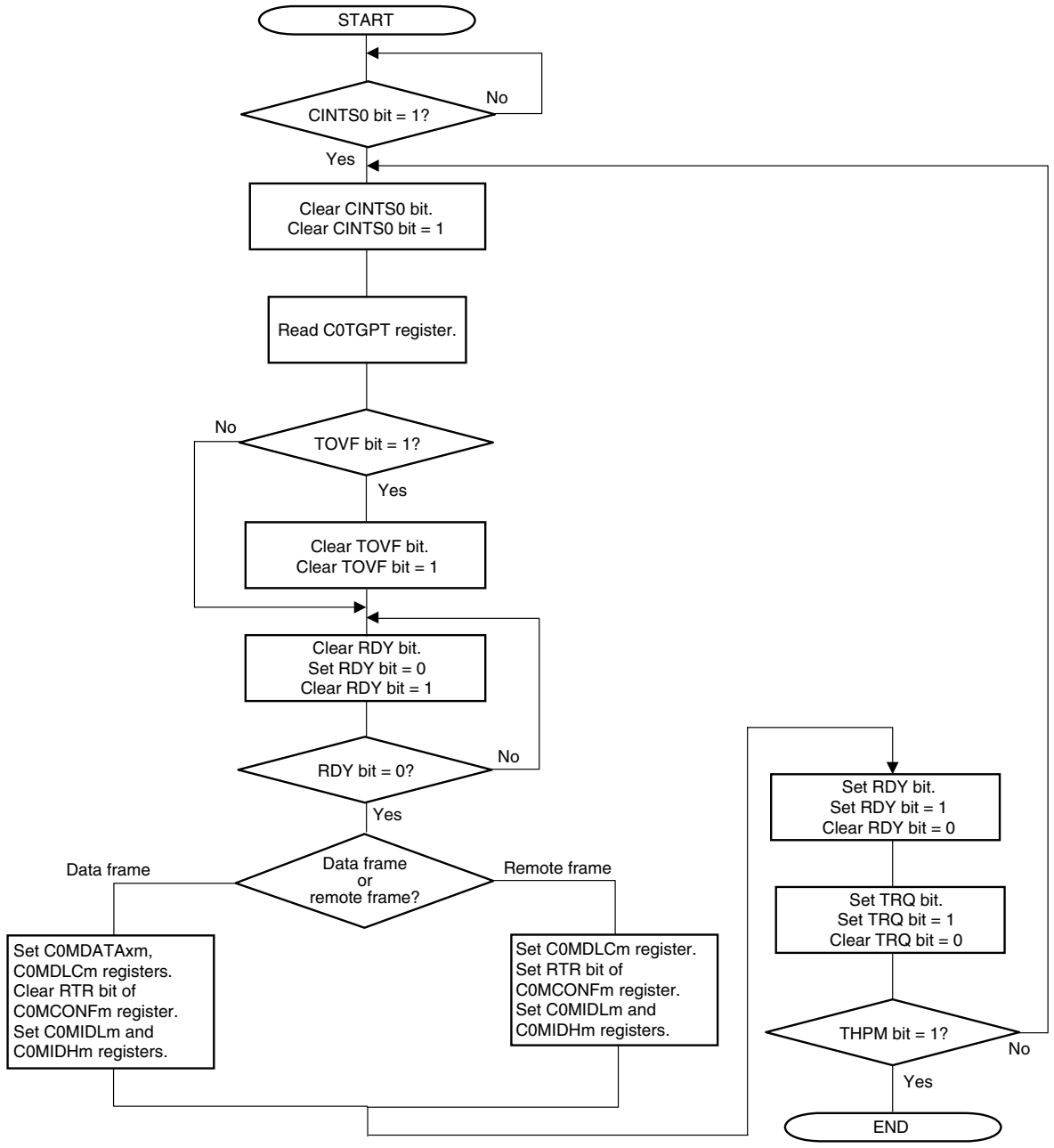
Cautions 1. The TRQ bit should be set after the RDY bit is set.

2. The RDY bit and TRQ bit should not be set at the same time.

Remarks 1. Check the MBON bit at the start and end of the interrupt routine to see if the message buffer and transmit history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again. It is therefore recommended to cancel the CAN sleep mode transition request before executing transmission interrupt servicing.

2. If the TOVF bit is set (1) again, the transmit history list contradicts. Therefore, scan all the transmit message buffers that have completed transmission.

Figure 17-44. Transmission via Software Polling



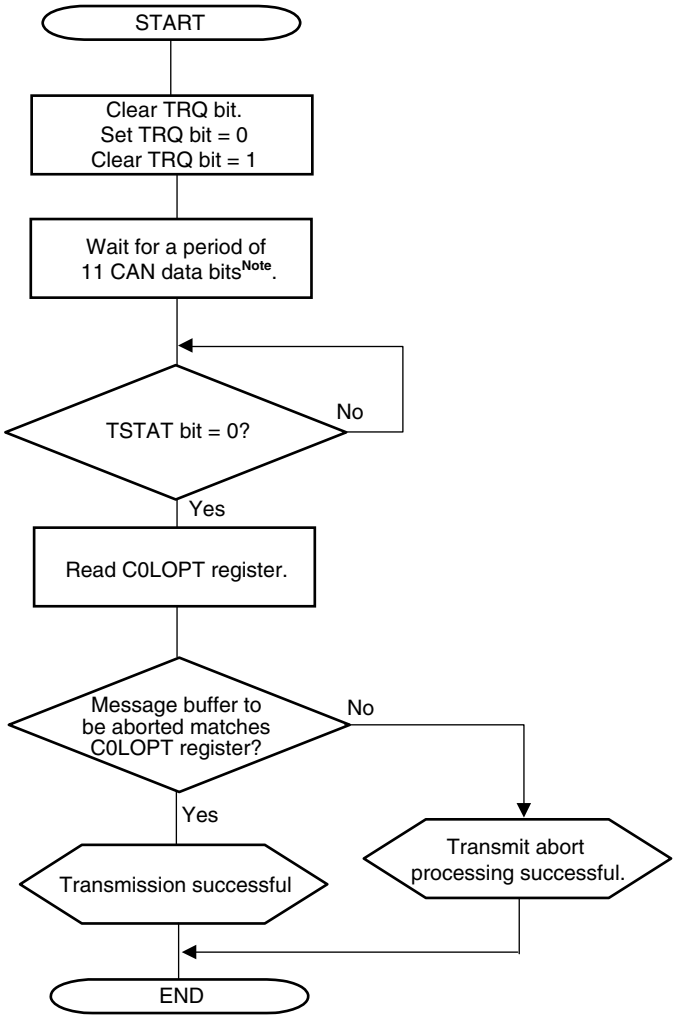
Cautions 1. The TRQ bit should be set after the RDY bit is set.

2. The RDY bit and TRQ bit should not be set at the same time.

Remarks 1. Check the MBON bit at the start and end of the polling routine to see if the message buffer and transmit history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again.

2. If the TOVF bit is set (1) again, the transmit history list contradicts. Therefore, scan all the transmit message buffers that have completed transmission.

Figure 17-45. Transmission Abort Processing (Other Than in Normal Operation Mode with ABT)



Note During a period of a total of 11 bits, 3 bits of interframe space and 8 bits of suspend transmission, the transmission request may have already been acknowledged by the protocol layer. Consequently, transmission may not be aborted but started even if the TRQ bit is cleared.

Cautions 1. Execute transmission abort processing by clearing the TRQ bit, not the RDY bit.

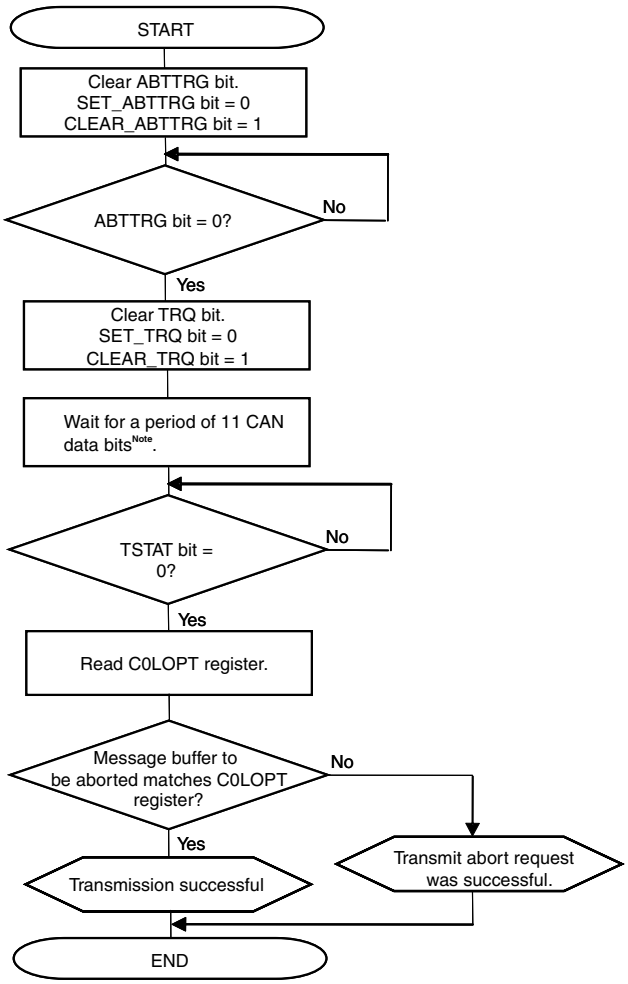
2. Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.

3. The TSTAT bit can be periodically checked by a user application or can be checked after the transmit completion interrupt.

4. Do not execute a new transmission request that includes other message buffers while transmission abort processing is in progress.

5. If data of the same message buffer are successively transmitted or if only one message buffer is used, judgments whether transmission has been successfully executed or failed may contradict. In such a case, make a judgment by using the history information of the C0TGPT register.

Figure 17-46. Transmission Abort Processing Except for ABT Transmission (Normal Operation Mode with ABT)



Note During a period of a total of 11 bits, 3 bits of interframe space and 8 bits of suspend transmission, the transmission request may have already been acknowledged by the protocol layer. Consequently, transmission may not be aborted but started even if the TRQ bit is cleared.

Cautions 1. Execute transmission abort processing by clearing the TRQ bit, not the RDY bit.

2. Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.

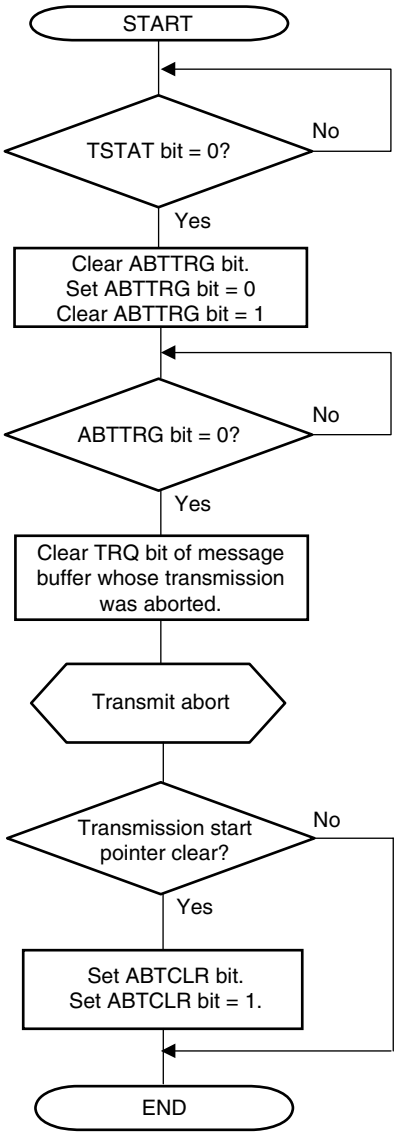
3. The TSTAT bit can be periodically checked by a user application or can be checked after the transmit completion interrupt.

4. Do not execute a new transmission request that includes other message buffers while transmission abort processing is in progress.

5. If data of the same message buffer are successively transmitted or if only one message buffer is used, judgments whether transmission has been successfully executed or failed may contradict. In such a case, make a judgment by using the history information of the C0TGPT register.

Figure 17-47 (a) shows processing that does not skip resuming the transmission of a message that was interrupted when the transmission of an ABT message buffer was aborted.

Figure 17-47 (a). Transmission Abort Processing (Normal Operation Mode with ABT)

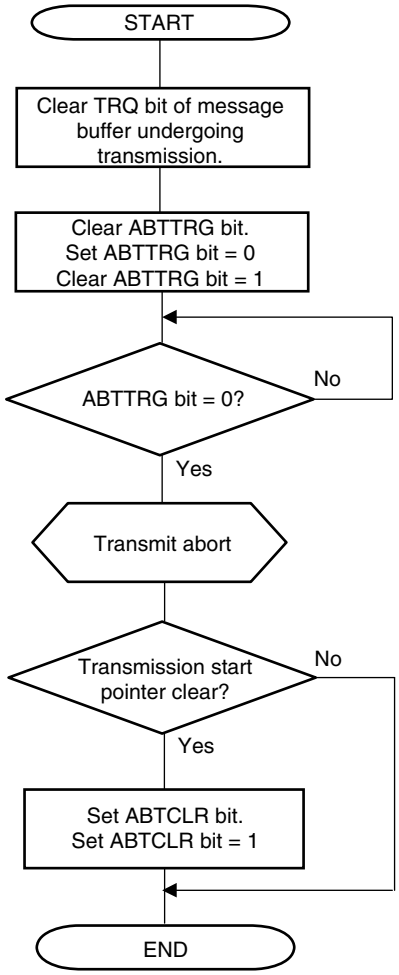


Cautions 1. Do not set any transmission requests while ABT transmission abort processing is in progress.

2. Make a CAN sleep mode/CAN stop mode transition request after the ABTTRG bit is cleared (after ABT mode is stopped) following the procedure shown in Figure 17-47 (a) or (b). When clearing a transmission request in an area other than the ABT area, follow the procedure shown in Figure 17-45.

Figure 17-47 (b) shows the processing that does not skip resuming the transmission of a message that was interrupted when the transmission of an ABT message buffer was aborted.

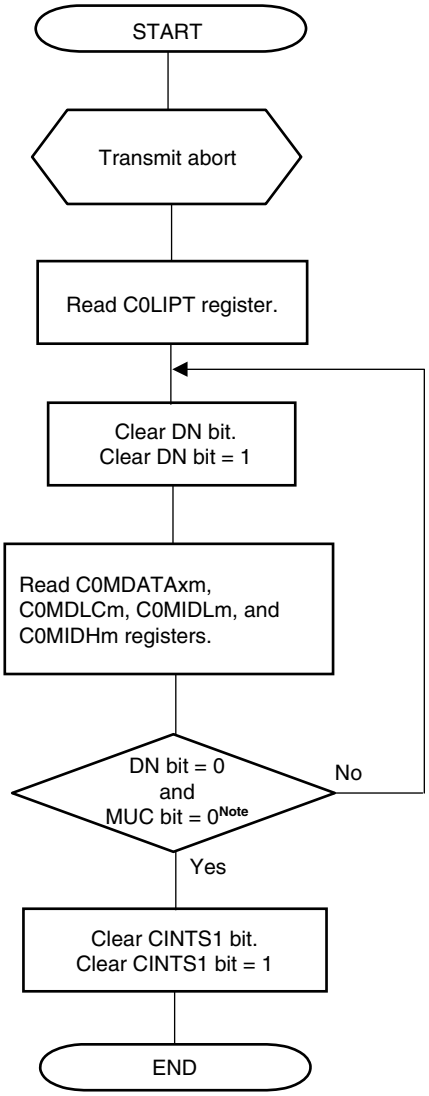
Figure 17-47 (b). Transmission Request Abort Processing (Normal Operation Mode with ABT)



Cautions 1. Do not set any transmission requests while ABT transmission abort processing is in progress.

2. Make a CAN sleep mode/CAN stop mode request after the ABTTRG bit is cleared (after ABT mode is stopped) following the procedure shown in Figure 17-47 (a) or (b). When clearing a transmission request in an area other than the ABT area, follow the procedure shown in Figure 17-45.

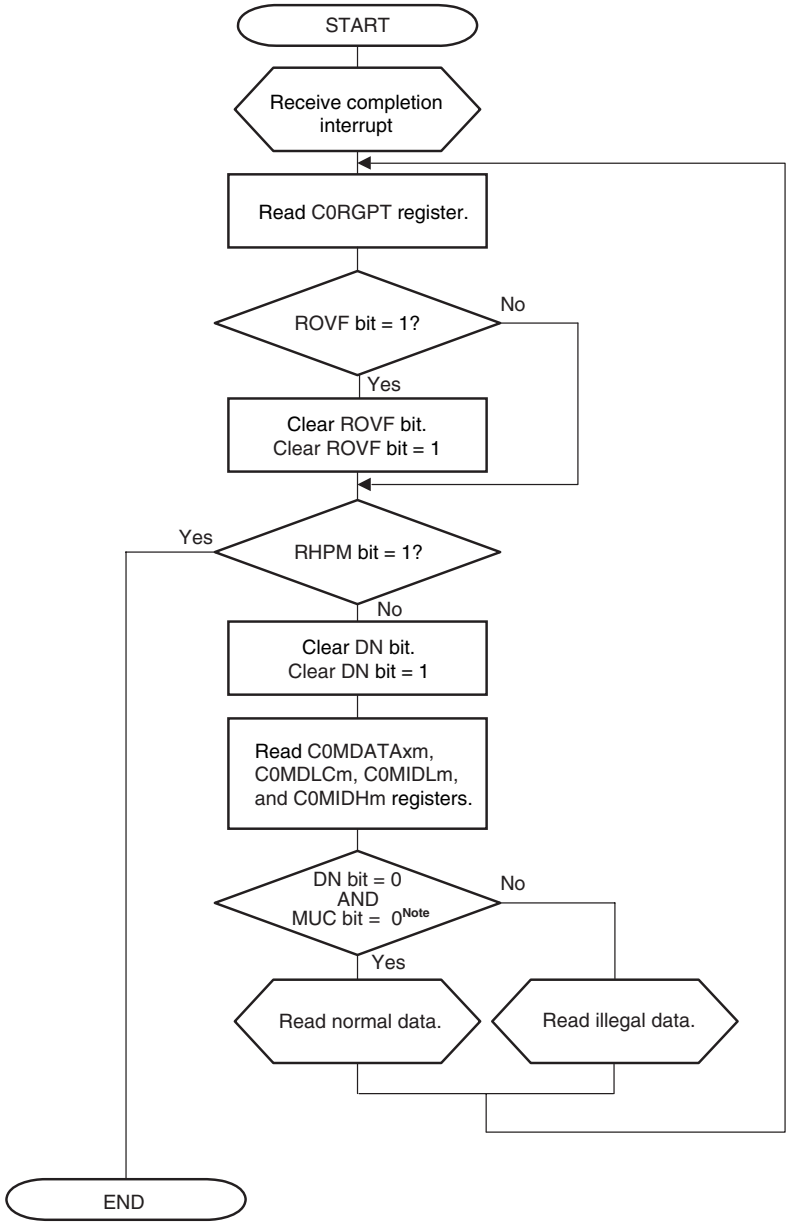
Figure 17-48. Reception via Interrupt (Using C0LIPT Register)



Note Check the MUC and DN bits using one read access.

Remark Check the MBON bit at the start and end of the interrupt routine to see if the message buffer and receive history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again. It is therefore recommended to cancel the CAN sleep mode transition request before executing reception interrupt servicing.

Figure 17-49. Reception via Interrupt (Using C0RGPT Register)

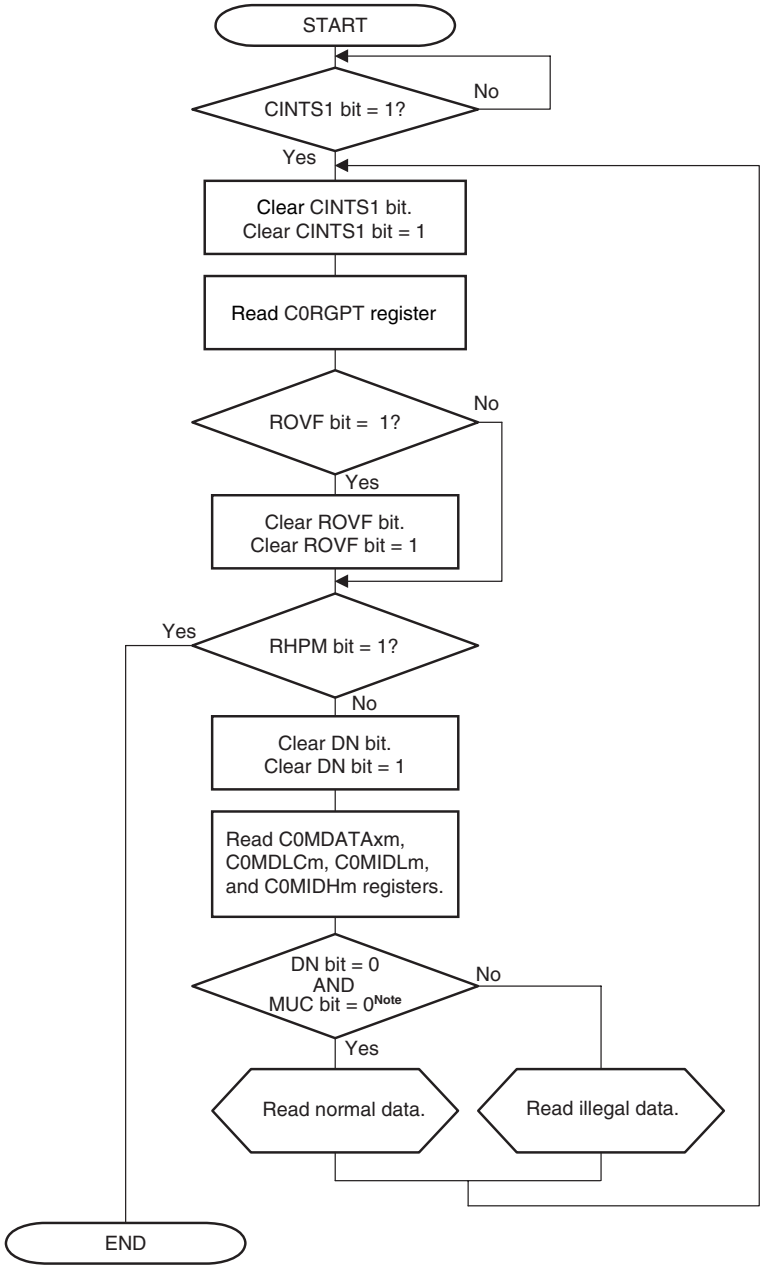


Note Check the MUC and DN bits using one read access.

Remarks 1. Check the MBON bit at the start and end of the interrupt routine to see if the message buffer and receive history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again. It is therefore recommended to cancel the CAN sleep mode transition request before executing reception interrupt servicing.

2. If the ROVF bit has been once set (1), the receive history list contradicts. Therefore, scan all the receive message buffers that have completed reception.

Figure 17-50. Reception via Software Polling

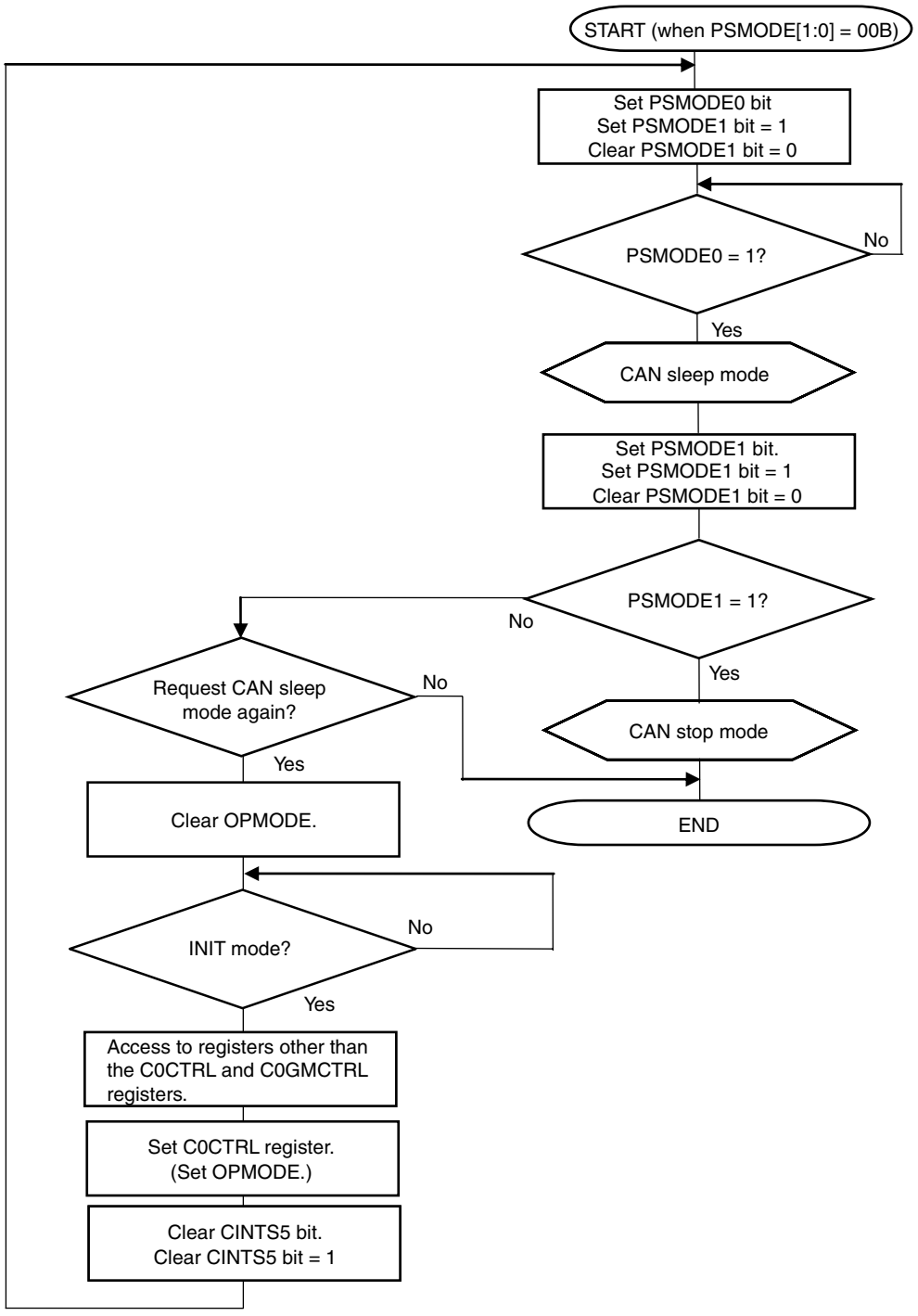


Note Check the MUC and DN bits using one read access.

Remarks 1. Check the MBON bit at the start and end of the polling routine to see if the message buffer and receive history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again.

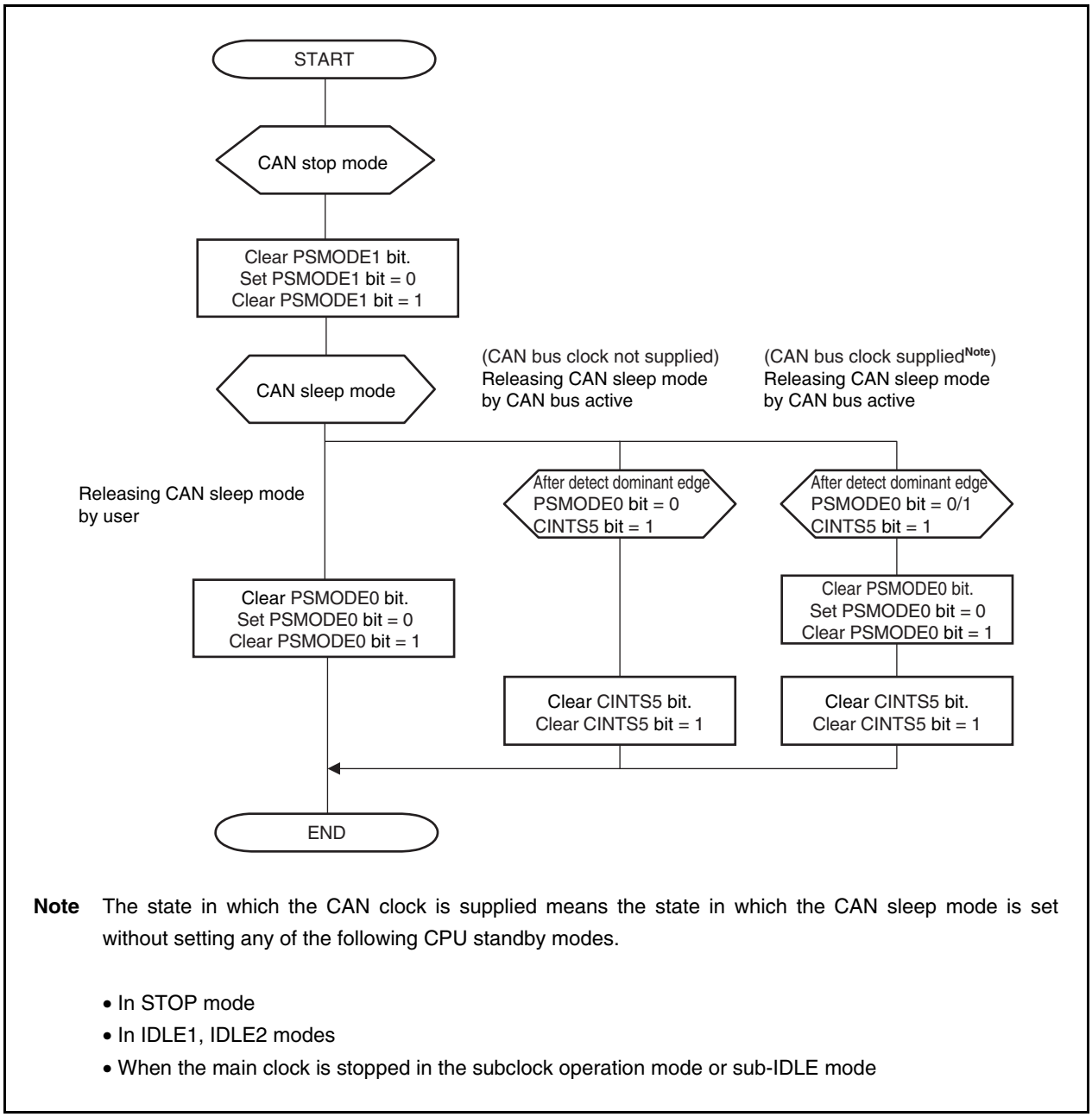
2. If the ROVF bit has been once set (1), the receive history list contradicts. Therefore, scan all the receive message buffers that have completed reception.

Figure 17-51. Setting CAN Sleep Mode/Stop Mode



Caution To abort transmission before making a request for the CAN sleep mode, perform processing according to Figures 17-45 to 17-47.

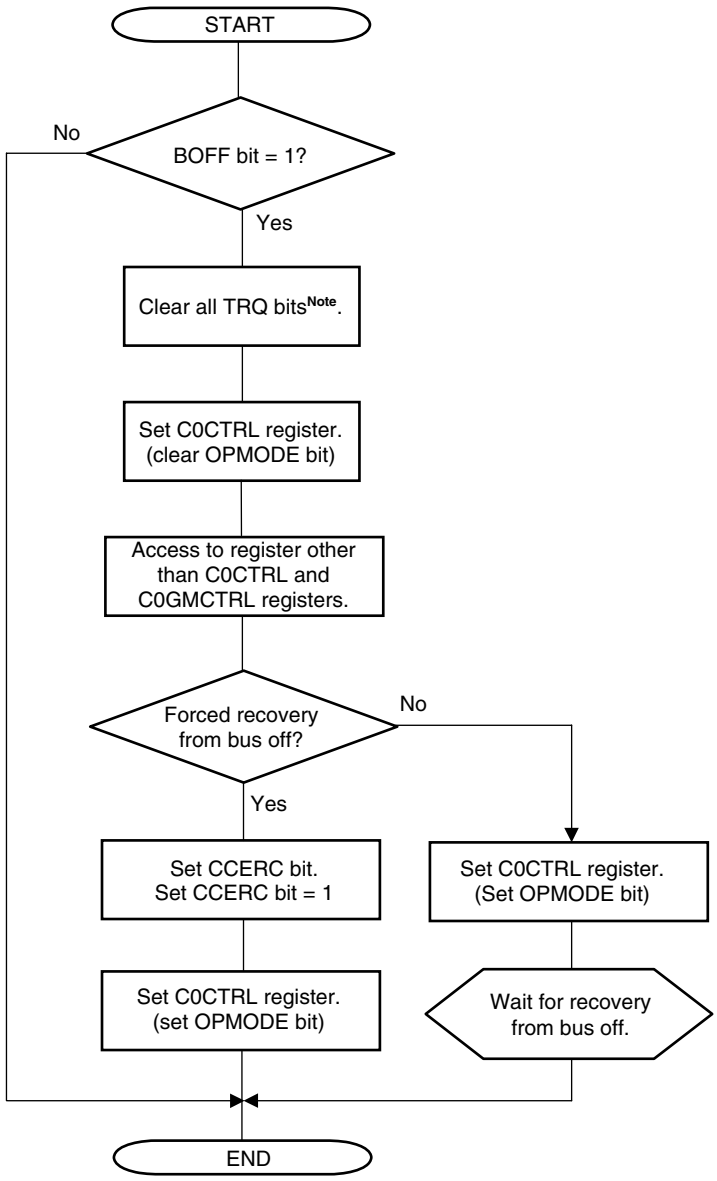
Figure 17-52. Clear CAN Sleep/Stop Mode



Note The state in which the CAN clock is supplied means the state in which the CAN sleep mode is set without setting any of the following CPU standby modes.

- In STOP mode
- In IDLE1, IDLE2 modes
- When the main clock is stopped in the subclock operation mode or sub-IDLE mode

Figure 17-53. Bus-off Recovery (Other Than in Normal Operation Mode with ABT)

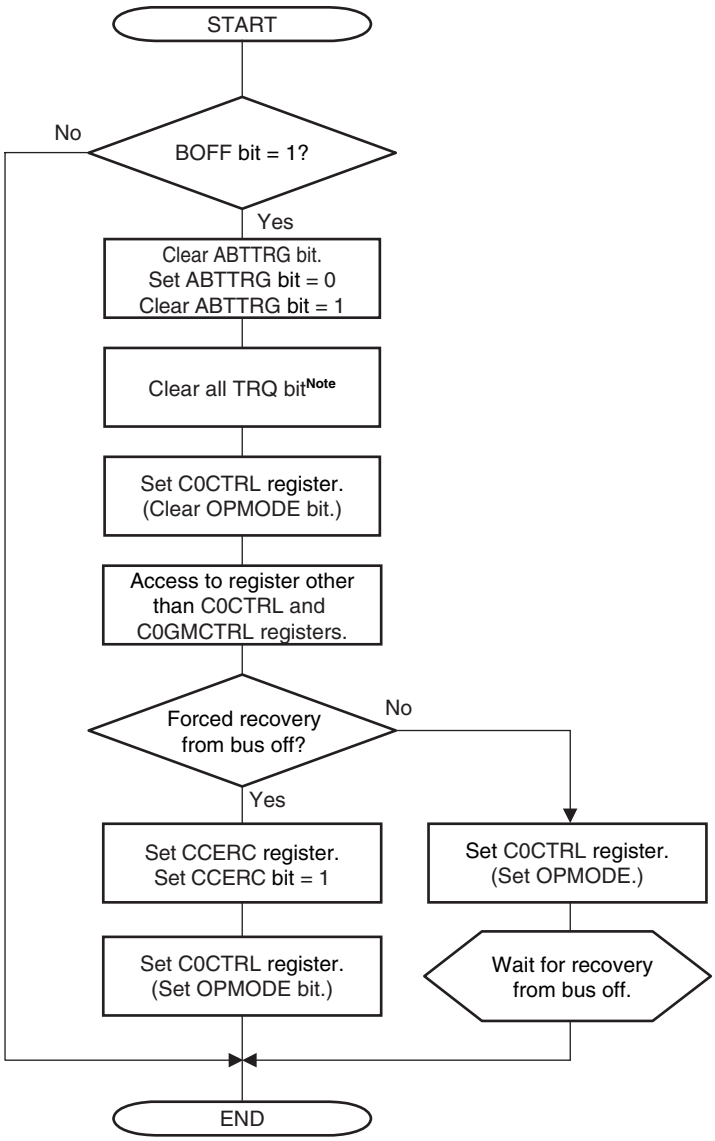


Note To initialize the message buffer by clearing the RDY bit before starting the bus-off recovery sequence, clear all the TRQ bits.

Caution If a request to change the mode from the initialization mode to any operation mode is made to execute the bus-off recovery sequence again during a bus-off recovery sequence, the receive error counter (C0ERC.REC0 to REC6 bits) is cleared. It is therefore necessary to detect 11 contiguous recessive bits 128 times on the bus again.

Remark OPMODE: Normal operation mode, normal operation mode with ABT, receive-only mode, single-shot mode, self-test mode

Figure 17-54. Bus-off Recovery (Normal Operation Mode with ABT)



Note To initialize the message buffer by clearing the RDY bit before starting the bus-off recovery sequence, clear all the TRQ bits.

Caution If a request to change the mode from the initialization mode to any operation mode is made to execute the bus-off recovery sequence again during a bus-off recovery sequence, the receive error counter (C0ERC.REC0 to REC6 bits) is cleared. It is therefore necessary to detect 11 contiguous recessive bits 128 times on the bus again.

Remark OPMODE: Normal operation mode, normal operation mode with ABT, receive-only mode, single-shot mode, self-test mode

Figure 17-55. Normal Shutdown Process

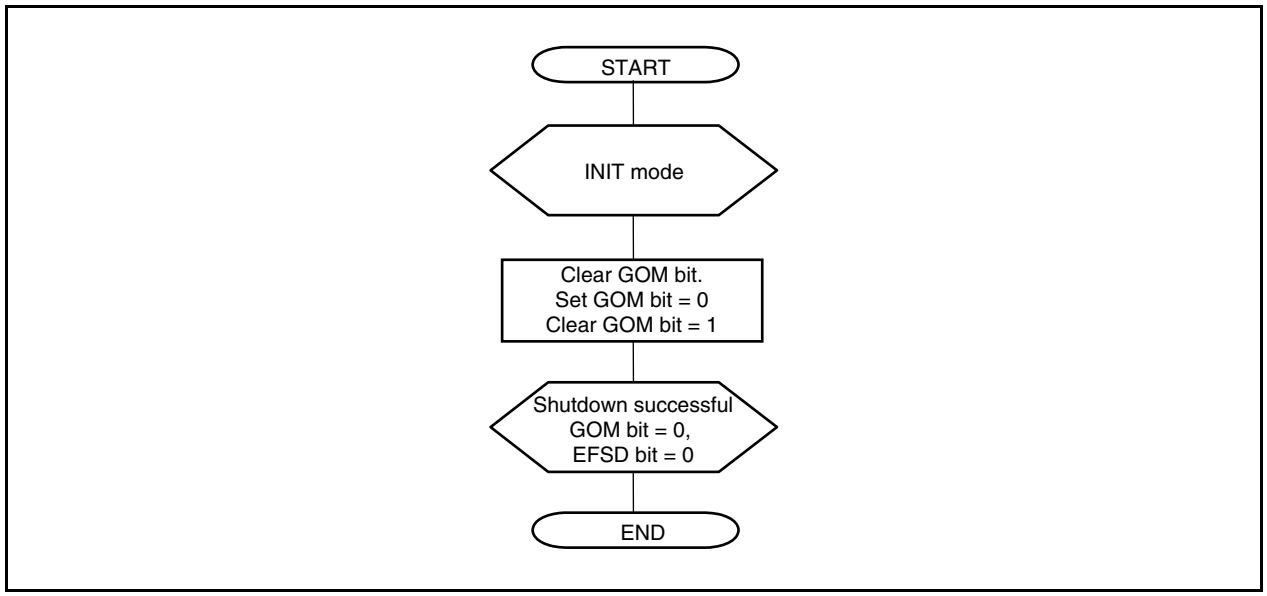
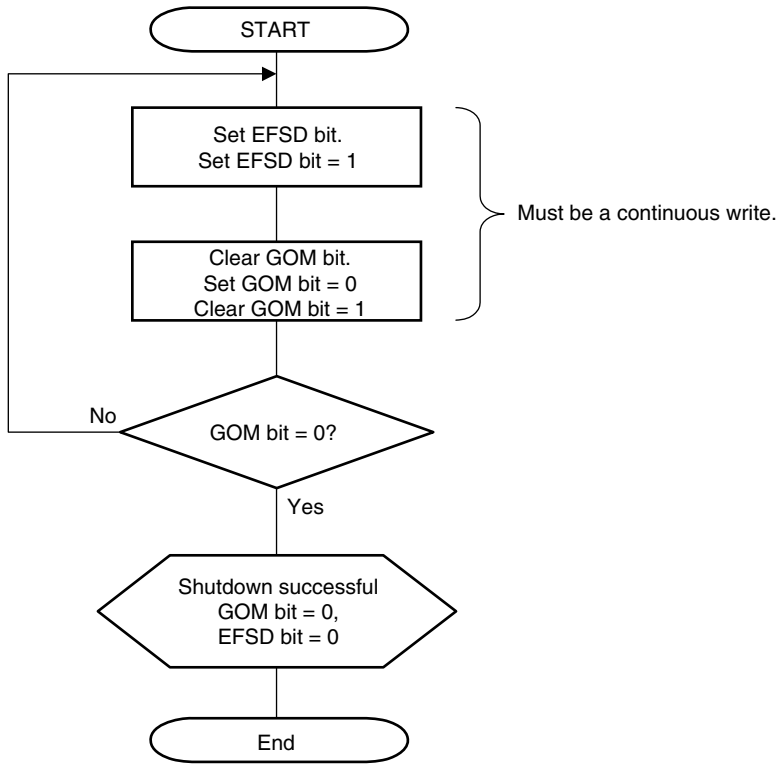


Figure 17-56. Forced Shutdown Process



Caution Do not read- or write-access any registers by software between setting the EFSD bit and clearing the GOM bit.

Figure 17-57. Error Handling

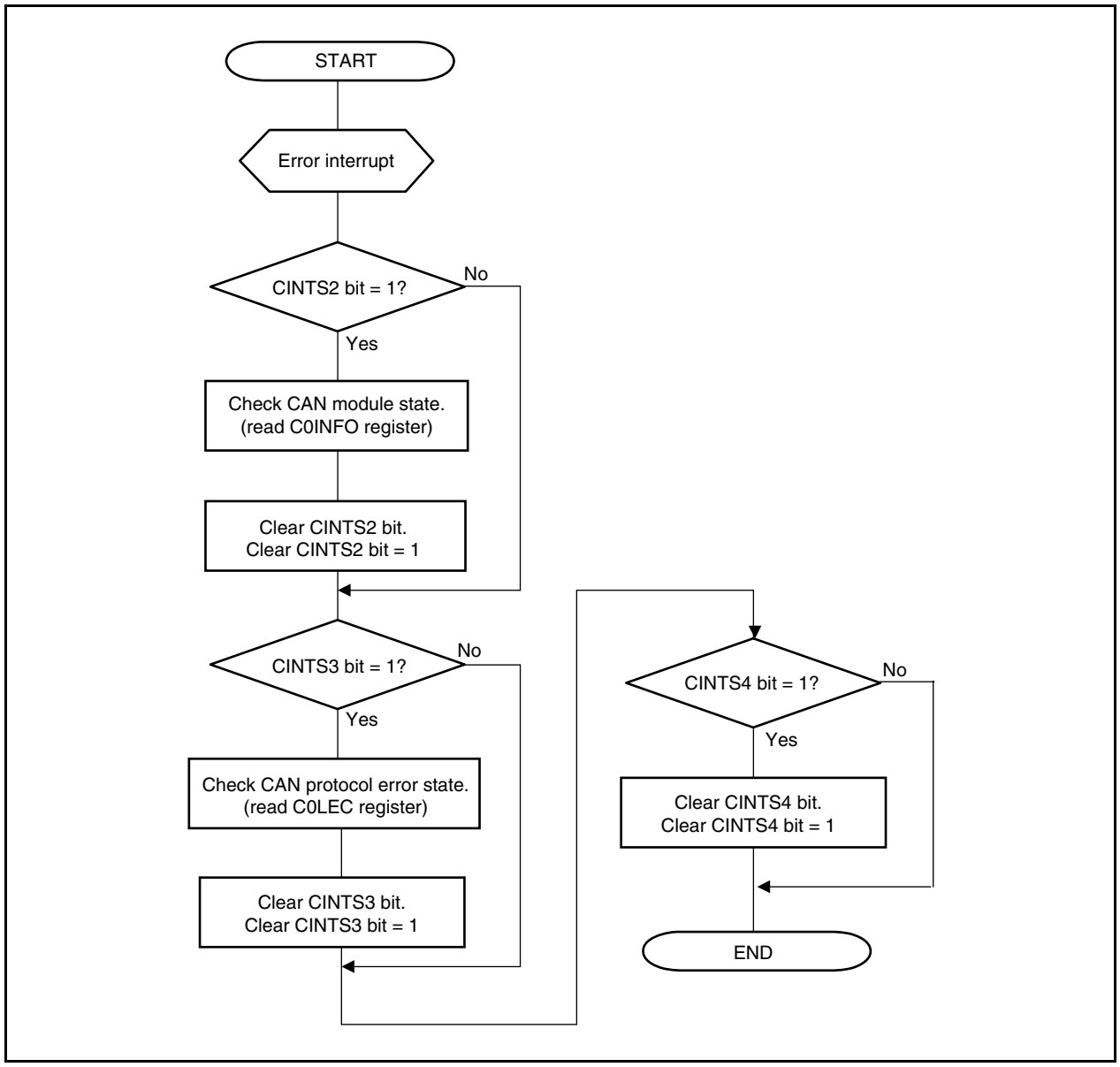
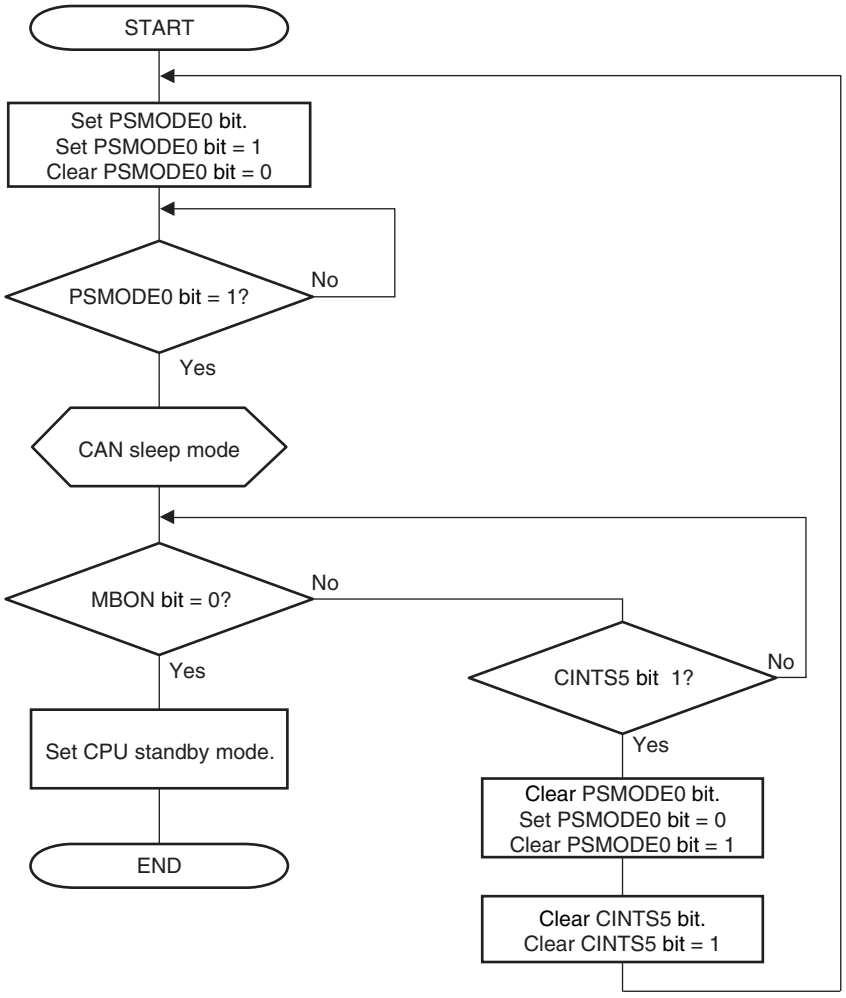
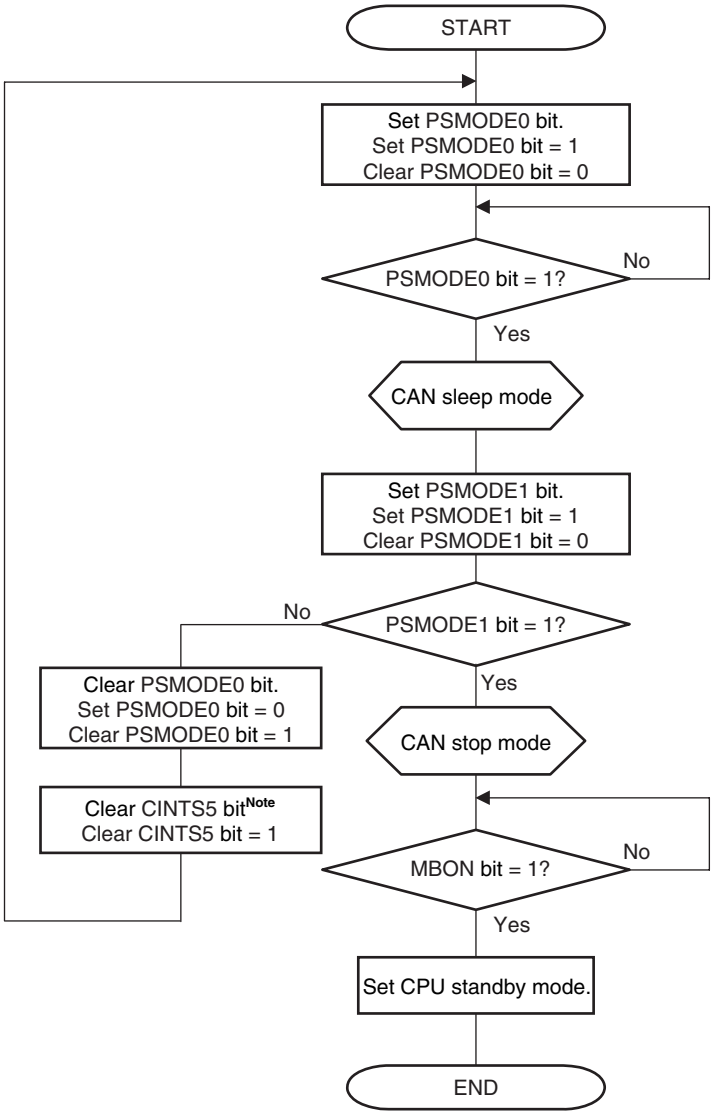


Figure 17-58. Setting CPU Standby (from CAN Sleep Mode)



Note Check if the CPU is in the CAN sleep mode before setting it to the standby mode. The CAN sleep mode may be released by wakeup after it is checked if the CPU is in the CAN sleep mode and before the CPU is set in the standby mode.

Figure 17-59. Setting CPU Standby (from CAN Stop Mode)



Note During wake-up interrupts

Caution The CAN stop mode can only be released by writing 01 to the C0CTRL.PSMODE1 and C0CTRL.PSMODE0 bits. It cannot be released by changing the CAN bus.

<R>

CHAPTER 18 DMA FUNCTION (DMA CONTROLLER)

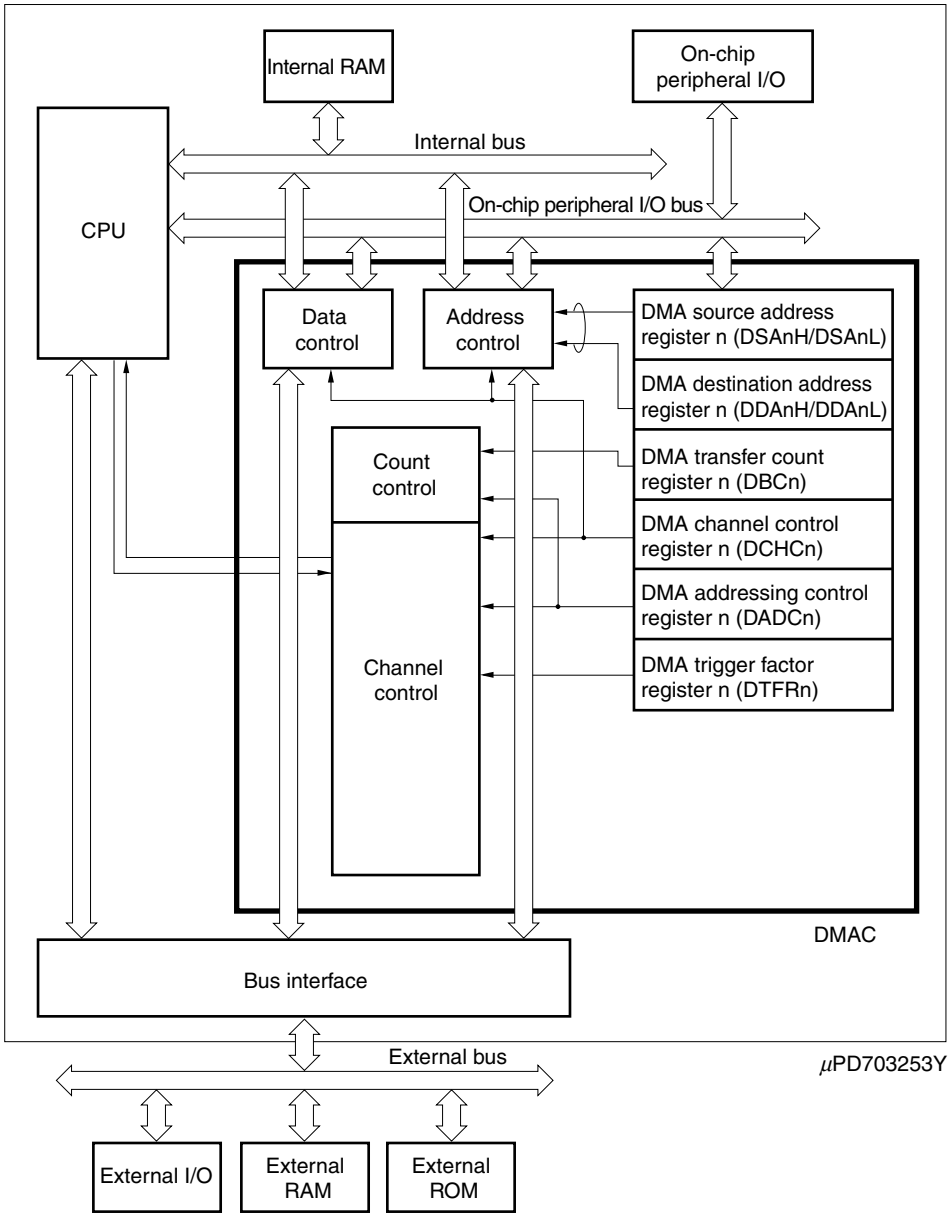
The DMA (Direct Memory Access) controller (DMAC) is provided only in the μPD703253Y.

The DMAC controls data transfer between memory and I/O, between memories, or between I/Os based on DMA requests issued by the on-chip peripheral I/O (serial interface, timer/counter, and A/D converter), interrupts from external input pins, or software triggers (memory refers to internal RAM or external memory).

18.1 Features

- 4 independent DMA channels
- Transfer unit: 8/16 bits
- Maximum transfer count: 65,536 (2^{16})
- Transfer type: Two-cycle transfer
- Transfer mode: Single transfer mode
- Transfer requests
  - Request by interrupts from on-chip peripheral I/O (serial interface, timer/counter, A/D converter) or interrupts from external input pin
  - Requests by software trigger
- Transfer targets
  - Internal RAM ↔ Peripheral I/O
  - Peripheral I/O ↔ Peripheral I/O
  - Internal RAM ↔ External memory
  - External memory ↔ Peripheral I/O
  - External memory ↔ External memory

18.2 Configuration



Remark n = 0 to 3

18.3 Registers

(1) DMA source address registers 0 to 3 (DSA0 to DSA3)

The DSA0 to DSA3 registers set the DMA source addresses (26 bits each) for DMA channel n (n = 0 to 3).

These registers are divided into two 16-bit registers, DSAnH and DSAnL.

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

After reset: Undefined R/W Address: DSA0H FFFFF082H, DSA1H FFFFF08AH, DSA2H FFFFF092H, DSA3H FFFFF09AH, DSA0L FFFFF080H, DSA1L FFFFF088H, DSA2L FFFFF090H, DSA3L FFFFF098H

DSAnH (n = 0 to 3)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IR	0	0	0	0	0	SA25	SA24	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16

DSAnL (n = 0 to 3)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0

IR	Specification of DMA transfer source
0	External memory or on-chip peripheral I/O
1	Internal RAM

SA25 to SA16	Set the address (A25 to A16) of the DMA transfer source (default value is undefined). During DMA transfer, the next DMA transfer source address is held. When DMA transfer is completed, the DMA address set first is held.

SA15 to SA0	Set the address (A15 to A0) of the DMA transfer source (default value is undefined). During DMA transfer, the next DMA transfer source address is held. When DMA transfer is completed, the DMA address set first is held.

Cautions 1. Be sure to clear bits 14 to 10 of the DSAnH register to 0.

2. Set the DSAnH and DSAnL registers at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).

- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

3. When the value of the DSAn register is read, two 16-bit registers, DSAnH and DSAnL, are read. If reading and updating conflict, the value being updated may be read (see 18.13 Cautions).

4. Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer. If these registers are not set, the operation when DMA transfer is started is not guaranteed.

(2) DMA destination address registers 0 to 3 (DDA0 to DDA3)

The DDA0 to DDA3 registers set the DMA destination address (26 bits each) for DMA channel n (n = 0 to 3). These registers are divided into two 16-bit registers, DDAnH and DDAnL.

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

After reset: Undefined R/W Address: DDA0H FFFFF086H, DDA1H FFFFF08EH, DDA2H FFFFF096H, DDA3H FFFFF09EH, DDA0L FFFFF084H, DDA1L FFFFF08CH, DDA2L FFFFF094H, DDA3L FFFFF09CH

DDAnH (n = 0 to 3)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IR	0	0	0	0	0	DA25	DA24	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16

DDAnL (n = 0 to 3)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0

IR	Specification of DMA transfer destination
0	External memory or on-chip peripheral I/O
1	Internal RAM

DA25 to DA16	Set an address (A25 to A16) of DMA transfer destination (default value is undefined). During DMA transfer, the next DMA transfer destination address is held. When DMA transfer is completed, the DMA transfer source address set first is held.

DA15 to DA0	Set an address (A15 to A0) of DMA transfer destination (default value is undefined). During DMA transfer, the next DMA transfer destination address is held. When DMA transfer is completed, the DMA transfer source address set first is held.

Cautions 1. Be sure to clear bits 14 to 10 of the DDAnH register to 0.

2. Set the DDAnH and DDAnL registers at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).

- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

3. When the value of the DDAn register is read, two 16-bit registers, DDAnH and DDAnL, are read. If reading and updating conflict, a value being updated may be read (see 18.13 Cautions).

4. Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer. If these registers are not set, the operation when DMA transfer is started is not guaranteed.

(3) DMA transfer count registers 0 to 3 (DBC0 to DBC3)

The DBC0 to DBC3 registers are 16-bit registers that set the byte transfer count for DMA channel n (n = 0 to 3).

These registers hold the remaining transfer count during DMA transfer.

These registers are decremented by 1 per one transfer regardless of the transfer data unit (8/16 bits), and the transfer is terminated if a borrow occurs.

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

After reset: Undefined R/W Address: DBC0 FFFFF0C0H, DBC1 FFFFF0C2H, DBC2 FFFFF0C4H, DBC3 FFFFF0C6H

DBCn (n = 0 to 3)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0

BC15 to BC0	Byte transfer count setting or remaining byte transfer count during DMA transfer
0000H	Byte transfer count 1 or remaining byte transfer count
0001H	Byte transfer count 2 or remaining byte transfer count
:	:
FFFFH	Byte transfer count 65,536 (2^{16}) or remaining byte transfer count
The number of transfer data set first is held when DMA transfer is complete.	

Cautions 1. Set the DBCn register at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).

- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

2. Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer. If these registers are not set, the operation when DMA transfer is started is not guaranteed.

(4) DMA addressing control registers 0 to 3 (DADC0 to DADC3)

The DADC0 to DADC3 registers are 16-bit registers that control the DMA transfer mode for DMA channel n (n = 0 to 3).

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

Reset sets these registers to 0000H.

After reset: 0000H R/W Address: DADC0 FFFFF0D0H, DADC1 FFFFF0D2H, DADC2 FFFFF0D4H, DADC3 FFFFF0D6H

DADCn (n = 0 to 3)

15	14	13	12	11	10	9	8
0	DS0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
SAD1	SAD0	DAD1	DAD0	0	0	0	0

DS0	Setting of transfer data size
0	8 bits
1	16 bits

SAD1	SAD0	Setting of count direction of the transfer source address
0	0	Increment
0	1	Decrement
1	0	Fixed
1	1	Setting prohibited

DAD1	DAD0	Setting of count direction of the destination address
0	0	Increment
0	1	Decrement
1	0	Fixed
1	1	Setting prohibited

Cautions 1. Be sure to clear bits 15, 13 to 8, and 3 to 0 of the DADCn register to "0".

2. Set the DADCn register at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).

- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

3. The DS0 bit specifies the size of the transfer data, and does not control bus sizing. If 8-bit data (DS0 bit = 0) is set, therefore, the lower data bus is not always used.

4. If the transfer data size is set to 16 bits (DS0 bit = 1), transfer cannot be started from an odd address. Transfer is always started from an address with the first bit of the lower address aligned to 0.

5. If DMA transfer is executed on an on-chip peripheral I/O register (as the transfer source or destination), be sure to specify the same transfer size as the register size. For example, to execute DMA transfer on an 8-bit register, be sure to specify 8-bit transfer.

(5) DMA channel control registers 0 to 3 (DCHC0 to DCHC3)

The DCHC0 to DCHC3 registers are 8-bit registers that control the DMA transfer operating mode for DMA channel n.

These registers can be read or written in 8-bit or 1-bit units. (However, bit 7 is read-only and bits 1 and 2 are write-only. If bit 1 or 2 is read, the read value is always 0.)

Reset sets these registers to 00H.

After reset: 00H R/W Address: DCHC0 FFFFF0E0H, DCHC1 FFFFF0E2H, DCHC2 FFFFF0E4H, DCHC3 FFFFF0E6H

	<7>	6	5	4	3	<2>	<1>	<0>
DCHCn	TCn Note 1	0	0	0	0	INITn Note 2	STGn Note 2	Enn

(n = 0 to 3)

TCn Note 1	Status flag indicates whether DMA transfer through DMA channel n has completed or not
0	DMA transfer had not completed.
1	DMA transfer had completed.
It is set to 1 on the last DMA transfer and cleared to 0 when it is read.	

INITn Note 2	If the INITn bit is set to 1 with DMA transfer disabled (Enn bit = 0), the DMA transfer status can be initialized. When re-setting the DMA transfer status (re-setting the DDAnH, DDAnL, DSAnH, DSAnL, DBCn, and DADCn registers) before DMA transfer is completed (before the TCn bit is set to 1), be sure to initialize the DMA channel. When initializing the DMA controller, however, be sure to observe the procedure described in **18.13 Cautions**.

STGn Note 2	This is a software startup trigger of DMA transfer. If this bit is set to 1 in the DMA transfer enable state (TCn bit = 0, Enn bit = 1), DMA transfer is started.

Enn	Setting of whether DMA transfer through DMA channel n is to be enabled or disabled
0	DMA transfer disabled
1	DMA transfer enabled
DMA transfer is enabled when the Enn bit is set to 1. When DMA transfer is completed (when a terminal count is generated), this bit is automatically cleared to 0. To abort DMA transfer, clear the Enn bit to 0 by software. To resume, set the Enn bit to 1 again. When aborting or resuming DMA transfer, however, be sure to observe the procedure described in **18.13 Cautions**.	

Notes 1. The TCn bit is read-only.

2. The INITn and STGn bits are write-only.

Cautions 1. Be sure to clear bits 6 to 3 of the DCHCn register to 0.

2. When DMA transfer is completed (when a terminal count is generated), the Enn bit is cleared to 0 and then the TCn bit is set to 1. If the DCHCn register is read while its bits are being updated, a value indicating "transfer not completed and transfer is disabled" (TCn bit = 0 and Enn bit = 0) may be read.

(6) DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)

The DTFR0 to DTFR3 registers are 8-bit registers that control the DMA transfer start trigger via interrupt request signals from on-chip peripheral I/O.

The interrupt request signals set by these registers serve as DMA transfer start factors.

These registers can be read or written in 8-bit units. However, DFn bit can be read or written in 1-bit units.

Reset sets these registers to 00H.

(1/2)

After reset: 00H R/W Address: DTFR0 FFFFF810H, DTFR1 FFFFF812H, DTFR2 FFFFF814H, DTFR3 FFFFF816H

	<7>	6	5	4	3	2	1	0
DTFRn	DFn	0	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0

(n = 0 to 3)

DFn[Note]	DMA transfer request status flag
0	No DMA transfer request
1	DMA transfer request

Note Do not set the DFn bit to “1” by software. Write 0 to this bit to clear a DMA transfer request if an interrupt that is specified as the cause of starting DMA transfer occurs while DMA transfer is disabled.

Caution 1. Set the IFCn5 to IFCn0 bits at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0).

- **Period from after reset to start of first DMA transfer**
- **Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer**
- **Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer**

(2/2)

Cautions 2. Be sure to follow the steps below when changing the DTFRn register settings.

- **When the values to be set to bits IFCn5 to IFCn0 are not set to bits IFCm5 to IFCm0 of another channel (n = 0 to 3, m = 0 to 3, n ≠ m)**
  - **<1> Stop the DMAn operation of the channel to be rewritten (DCHCn.Enn bit = 0).**
  - **<2> Change the DTFRn register settings. (Be sure to set DFn bit = 0 and change the settings in the 8-bit manipulation.)**
  - **<3> Confirm that DFn bit = 0. (Stop the interrupt generation source operation beforehand.)**
  - **<4> Enable the DMAn operation (Enn bit = 1).**
- **When the values to be set to bits IFCn5 to IFCn0 are set to bits IFCm5 to IFCm0 of another channel (n = 0 to 3, m = 0 to 3, n ≠ m)**
  - **<1> Stop the DMAn operation of the channel to be rewritten (DCHCn.Enn bit = 0).**
  - **<2> Stop the DMAm operation of the channel where the same values are set to bits IFCm5 to IFCm0 as the values to be used to rewrite bits IFCn5 to IFCn0 (DCHCm.Emm bit = 0).**
  - **<3> Change the DTFRn register settings. (Be sure to set DFn bit = 0 and change the settings in the 8-bit manipulation.)**
  - **<4> Confirm that bits DFn and DFm = 0. (Stop the interrupt generation source operation beforehand.)**
  - **<5> Enable the DMAn operation (bits Enn and Emm = 1).**

3. An interrupt request that is generated in the standby mode (IDEL1, IDLE2, STOP, or sub-IDLE mode) does not start the DMA transfer cycle (nor is the DFn bit set to 1).

4. If a DMA start factor is selected by the IFCn5 to IFCn0 bits, the DFn bit is set to 1 when an interrupt occurs from the selected on-chip peripheral I/O, regardless of whether the DMA transfer is enabled or disabled. If DMA is enabled in this status, DMA transfer is immediately started.

Remark For the IFCn5 to IFCn0 bits, see **Table 18-1 DMA Start Factors**.

Table 18-1. DMA Start Factors

IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt Source
0	0	0	0	0	0	DMA request by interrupt disabled
0	0	0	0	0	1	INTP0
0	0	0	0	1	0	INTP1
0	0	0	0	1	1	INTP2
0	0	0	1	0	0	INTP3
0	0	0	1	0	1	INTP4
0	0	0	1	1	0	INTP5
0	0	0	1	1	1	INTP6
0	0	1	0	0	0	INTP7
0	0	1	1	1	0	INTTP0OV
0	0	1	1	1	1	INTTP0CC0
0	1	0	0	0	0	INTTP0CC1
0	1	0	0	0	1	INTTP1OV
0	1	0	0	1	0	INTTP1CC0
0	1	0	0	1	1	INTTP1CC1
0	1	0	1	0	0	INTTP2OV
0	1	0	1	0	1	INTTP2CC0
0	1	0	1	1	0	INTTP2CC1
0	1	0	1	1	1	INTTP3CC0
0	1	1	0	0	0	INTTP3CC1
0	1	1	0	0	1	INTTP4CC0
0	1	1	0	1	0	INTTP4CC1
0	1	1	0	1	1	INTTP5CC0[Note]
0	1	1	1	0	0	INTTP5CC1[Note]
0	1	1	1	0	1	INTTM0EQ0
0	1	1	1	1	0	INTCB0R/INTIIC1
0	1	1	1	1	1	INTCB0T
1	0	0	0	0	0	INTCB1R
1	0	0	0	0	1	INTCB1T
1	0	0	1	0	0	INTCB3R
1	0	0	1	0	1	INTCB3T
1	0	0	1	1	0	INTUA0R
1	0	0	1	1	1	INTUA0T
1	0	1	0	0	0	INTUA1R
1	0	1	0	0	1	INTUA1T
1	0	1	0	1	0	INTIIC0
1	0	1	1	0	0	INTAD
1	0	1	1	0	1	INTKR
Other than above						Setting prohibited

Note μPD703253Y only

Remark n = 0 to 3

18.4 Transfer Targets

Table 18-2 shows the relationship between the transfer targets (√: Transfer enabled, ×: Transfer disabled).

Table 18-2. Relationship Between Transfer Targets

		Transfer Destination			
		Internal ROM	On-Chip Peripheral I/O	Internal RAM	External Memory
Source	On-chip peripheral I/O	×	√	√	√
	Internal RAM	×	√	×	√
	External memory	×	√	√	√
	Internal ROM	×	×	×	×

Caution The operation is not guaranteed for combinations of transfer destination and source marked with "×" in Table 18-2.

18.5 Transfer Modes

Single transfer is supported as the transfer mode.

In single transfer mode, the bus is released at each byte/halfword transfer. If there is a subsequent DMA transfer request, transfer is performed again once. This operation continues until a terminal count occurs.

When the DMAC has released the bus, if another higher priority DMA transfer request is issued, the higher priority DMA request always takes precedence.

If a new transfer request of the same channel and a transfer request of another channel with a lower priority are generated in a transfer cycle, DMA transfer of the channel with the lower priority is executed after the bus is released to the CPU (the new transfer request of the same channel is ignored in the transfer cycle).

18.6 Transfer Types

As a transfer type, the 2-cycle transfer is supported.

In two-cycle transfer, data transfer is performed in two cycles, a read cycle and a write cycle.

In the read cycle, the transfer source address is output and reading is performed from the source to the DMAC. In the write cycle, the transfer destination address is output and writing is performed from the DMAC to the destination.

An idle cycle of one clock is always inserted between a read cycle and a write cycle. If the data bus width differs between the transfer source and destination for DMA transfer of two cycles, the operation is performed as follows.

<16-bit data transfer>

<1> Transfer from 32-bit bus → 16-bit bus
A read cycle (the higher 16 bits are in a high-impedance state) is generated, followed by generation of a write cycle (16 bits).

<2> Transfer from 16-/32-bit bus to 8-bit bus
A 16-bit read cycle is generated once, and then an 8-bit write cycle is generated twice.

<3> Transfer from 8-bit bus to 16-/32-bit bus
An 8-bit read cycle is generated twice, and then a 16-bit write cycle is generated once.

<4> Transfer between 16-bit bus and 32-bit bus
A 16-bit read cycle is generated once, and then a 16-bit write cycle is generated once.

For DMA transfer executed to an on-chip peripheral I/O register (transfer source/destination), be sure to specify the same transfer size as the register size. For example, for DMA transfer to an 8-bit register, be sure to specify byte (8-bit) transfer.

Remark The bus width of each transfer target (transfer source/destination) is as follows.

- On-chip peripheral I/O: 16-bit bus width
- Internal RAM: 32-bit bus width
- External memory: 8-bit or 16-bit bus width

18.7 DMA Channel Priorities

The DMA channel priorities are fixed as follows.

DMA channel 0 > DMA channel 1 > DMA channel 2 > DMA channel 3

The priorities are checked for every transfer cycle.

18.8 Time Related to DMA Transfer

The time required to respond to a DMA request, and the minimum number of clocks required for DMA transfer are shown below.

Single transfer: DMA response time (<1>) + Transfer source memory access (<2>) + 1[Note 1] + Transfer destination memory access (<2>)

DMA Cycle		Minimum Number of Execution Clocks
<1> DMA request response time		4 clocks (MIN.) + Noise elimination time[Note 2]
<2> Memory access	External memory access	Depends on connected memory.
	Internal RAM access	2 clocks[Note 3]
	Peripheral I/O register access	3 clocks + Number of wait cycles specified by VSWC register[Note 4]

Notes 1. One clock is always inserted between a read cycle and a write cycle in DMA transfer.

2. If an external interrupt (INTPn) is specified as the trigger to start DMA transfer, noise elimination time is added (n = 0 to 7).

3. Two clocks are required for a DMA cycle.

4. More wait cycles are necessary for accessing a specific peripheral I/O register (for details, see **3.4.9 (2)**).

18.9 DMA Transfer Start Factors

There are two types of DMA transfer start factors, as shown below.

(1) Request by software

If the STGn bit is set to 1 while the DCHCn.TCn bit = 0 and Enn bit = 1 (DMA transfer enabled), DMA transfer is started.

To request the next DMA transfer cycle immediately after that, confirm, by using the DBCn register, that the preceding DMA transfer cycle has been completed, and set the STGn bit to 1 again (n = 0 to 3).

TCn bit = 0, Enn bit = 1
↓
STGn bit = 1 … Starts the first DMA transfer.
↓
Confirm that the contents of the DBCn register have been updated.
STGn bit = 1 … Starts the second DMA transfer.
↓
:
↓
Generation of terminal count … Enn bit = 0, TCn bit = 1, and INTDMAn signal is generated.

(2) Request by on-chip peripheral I/O

If an interrupt request is generated from the on-chip peripheral I/O set by the DTFRn register when the DCHCn.TCn bit = 0 and Enn bit = 1 (DMA transfer enabled), DMA transfer is started.

Cautions 1. Two start factors (software trigger and hardware trigger) cannot be used for one DMA channel. If two start factors are simultaneously generated for one DMA channel, only one of them is valid. The start factor that is valid cannot be identified.

2. A new transfer request that is generated after the preceding DMA transfer request was generated or in the preceding DMA transfer cycle is ignored (cleared).

3. The transfer request interval of the same DMA channel varies depending on the setting of bus wait in the DMA transfer cycle, the start status of the other channels, or the external bus hold request. In particular, as described in Caution 2, a new transfer request that is generated for the same channel before the DMA transfer cycle or during the DMA transfer cycle is ignored. Therefore, the transfer request intervals for the same DMA channel must be sufficiently separated by the system. When the software trigger is used, completion of the DMA transfer cycle that was generated before can be checked by updating the DBCn register.

18.10 DMA Abort Factors

DMA transfer is aborted if a bus hold occurs.

The same applies if transfer is executed between the internal memory/on-chip peripheral I/O and internal memory/on-chip peripheral I/O.

When the bus hold is cleared, DMA transfer is resumed.

18.11 End of DMA Transfer

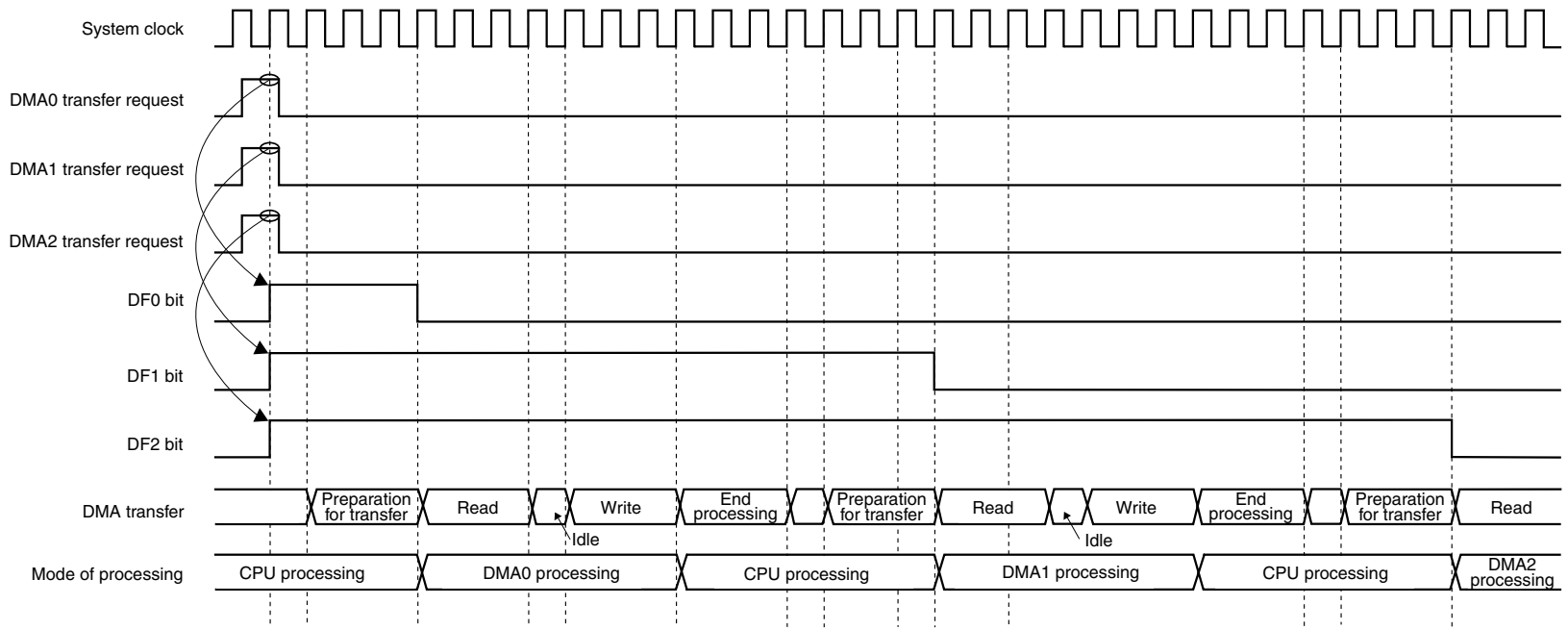
When DMA transfer has been completed the number of times set to the DBCn register and when the DCHCn.Enn bit is cleared to 0 and TCn bit is set to 1, a DMA transfer end interrupt request signal (INTDMAn) is generated for the interrupt controller (INTC) (n = 0 to 3).

The μPD703253Y does not output a terminal count signal to an external device. Therefore, confirm completion of DMA transfer by using the DMA transfer end interrupt or polling the TCn bit.

18.12 Operation Timing

Figures 18-1 to 18-4 show DMA operation timing.

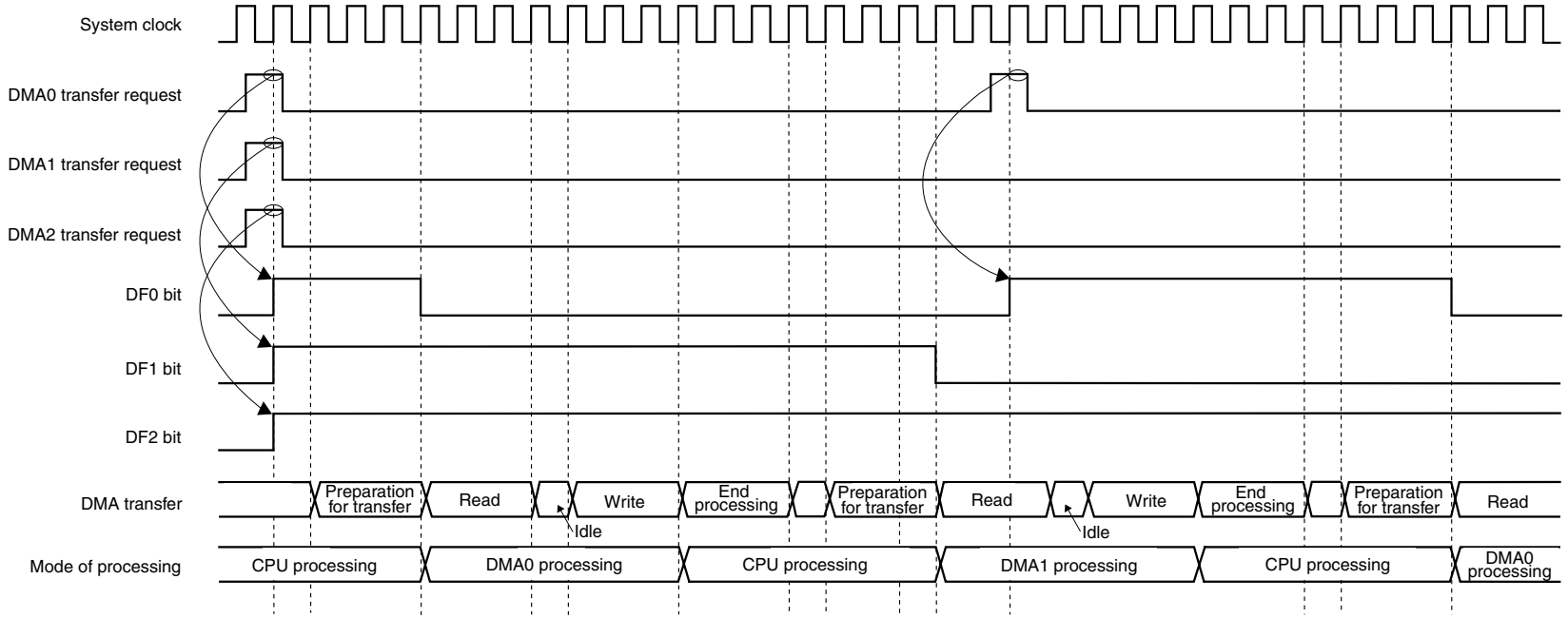
Figure 18-1. Priority of DMA (1)



Remarks 1. Transfer in the order of DMA0 → DMA1 → DMA2

2. In the case of transfer between external memory spaces (multiplexed bus, no wait)

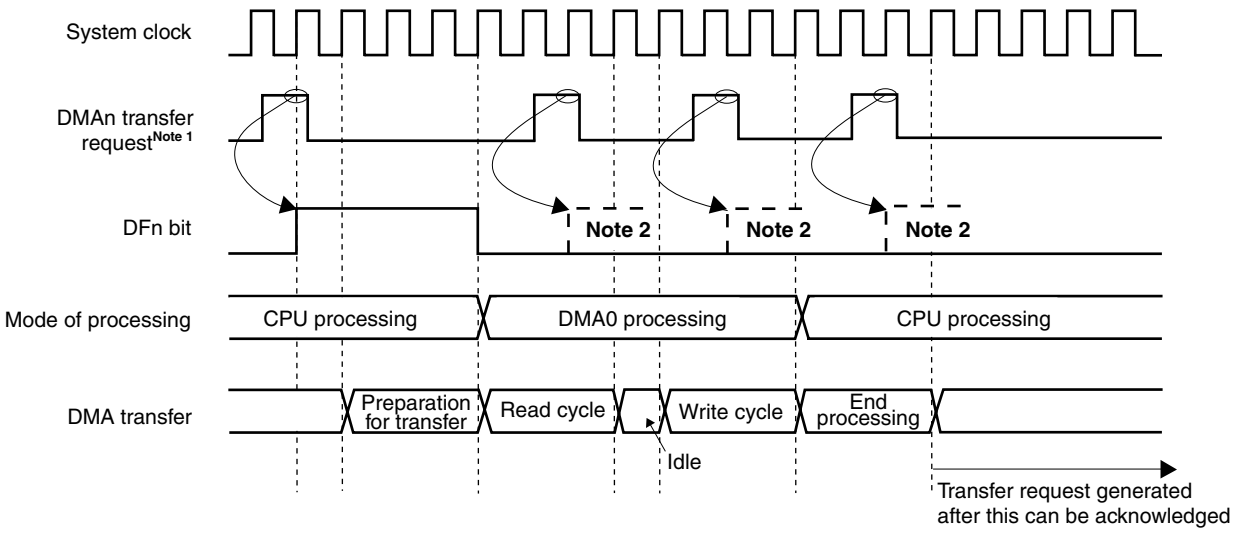
Figure 18-2. Priority of DMA (2)



Remarks 1. Transfer in the order of DMA0 → DMA1 → DMA0 (DMA2 is held pending.)

2. In the case of transfer between external memory spaces (multiplexed bus, no wait)

Figure 18-3. Period in Which DMA Transfer Request Is Ignored (1)

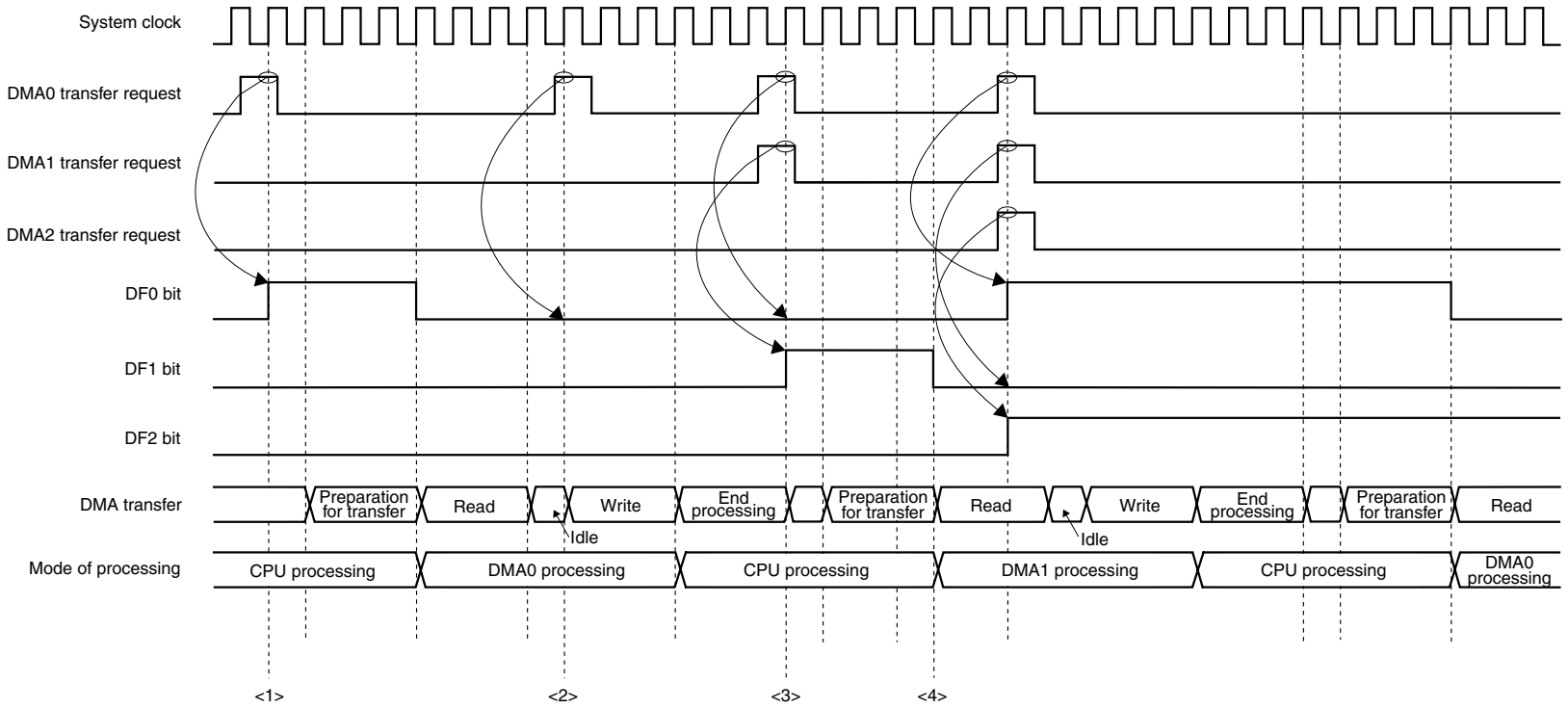


Notes 1. Interrupt from on-chip peripheral I/O, or software trigger (STGn bit)

2. New DMA request of the same channel is ignored between when the first request is generated and the end processing is complete.

Remark In the case of transfer between external memory spaces (multiplexed bus, no wait)

Figure 18-4. Period in Which DMA Transfer Request Is Ignored (2)



<1> DMA0 transfer request

<2> New DMA0 transfer request is generated during DMA0 transfer.

→ A DMA transfer request of the same channel is ignored during DMA transfer.

<3> Requests for DMA0 and DMA1 are generated at the same time.

→ DMA0 request is ignored (a DMA transfer request of the same channel during transfer is ignored).

→ DMA1 request is acknowledged.

<4> Requests for DMA0, DMA1, and DMA2 are generated at the same time.

→ DMA1 request is ignored (a DMA transfer request of the same channel during transfer is ignored).

→ DMA0 request is acknowledged according to priority. DMA2 request is held pending (transfer of DMA2 occurs next).

18.13 Cautions

(1) Caution for VSWC register

When using the DMAC, be sure to set an appropriate value, in accordance with the operating frequency, to the VSWC register.

When the default value (77H) of the VSWC register is used, or if an inappropriate value is set to the VSWC register, the operation is not correctly performed (for details of the VSWC register, see **3.4.9 (1) (a) System wait control register (VSWC)**).

(2) Caution for DMA transfer executed on internal RAM

When executing the following instructions located in the internal RAM, do not execute a DMA transfer that transfers data to/from the internal RAM (transfer source/destination), because the CPU may not operate correctly afterward.

- Bit manipulation instruction located in internal RAM (SET1, CLR1, or NOT1)
- Data access instruction to misaligned address located in internal RAM

Conversely, when executing a DMA transfer to transfer data to/from the internal RAM (transfer source/destination), do not execute the above two instructions.

(3) Caution for reading DCHCn.TCn bit (n = 0 to 3)

The TCn bit is cleared to 0 when it is read, but it is not automatically cleared even if it is read at a specific timing. To accurately clear the TCn bit, add the following processing.

(a) When waiting for completion of DMA transfer by polling TCn bit

Confirm that the TCn bit has been set to 1 (after TCn bit = 1 is read), and then read the TCn bit three more times.

(b) When reading TCn bit in interrupt servicing routine

Execute reading the TCn bit three times.

(4) DMA transfer initialization procedure (setting DCHCn.INITn bit to 1)

Even if the INITn bit is set to 1 when the channel executing DMA transfer is to be initialized, the channel may not be initialized. To accurately initialize the channel, execute either of the following two procedures.

(a) Temporarily stop transfer of all DMA channels

Initialize the channel executing DMA transfer using the procedure in <1> to <7> below.

Note, however, that TCn bit is cleared to 0 when step <5> is executed. Make sure that the other processing programs do not expect that the TCn bit is 1.

<1> Disable interrupts (DI).

<2> Read the DCHCn.Enn bit of DMA channels other than the one to be forcibly terminated, and transfer the value to a general-purpose register.

<3> Clear the Enn bit of the DMA channels used (including the channel to be forcibly terminated) to 0. To clear the Enn bit of the last DMA channel, execute the clear instruction twice. If the target of DMA transfer (transfer source/destination) is the internal RAM, execute the instruction three times.

Example: Execute instructions in the following order if channels 0, 1, and 2 are used (if the target of transfer is not the internal RAM).

- Clear DCHC0.E00 bit to 0.
- Clear DCHC1.E11 bit to 0.
- Clear DCHC2.E22 bit to 0.
- Clear DCHC2.E22 bit to 0 again.

<4> Set the INITn bit of the channel to be forcibly terminated to 1.

<5> Read the TCn bit of each channel not to be forcibly terminated. If both the TCn bit and the Enn bit read in <2> are 1 (logical product (AND) is 1), clear the saved Enn bit to 0.

<6> After the operation in <5>, write the Enn bit value to the DCHCn register.

<7> Enable interrupts (EI).

Caution Be sure to execute step <5> above to prevent illegal setting of the Enn bit of the channels whose DMA transfer has been normally completed between <2> and <3>.

(b) Repeatedly execute setting INITn bit until transfer is forcibly terminated correctly

<1> Suppress a request from the DMA request source of the channel to be forcibly terminated (stop operation of the on-chip peripheral I/O).
<2> Check that the DMA transfer request of the channel to be forcibly terminated is not held pending, by using the DTFRn.DFn bit. If a DMA transfer request is held pending, wait until execution of the pending request is completed.
<3> When it has been confirmed that the DMA request of the channel to be forcibly terminated is not held pending, clear the Enn bit to 0.
<4> Again, clear the Enn bit of the channel to be forcibly terminated.
If the target of transfer for the channel to be forcibly terminated (transfer source/destination) is the internal RAM, execute this operation once more.
<5> Copy the initial number of transfers of the channel to be forcibly terminated to a general-purpose register.
<6> Set the INITn bit of the channel to be forcibly terminated to 1.
<7> Read the value of the DBCn register of the channel to be forcibly terminated, and compare it with the value copied in <5>. If the two values do not match, repeat operations <6> and <7>.

Remarks 1. When the value of the DBCn register is read in <7>, the initial number of transfers is read if forced termination has been correctly completed. If not, the remaining number of transfers is read.
2. Note that method (b) may take a long time if the application frequently uses DMA transfer for a channel other than the DMA channel to be forcibly terminated.

(5) Procedure of temporarily stopping DMA transfer (clearing Enn bit)

Stop and resume the DMA transfer under execution using the following procedure.

<1> Suppress a transfer request from the DMA request source (stop the operation of the on-chip peripheral I/O).
<2> Check the DMA transfer request is not held pending, by using the DFn bit (check if the DFn bit = 0).
If a request is pending, wait until execution of the pending DMA transfer request is completed.
<3> If it has been confirmed that no DMA transfer request is held pending, clear the Enn bit to 0 (this operation stops DMA transfer).
<4> Set the Enn bit to 1 to resume DMA transfer.
<5> Resume the operation of the DMA request source that has been stopped (start the operation of the on-chip peripheral I/O).

(6) Memory boundary

The operation is not guaranteed if the address of the transfer source or destination exceeds the area of the DMA target (external memory, internal RAM, or on-chip peripheral I/O) during DMA transfer.

(7) Transferring misaligned data

DMA transfer of misaligned data with a 16-bit bus width is not supported.
If an odd address is specified as the transfer source or destination, the least significant bit of the address is forcibly assumed to be 0.

(8) Bus arbitration for CPU

Because the DMA controller has a higher priority bus mastership than the CPU, a CPU access that takes place during DMA transfer is held pending until the DMA transfer cycle is completed and the bus is released to the CPU.

However, the CPU can access the internal ROM and internal RAM for which DMA transfer is not being executed.

- The CPU can access the internal ROM and internal RAM when DMA transfer is being executed between the external memory and on-chip peripheral I/O.
- The CPU can access the internal ROM when DMA transfer is being executed between the on-chip peripheral I/O and internal RAM.

(9) Registers/bits that must not be rewritten during DMA operation

Set the following registers at the following timing when a DMA operation is not under execution.

[Registers]

- DSAnH, DSAnL, DDAnH, DDAnL, DBCn, and DADCn registers
- DTFRn.IFCn5 to DTFRn.IFCn0 bits

[Timing of setting]

- Period from after reset to start of the first DMA transfer
- Time after channel initialization to start of DMA transfer
- Period from after completion of DMA transfer (TCn bit = 1) to start of the next DMA transfer

(10) Be sure to set the following register bits to 0.

- Bits 14 to 10 of DSAnH register
- Bits 14 to 10 of DDAnH register
- Bits 15, 13 to 8, and 3 to 0 of DADCn register
- Bits 6 to 3 of DCHCn register

(11) DMA start factor

Care must be exercised when setting the same start trigger for multiple DMA channels.

If DMA transfers via such DMA channels are activated, the DMA channel with a lower priority may be acknowledged prior to the DMA channel with a higher priority.

(12) Read values of DSAn and DDAn registers

Values in the middle of updating may be read from the DSAn and DDAn registers during DMA transfer (n = 0 to 3).

For example, if the DSAnH register and then the DSAnL register are read when the DMA transfer source address (DSAn register) is 0000FFFFH and the count direction is incremental (DADCn.SAD1 and DADCn.SAD0 bits = 00), the value of the DSAn register differs as follows, depending on whether DMA transfer is executed immediately after the DSAnH register is read.

(a) If DMA transfer does not occur while DSAn register is read

<1> Read value of DSAnH register: DSAnH = 0000H

<2> Read value of DSAnL register: DSAnL = FFFFH

(b) If DMA transfer occurs while DSAn register is read

<1> Read value of DSAnH register: DSAnH = 0000H

<2> Occurrence of DMA transfer

<3> Incrementing DSAn register: DSAn = 00100000H

<4> Read value of DSAnL register: DSAnL = 0000H

CHAPTER 19 INTERRUPT/EXCEPTION PROCESSING FUNCTION

The V850ES/SG1 is provided with a dedicated interrupt controller (INTC) for interrupt servicing and can process a
<R> total of 41 to 52 interrupt requests.

An interrupt is an event that occurs independently of program execution, and an exception is an event whose occurrence is dependent on program execution.

The V850ES/SG1 can process interrupt request signals from the on-chip peripheral hardware and external sources. Moreover, exception processing can be started by the TRAP instruction (software exception) or by generation of an exception event (i.e. fetching of an illegal opcode) (exception trap).

19.1 Features

○ Interrupts
- Non-maskable interrupts: 2 sources
<R> • Maskable interrupts: External: 8, Internal: 31/35/42 sources
- 8 levels of programmable priorities (maskable interrupts)
- Multiple interrupt control according to priority
- Masks can be specified for each maskable interrupt request.
- Noise elimination, edge detection, and valid edge specification for external interrupt request signals.

○ Exceptions
- Software exceptions: 32 sources
- Exception trap: 2 sources (illegal opcode exception)

Interrupt/exception sources are listed in Table 19-1.

Table 19-1. Interrupt Source List (1/4)

Type	Classification	Default Priority	Name	Trigger	Generating Unit	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Reset	Interrupt	–	RESET	$\overline{\text{RESET}}$ pin input Reset input by internal source	RESET	0000H	00000000H	Undefined	–
Non-maskable	Interrupt	–	NMI	NMI pin valid edge input	Pin	0010H	00000010H	nextPC	–
		–	INTWDT2	WDT2 overflow	WDT2	0020H	00000020H	**Note 1**	–
Software exception	Exception	–	TRAP0n[Note 2]	TRAP instruction	–	004nH[Note 2]	00000040H	nextPC	–
		–	TRAP1n[Note 2]	TRAP instruction	–	005nH[Note 2]	00000050H	nextPC	–
Exception trap	Exception	–	ILGOP/ DBG0	Illegal opcode/ DBTRAP instruction	–	0060H	00000060H	nextPC	–

Notes 1. For the restoring in the case of INTWDT2, see **19.2.2 (2) INTWDT2 signal**.

2. n = 0 to FH

Table 19-1. Interrupt Source List (2/4)

Type	Classification	Default Priority	Name	Trigger	Generating Unit	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Maskable	Interrupt	1	INTP0	External interrupt pin input edge detection (INTP0)	Pin	0090H	00000090H	nextPC	PIC0
		2	INTP1	External interrupt pin input edge detection (INTP1)	Pin	00A0H	000000A0H	nextPC	PIC1
		3	INTP2	External interrupt pin input edge detection (INTP2)	Pin	00B0H	000000B0H	nextPC	PIC2
		4	INTP3	External interrupt pin input edge detection (INTP3)	Pin	00C0H	000000C0H	nextPC	PIC3
		5	INTP4	External interrupt pin input edge detection (INTP4)	Pin	00D0H	000000D0H	nextPC	PIC4
		6	INTP5	External interrupt pin input edge detection (INTP5)	Pin	00E0H	000000E0H	nextPC	PIC5
		7	INTP6	External interrupt pin input edge detection (INTP6)	Pin	00F0H	000000F0H	nextPC	PIC6
		8	INTP7	External interrupt pin input edge detection (INTP7)	Pin	0100H	00000100H	nextPC	PIC7
		14	INTTP0OV	TMP0 overflow	TMP0	0160H	00000160H	nextPC	TP0OVIC
		15	INTTP0CC0	TMP0 capture 0/ compare 0 match	TMP0	0170H	00000170H	nextPC	TP0CCIC0
		16	INTTP0CC1	TMP0 capture 1/ compare 1 match	TMP0	0180H	00000180H	nextPC	TP0CCIC1
		17	INTTP1OV	TMP1 overflow	TMP1	0190H	00000190H	nextPC	TP1OVIC
		18	INTTP1CC0	TMP1 capture 0/ compare 0 match	TMP1	01A0H	000001A0H	nextPC	TP1CCIC0
		19	INTTP1CC1	TMP1 capture 1/ compare 1 match	TMP1	01B0H	000001B0H	nextPC	TP1CCIC1
		20	INTTP2OV	TMP2 overflow	TMP2	01C0H	000001C0H	nextPC	TP2OVIC
		21	INTTP2CC0	TMP2 capture 0/ compare 0 match	TMP2	01D0H	000001D0H	nextPC	TP2CCIC0
		22	INTTP2CC1	TMP2 capture 1/ compare 1 match	TMP2	01E0H	000001E0H	nextPC	TP2CCIC1
		23	INTTP3OV	TMP3 overflow	TMP3	01F0H	000001F0H	nextPC	TP3OVIC
		24	INTTP3CC0	TMP3 capture 0/ compare 0 match	TMP3	0200H	00000200H	nextPC	TP3CCIC0
		25	INTTP3CC1	TMP3 capture 1/ compare 1 match	TMP3	0210H	00000210H	nextPC	TP3CCIC1
		26	INTTP4OV	TMP4 overflow	TMP4	0220H	00000220H	nextPC	TP4OVIC
		27	INTTP4CC0	TMP4 capture 0/ compare 0 match	TMP4	0230H	00000230H	nextPC	TP4CCIC0
		28	INTTP4CC1	TMP4 capture 1/ compare 1 match	TMP4	0240H	00000240H	nextPC	TP4CCIC1
<R>		29	INTTP5OV[Note]	TMP5 overflow	TMP5	0250H	00000250H	nextPC	TP5OVIC
<R>		30	INTTP5CC0[Note]	TMP5 capture 0/ compare 0 match	TMP5	0260H	00000260H	nextPC	TP5CCIC0
<R>		31	INTTP5CC1[Note]	TMP5 capture 1/ compare 1 match	TMP5	0270H	00000270H	nextPC	TP5CCIC1

<R> **Note** μPD703253Y only

Table 19-1. Interrupt Source List (3/4)

Type	Classification	Default Priority	Name	Trigger	Generating Unit	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Maskable	Interrupt	32	INTTM0EQ0	TMM0 compare match	TMM0	0280H	00000280H	nextPC	TM0EQIC0
		33	INTCB0R/ INTIIC1	CSIB0 reception completion/ CSIB0 reception error/ IIC1 transfer completion	CSIB0/ IIC1	0290H	00000290H	nextPC	CB0RIC/ IICIC1
		34	INTCB0T	CSIB0 consecutive transmission write enable	CSIB0	02A0H	000002A0H	nextPC	CB0TIC
		35	INTCB1R	CSIB1 reception completion/ CSIB1 reception error	CSIB1	02B0H	000002B0H	nextPC	CB1RIC
		36	INTCB1T	CSIB1 consecutive transmission write enable	CSIB1	02C0H	000002C0H	nextPC	CB1TIC
		39	INTCB3R	CSIB3 reception completion/ CSIB3 reception error	CSIB3	02F0H	000002F0H	nextPC	CB3RIC
		40	INTCB3T	CSIB3 consecutive transmission write enable	CSIB3	0300H	00000300H	nextPC	CB3TIC
		41	INTUA0R	UARTA0 reception completion/ UART0 reception error	UARTA0	0310H	00000310H	nextPC	UA0RIC
		42	INTUA0T	UARTA0 consecutive transmission enable	UARTA0	0320H	00000320H	nextPC	UA0TIC
		43	INTUA1R	UARTA1 reception completion/ UARTA1 reception error	UARTA1	0330H	00000330H	nextPC	UA1RIC
		44	INTUA1T	UARTA1 consecutive transmission enable	UARTA1	0340H	00000340H	nextPC	UA1TIC
		45	INTIIC0	IIC0 transfer completion	IIC0	0350H	00000350H	nextPC	IICIC0
		47	INTAD	A/D conversion completion	A/D	0370H	00000370H	nextPC	ADIC
<R>		48	INTDMA0[Note]	DMA0 transfer completion	DMA	0380H	00000380H	nextPC	DMAIC0
<R>		49	INTDMA1[Note]	DMA1 transfer completion	DMA	0390H	00000390H	nextPC	DMAIC1
<R>		50	INTDMA2[Note]	DMA2 transfer completion	DMA	03A0H	000003A0H	nextPC	DMAIC2
<R>		51	INTDMA3[Note]	DMA3 transfer completion	DMA	03B0H	000003B0H	nextPC	DMAIC3
		52	INTKR	Key return interrupt	KR	03C0H	000003C0H	nextPC	KRIC
		53	INTWTI	Watch timer interval	WT	03D0H	000003D0H	nextPC	WTIIC
		54	INTWT	Watch timer reference time	WT	03E0H	000003E0H	nextPC	WTIC
<R>		55	INTC0ERR[Note 1]/ INTERR[Note 2]	AFCAN0 error/IEBus error	AFCAN0/ IEBus	03F0H	000003F0H	nextPC	ERRIC0/ ERRIC
<R>		56	INTC0WUP[Note 1]/ INTSTA[Note 2]	AFCAN0 wakeup/ IEBus status	AFCAN0/ IEBus	0400H	00000400H	nextPC	WUPIC0/ STSAIC
<R>		57	INTC0REC[Note 1]/ INTIE1[Note 2]	AFCAN0 reception/ IEBus data interrupt	AFCAN0/ IEBus	0410H	00000410H	nextPC	RECIC0/ IEIC1
<R>		58	INTC0TRX[Note 1]/ INTIE2[Note 2]	AFCAN0 transmission/ IEBus error/IEBus status	AFCAN0/ IEBus	0420H	00000420H	nextPC	TRXIC0/ IEIC2

<R> **Notes 1.** μPD703253Y only

2. μPD703252Y only

Remarks 1. Default Priority: The priority order when two or more maskable interrupt requests occur at the same time. The highest priority is 1.

The priority order of non-maskable interrupt is INTWDT2 > NMI.

Restored PC: The value of the program counter (PC) saved to EIPC, FEPC, or DBPC when interrupt servicing is started. Note, however, that the restored PC when a non-maskable or maskable interrupt is acknowledged while one of the following instructions is being executed does not become the nextPC (if an interrupt is acknowledged during interrupt execution, execution stops, and then resumes after the interrupt servicing has finished).

- Load instructions (SLD.B, SLD.BU, SLD.H, SLD.HU, SLD.W)
- Division instructions (DIV, DIVH, DIVU, DIVHU)
- PREPARE, DISPOSE instructions (only if an interrupt is generated before the stack pointer is updated)

nextPC: The PC value that starts the processing following interrupt/exception processing.

2. The execution address of the illegal instruction when an illegal opcode exception occurs is calculated by (Restored PC − 4).

19.2 Non-Maskable Interrupts

A non-maskable interrupt request signal is acknowledged unconditionally, even when interrupts are in the interrupt disabled (DI) status. An NMI is not subject to priority control and takes precedence over all the other interrupt request signals.

This product has the following two non-maskable interrupt request signals.

- NMI pin input (NMI)
- Non-maskable interrupt request signal generated by overflow of watchdog timer (INTWDT2)

The valid edge of the NMI pin can be selected from four types: “rising edge”, “falling edge”, “both edges”, and “no edge detection”.

The non-maskable interrupt request signal generated by overflow of watchdog timer 2 (INTWDT2) functions when the WDTM2.WDM21 and WDTM2.WDM20 bits are set to “01”.

If two or more non-maskable interrupt request signals occur at the same time, the interrupt with the higher priority is serviced, as follows (the interrupt request signal with the lower priority is ignored).

INTWDT2 > NMI

If a new NMI or INTWDT2 request signal is issued while an NMI is being serviced, it is serviced as follows.

(1) If new NMI request signal is issued while NMI is being serviced

The new NMI request signal is held pending, regardless of the value of the PSW.NP bit. The pending NMI request signal is acknowledged after the NMI currently under execution has been serviced (after the RETI instruction has been executed).

(2) If INTWDT2 request signal is issued while NMI is being serviced

The INTWDT2 request signal is held pending if the NP bit remains set (1) while the NMI is being serviced. The pending INTWDT2 request signal is acknowledged after the NMI currently under execution has been serviced (after the RETI instruction has been executed).

If the NP bit is cleared (0) while the NMI is being serviced, the newly generated INTWDT2 request signal is executed (the NMI servicing is stopped).

Caution For the non-maskable interrupt servicing executed by the non-maskable interrupt request signal (INTWDT2), see 19.2.2 (2) INTWDT2 signal.

Figure 19-1. Non-Maskable Interrupt Request Signal Acknowledgment Operation (1/2)

(a) NMI and INTWDT2 request signals generated at the same time

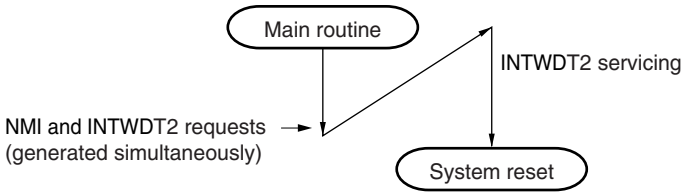
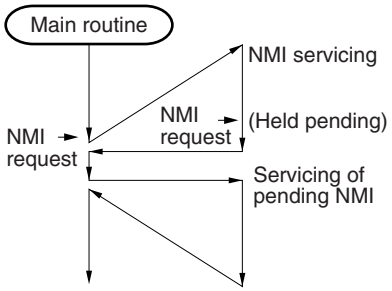
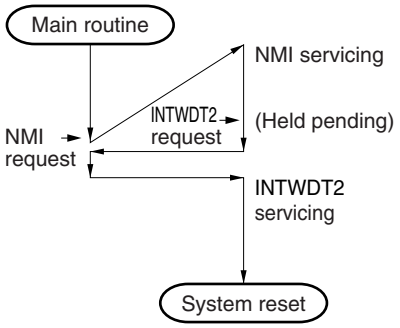
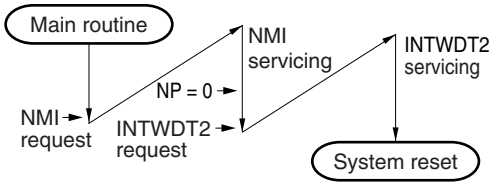
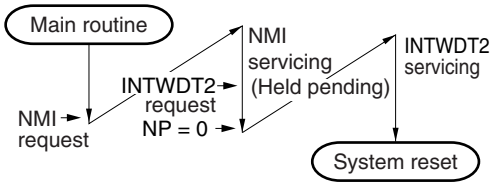
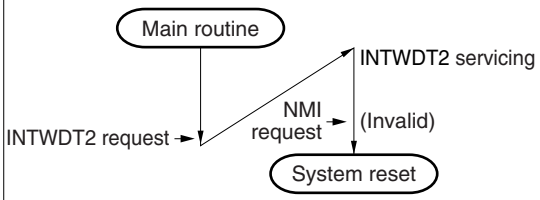
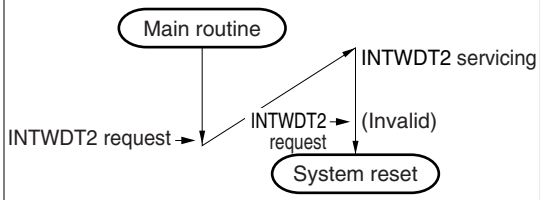


Figure 19-1. Non-Maskable Interrupt Request Signal Acknowledgment Operation (2/2)

(b) Non-maskable interrupt request signal generated during non-maskable interrupt servicing

Non-maskable interrupt being serviced	Non-maskable interrupt request signal generated during non-maskable interrupt servicing	
	NMI	INTWDT2
NMI	• NMI request generated during NMI servicing 	• INTWDT2 request generated during NMI servicing (NP bit = 1 retained before INTWDT2 request)  • INTWDT2 request generated during NMI servicing (NP bit = 0 set before INTWDT2 request)  • INTWDT2 request generated during NMI servicing (NP = 0 set after INTWDT2 request) 
INTWDT2	• NMI request generated during INTWDT2 servicing 	• INTWDT2 request generated during INTWDT2 servicing 

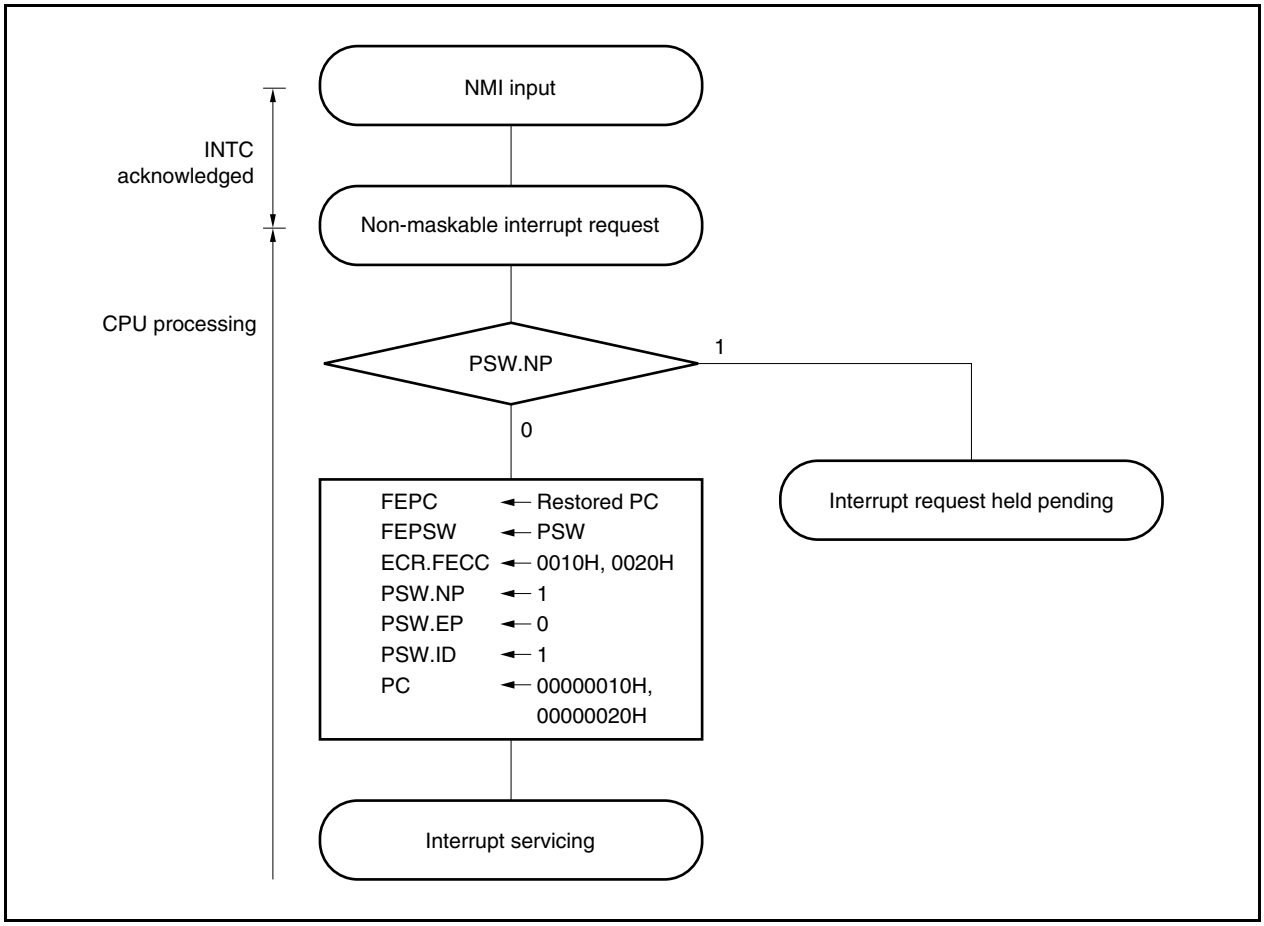
19.2.1 Operation

If a non-maskable interrupt request signal is generated, the CPU performs the following processing, and transfers control to the handler routine.

<1> Saves the restored PC to FEPC.
<2> Saves the current PSW to FEPSW.
<3> Writes exception code (0010H, 0020H) to the higher halfword (FECC) of ECR.
<4> Sets the PSW.NP and PSW.ID bits to 1 and clears the PSW.EP bit to 0.
<5> Sets the handler address (00000010H, 00000020H) corresponding to the non-maskable interrupt to the PC, and transfers control.

The servicing configuration of a non-maskable interrupt is shown in Figure 19-2.

Figure 19-2. Servicing Configuration of Non-Maskable Interrupt



19.2.2 Restoration

(1) From NMI pin input

Execution is restored from the NMI servicing by the RETI instruction.

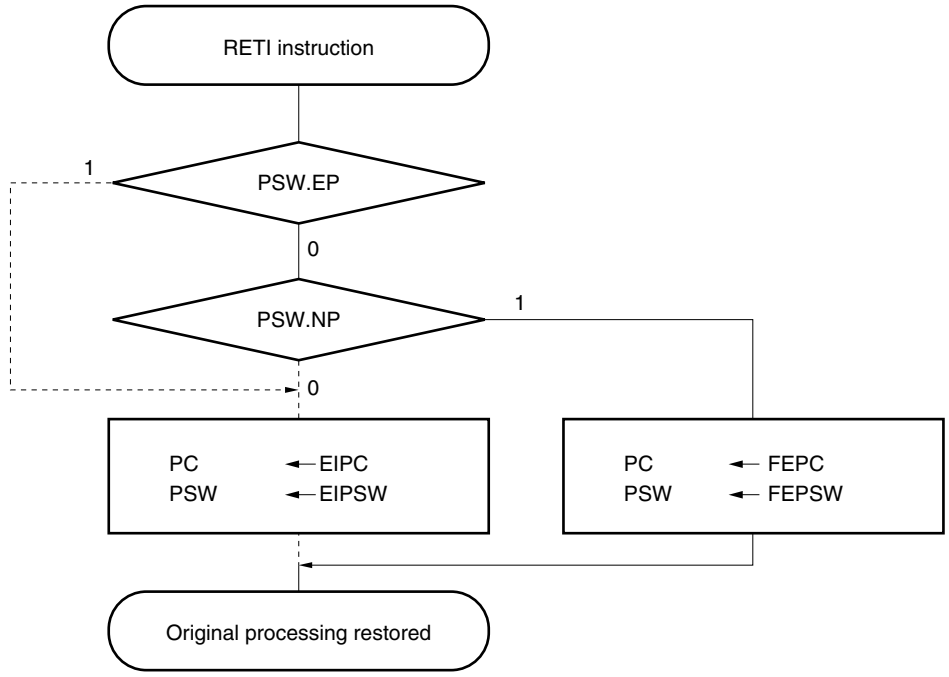
When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

<1> Loads the restored PC and PSW from FEPC and FEPSW, respectively, because the PSW.EP bit is 0 and the PSW.NP bit is 1.

<2> Transfers control back to the address of the restored PC and PSW.

Figure 19-3 illustrates the processing of the RETI instruction.

Figure 19-3. RETI Instruction Processing



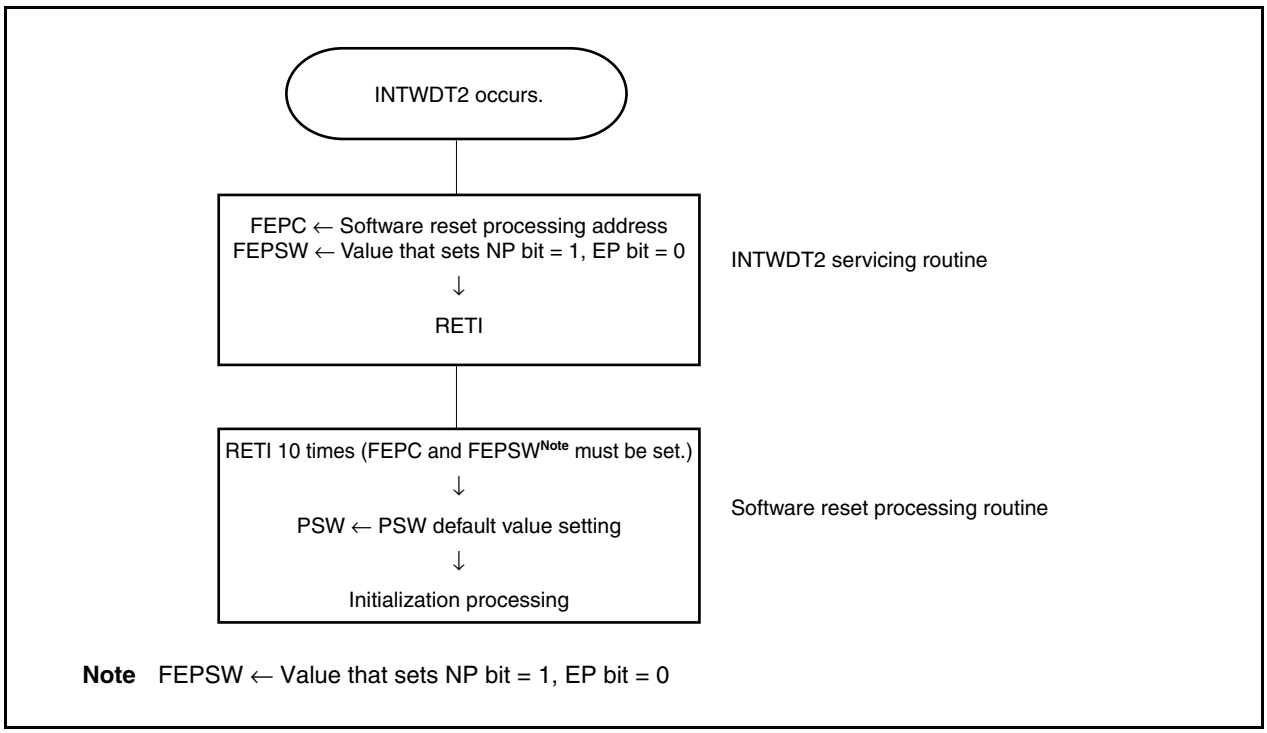
Caution When the EP and NP bits are changed by the LDSR instruction during non-maskable interrupt servicing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 0 and the NP bit back to 1 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

(2) From INTWDT2 signal

Restoring from non-maskable interrupt servicing executed by the non-maskable interrupt request (INTWDT2) by using the RETI instruction is disabled. Execute the following software reset processing.

Figure 19-4. Software Reset Processing



Note FEPSW ← Value that sets NP bit = 1, EP bit = 0

19.2.3 NP flag

The NP flag is a status flag that indicates that non-maskable interrupt servicing is under execution.

This flag is set when a non-maskable interrupt request signal has been acknowledged, and masks non-maskable interrupt requests to prohibit multiple interrupts from being acknowledged.

After reset: 00000020H

	31 … 8	7	6	5	4	3	2	1	0
PSW	0	**NP**	EP	ID	SAT	CY	OV	S	Z

NP	NMI interrupt servicing status
0	No NMI interrupt servicing
1	NMI interrupt currently being serviced

19.3 Maskable Interrupts

<R> Maskable interrupt request signals can be masked by interrupt control registers. The V850ES/SG1 has 39 to 50 maskable interrupt sources.

If two or more maskable interrupt request signals are generated at the same time, they are acknowledged according to the default priority. In addition to the default priority, eight levels of priorities can be specified by using the interrupt control registers (programmable priority control).

When an interrupt request signal has been acknowledged, the acknowledgment of other maskable interrupt request signals is disabled and the interrupt disabled (DI) status is set.

When the EI instruction is executed in an interrupt service routine, the interrupt enabled (EI) status is set, which enables servicing of interrupts having a higher priority than the interrupt request signal in progress (specified by the interrupt control register). Note that only interrupts with a higher priority will have this capability; interrupts with the same priority level cannot be nested.

To enable multiple interrupts, however, save EIPC and EIPSW to memory or general-purpose registers before executing the EI instruction, and execute the DI instruction before the RETI instruction to restore the original values of EIPC and EIPSW.

19.3.1 Operation

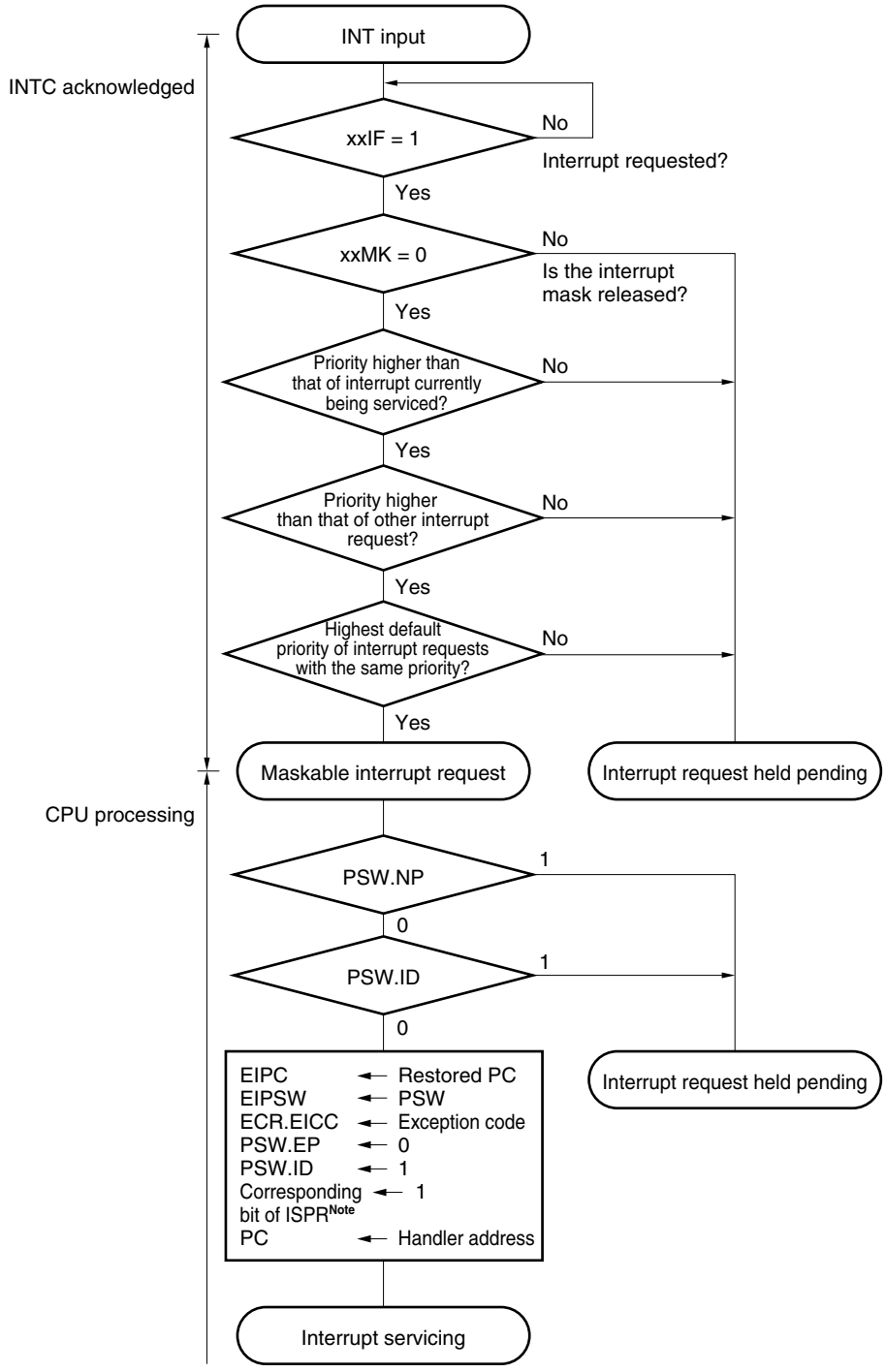
If a maskable interrupt occurs, the CPU performs the following processing, and transfers control to a handler routine.

<1> Saves the restored PC to EIPC.
<2> Saves the current PSW to EIPSW.
<3> Writes an exception code to the lower halfword of ECR (EICC).
<4> Sets the PSW. ID bit to 1 and clears the PSW. EP bit to 0.
<5> Sets the handler address corresponding to each interrupt to the PC, and transfers control.

The maskable interrupt request signal masked by INTC and the maskable interrupt request signal generated while another interrupt is being serviced (while the PSW.NP bit = 1 or the PSW.ID bit = 1) are held pending inside INTC. In this case, servicing a new maskable interrupt is started in accordance with the priority of the pending maskable interrupt request signal if either the maskable interrupt is unmasked or the NP and ID bits are cleared to 0 by using the RETI or LDSR instruction.

How maskable interrupts are serviced is illustrated below.

Figure 19-5. Maskable Interrupt Servicing



Note For the ISPR register, see **19.3.6 In-service priority register (ISPR)**.

19.3.2 Restoration

Restoration from maskable interrupt servicing is carried out by the RETI instruction.

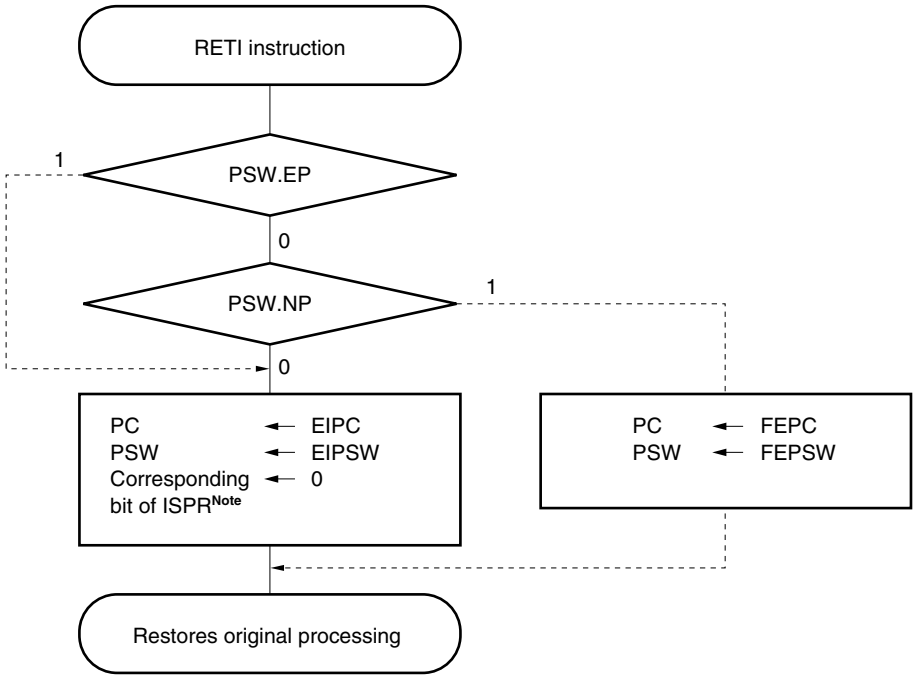
When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

<1> Loads the restored PC and PSW from EIPC and EIPSW because the PSW.EP bit is 0 and the PSW.NP bit is 0.

<2> Transfers control back to the address of the restored PC and PSW.

Figure 19-6 illustrates the processing of the RETI instruction.

Figure 19-6. RETI Instruction Processing



Note For the ISPR register, see **19.3.6 In-service priority register (ISPR)**.

Caution When the EP and NP bits are changed by the LDSR instruction during maskable interrupt servicing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 0 and the NP bit back to 0 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

19.3.3 Priorities of maskable interrupts

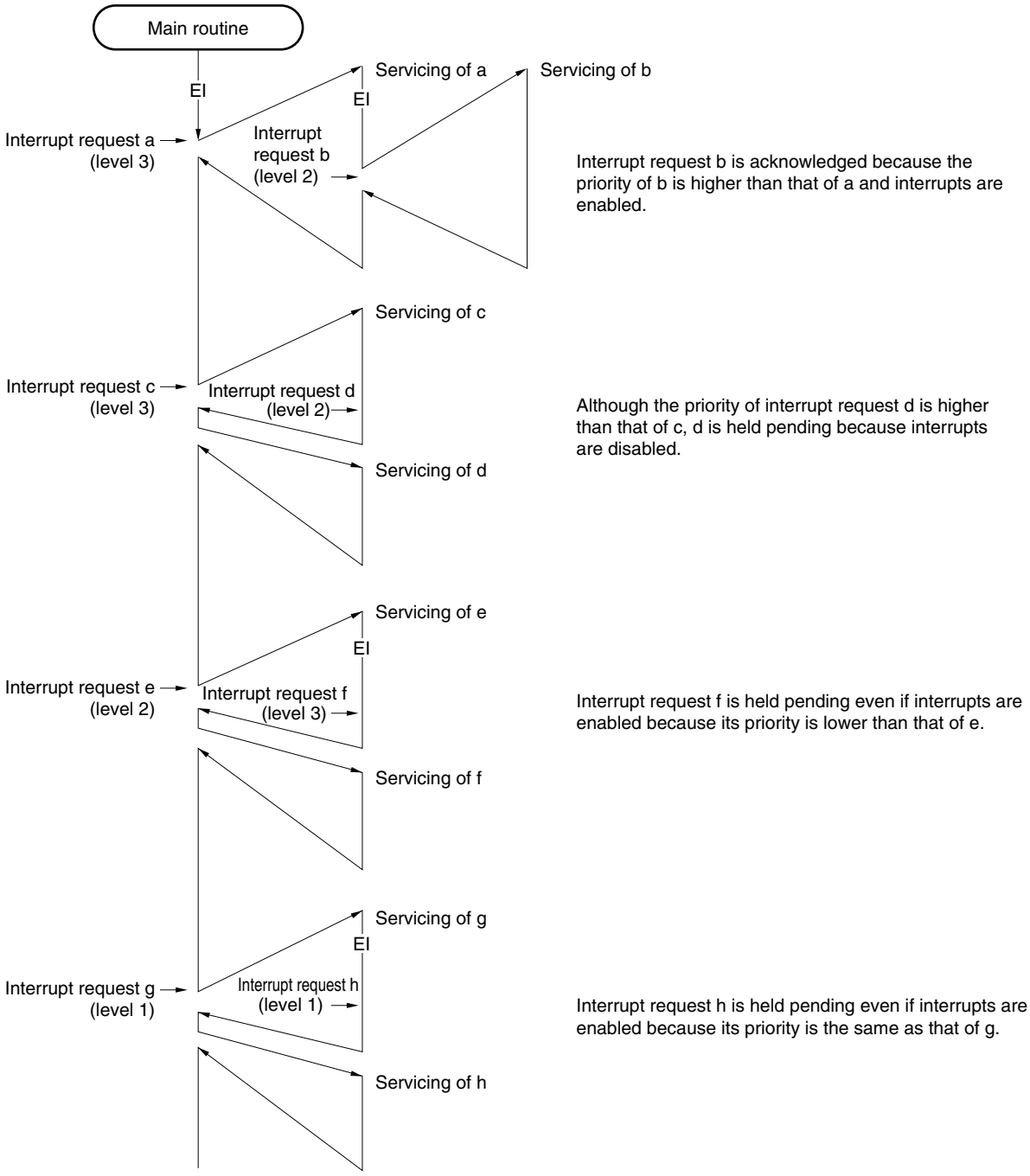
The INTC performs multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced. Multiple interrupts can be controlled by priority levels.

There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels that are specified by the interrupt priority level specification bit (xxPRn) of the interrupt control register (xxICn). When two or more interrupts having the same priority level specified by the xxPRn bit are generated at the same time, interrupt request signals are serviced in order depending on the priority level allocated to each interrupt request type (default priority level) beforehand. For more information, see **Table 19-1 Interrupt/Exception Source List**. The programmable priority control customizes interrupt request signals into eight levels by setting the priority level specification flag.

Note that when an interrupt request signal is acknowledged, the PSW.ID flag is automatically set to 1. Therefore, when multiple interrupts are to be used, clear the ID flag to 0 beforehand (for example, by placing the EI instruction in the interrupt service program) to set the interrupt enable mode.

Remark xx: Identification name of each peripheral unit (see **Table 19-2 Interrupt Control Register (xxICn)**)
n: Peripheral unit number (see **Table 19-2 Interrupt Control Register (xxICn)**).

Figure 19-7. Example of Processing in Which Another Interrupt Request Signal Is Issued While an Interrupt Is Being Serviced (1/2)

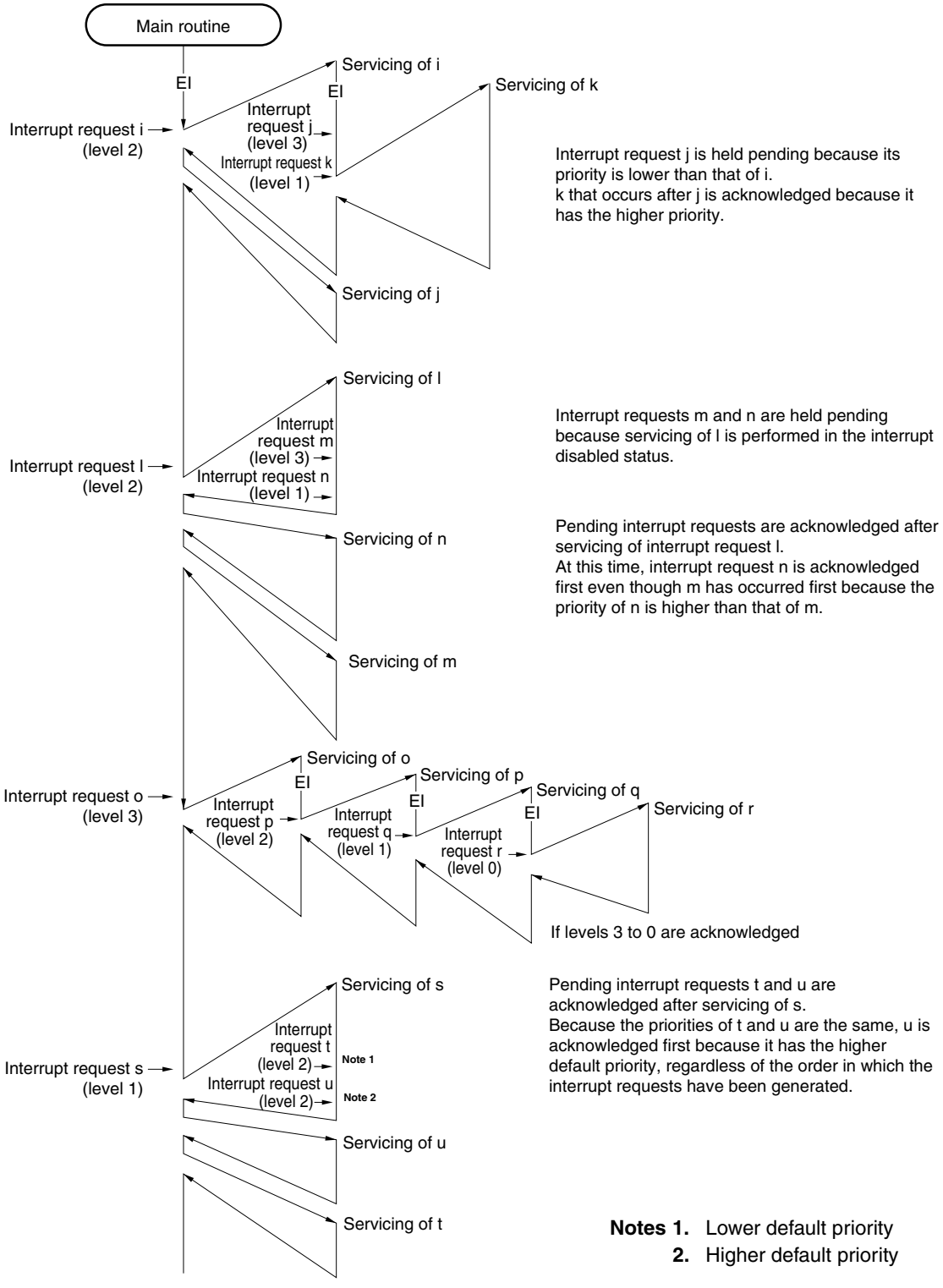


Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Remarks 1. a to u in the figure are the temporary names of interrupt request signals shown for the sake of explanation.

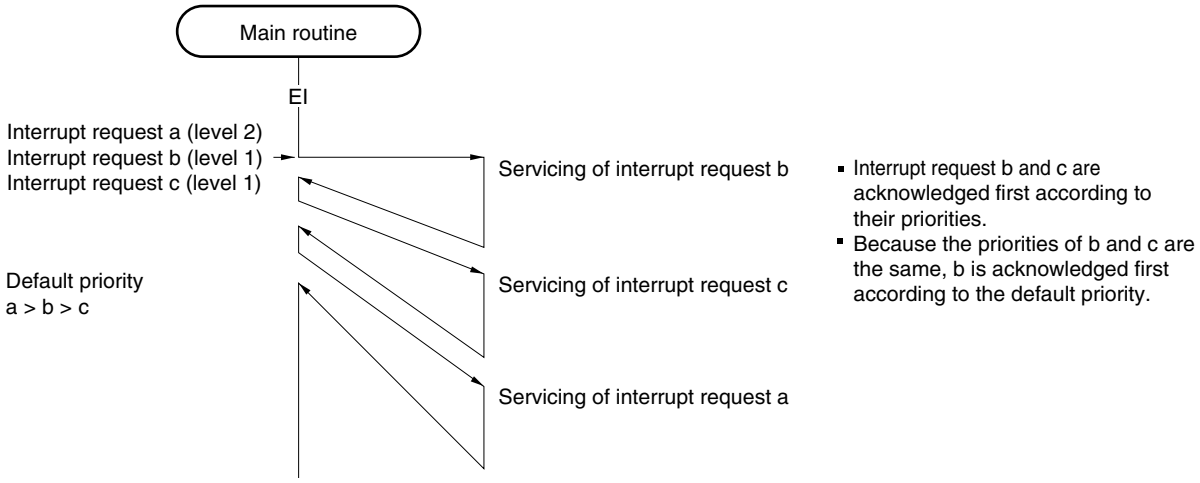
2. The default priority in the figure indicates the relative priority between two interrupt request signals.

Figure 19-7. Example of Processing in Which Another Interrupt Request Signal Is Issued While an Interrupt Is Being Serviced (2/2)



Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Figure 19-8. Example of Servicing Interrupt Request Signals Simultaneously Generated



Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

Remarks 1. a to c in the figure are the temporary names of interrupt request signals shown for the sake of explanation.

2. The default priority in the figure indicates the relative priority between two interrupt request signals.

19.3.4 Interrupt control register (xxICn)

The xxICn register is assigned to each interrupt request signal (maskable interrupt) and sets the control conditions for each maskable interrupt request.

This register can be read or written in 8-bit or 1-bit units.

Reset input sets this register to 47H.

Cautions 1. Disable interrupts (DI) or mask the interrupt to read the xxICn.xxIFn bit. If the xxIFn bit is read while interrupts are enabled (EI) or while the interrupt is unmasked, the correct value may not be read when acknowledging an interrupt and reading the bit conflict.

<R> **2. When manipulating the xxICn.xxMKn bit while interrupt requests may occur (including the state in which interrupts are disabled (DI)), be sure to use a bit manipulation instruction or use the IMRm.xxMKn bit (m = 0 to 3).**

After reset: 47H R/W Address: FFFFF112H to FFFFF184H

	<7>	<6>	5	4	3	2	1	0
xxICn	xxIFn	xxMKn	0	0	0	xxPRn2	xxPRn1	xxPRn0

xxIFn	Interrupt request flag[Note]
0	Interrupt request not issued
1	Interrupt request issued

xxMKn	Interrupt mask flag
0	Interrupt servicing enabled
1	Interrupt servicing disabled (pending)

xxPRn2	xxPRn1	xxPRn0	Interrupt priority specification bit
0	0	0	Specifies level 0 (highest).
0	0	1	Specifies level 1.
0	1	0	Specifies level 2.
0	1	1	Specifies level 3.
1	0	0	Specifies level 4.
1	0	1	Specifies level 5.
1	1	0	Specifies level 6.
1	1	1	Specifies level 7 (lowest).

Note The flag xxIFn is reset automatically by the hardware if an interrupt request signal is acknowledged.

Remark xx: Identification name of each peripheral unit (see **Table 19-2 Interrupt Control Register (xxICn)**)
n: Peripheral unit number (see **Table 19-2 Interrupt Control Register (xxICn)**).

The addresses and bits of the interrupt control registers are as follows.

Table 19-2. Interrupt Control Register (xxICn) (1/2)

	Address	Register	Bit							
			<7>	<6>	5	4	3	2	1	0
	FFFFF112H	PIC0	PIF0	PMK0	0	0	0	PPR02	PPR01	PPR00
	FFFFF114H	PIC1	PIF1	PMK1	0	0	0	PPR12	PPR11	PPR10
	FFFFF116H	PIC2	PIF2	PMK2	0	0	0	PPR22	PPR21	PPR20
	FFFFF118H	PIC3	PIF3	PMK3	0	0	0	PPR32	PPR31	PPR30
	FFFFF11AH	PIC4	PIF4	PMK4	0	0	0	PPR42	PPR41	PPR40
	FFFFF11CH	PIC5	PIF5	PMK5	0	0	0	PPR52	PPR51	PPR50
	FFFFF11EH	PIC6	PIF6	PMK6	0	0	0	PPR62	PPR61	PPR60
	FFFFF120H	PIC7	PIF7	PMK7	0	0	0	PPR72	PPR71	PPR70
	FFFFF12CH	TP0OVIC	TP0OVIF	TP0OVMK	0	0	0	TP0OVPR2	TP0OVPR1	TP0OVPR0
	FFFFF12EH	TP0CCIC0	TP0CCIF0	TP0CCMK0	0	0	0	TP0CCPR02	TP0CCPR01	TP0CCPR00
	FFFFF130H	TP0CCIC1	TP0CCIF1	TP0CCMK1	0	0	0	TP0CCPR12	TP0CCPR11	TP0CCPR10
	FFFFF132H	TP1OVIC	TP1OVIF	TP1OVMK	0	0	0	TP1OVPR2	TP1OVPR1	TP1OVPR0
	FFFFF134H	TP1CCIC0	TP1CCIF0	TP1CCMK0	0	0	0	TP1CCPR02	TP1CCPR01	TP1CCPR00
	FFFFF136H	TP1CCIC1	TP1CCIF1	TP1CCMK1	0	0	0	TP1CCPR12	TP1CCPR11	TP1CCPR10
	FFFFF138H	TP2OVIC	TP2OVIF	TP2OVMK	0	0	0	TP2OVPR2	TP2OVPR1	TP2OVPR0
	FFFFF13AH	TP2CCIC0	TP2CCIF0	TP2CCMK0	0	0	0	TP2CCPR02	TP2CCPR01	TP2CCPR00
	FFFFF13CH	TP2CCIC1	TP2CCIF1	TP2CCMK1	0	0	0	TP2CCPR12	TP2CCPR11	TP2CCPR10
	FFFFF13EH	TP3OVIC	TP3OVIF	TP3OVMK	0	0	0	TP3OVPR2	TP3OVPR1	TP3OVPR0
	FFFFF140H	TP3CCIC0	TP3CCIF0	TP3CCMK0	0	0	0	TP3CCPR02	TP3CCPR01	TP3CCPR00
	FFFFF142H	TP3CCIC1	TP3CCIF1	TP3CCMK1	0	0	0	TP3CCPR12	TP3CCPR11	TP3CCPR10
	FFFFF144H	TP4OVIC	TP4OVIF	TP4OVMK	0	0	0	TP4OVPR2	TP4OVPR1	TP4OVPR0
	FFFFF146H	TP4CCIC0	TP4CCIF0	TP4CCMK0	0	0	0	TP4CCPR02	TP4CCPR01	TP4CCPR00
	FFFFF148H	TP4CCIC1	TP4CCIF1	TP4CCMK1	0	0	0	TP4CCPR12	TP4CCPR11	TP4CCPR10
<R>	FFFFF14AH	TP5OVIC[Note]	TP5OVIF	TP5OVMK	0	0	0	TP5OVPR2	TP5OVPR1	TP5OVPR0
<R>	FFFFF14CH	TP5CCIC0[Note]	TP5CCIF0	TP5CCMK0	0	0	0	TP5CCPR02	TP5CCPR01	TP5CCPR00
<R>	FFFFF14EH	TP5CCIC1[Note]	TP5CCIF1	TP5CCMK1	0	0	0	TP5CCPR12	TP5CCPR11	TP5CCPR10
	FFFFF150H	TM0EQIC0	TM0EQIF0	TM0EQMK0	0	0	0	TM0EQPR02	TM0EQPR01	TM0EQPR00
	FFFFF152H	CB0RIC/ IICIC1	CB0RIF/ IICIF1	CB0RMK/ IICMK1	0	0	0	CB0RPR2/ IICPR12	CB0RPR1/ IICPR11	CB0RPR0/ IICPR10
	FFFFF154H	CB0TIC	CB0TIF	CB0TMK	0	0	0	CB0TPR2	CB0TPR1	CB0TPR0
	FFFFF156H	CB1RIC	CB1RIF	CB1RMK	0	0	0	CB1RPR2	CB1RPR1	CB1RPR0
	FFFFF158H	CB1TIC	CB1TIF	CB1TMK	0	0	0	CB1TPR2	CB1TPR1	CB1TPR0
	FFFFF15EH	CB3RIC	CB3RIF	CB3RMK	0	0	0	CB3RPR2	CB3RPR1	CB3RPR0
	FFFFF160H	CB3TIC	CB3TIF	CB3TMK	0	0	0	CB3TPR2	CB3TPR1	CB3TPR0
	FFFFF162H	UA0RIC	UA0RIF	UA0RMK	0	0	0	UA0RPR2	UA0RPR1	UA0RPR0
	FFFFF164H	UA0TIC	UA0TIF	UA0TMK	0	0	0	UA0TPR2	UA0TPR1	UA0TPR0
	FFFFF166H	UA1RIC	UA1RIF	UA1RMK	0	0	0	UA1RPR2	UA1RPR1	UA1RPR0
	FFFFF168H	UA1TIC	UA1TIF	UA1TMK	0	0	0	UA1TPR2	UA1TPR1	UA1TPR0
	FFFFF16AH	IICIC0	IICIF0	IICMK0	0	0	0	IICPR02	IICPR01	IICPR00
	FFFFF16EH	ADIC	ADIF	ADMK	0	0	0	ADPR2	ADPR1	ADPR0
<R>	FFFFF170H	DMAIC0[Note]	DMAIF0	DMAMK0	0	0	0	DMAPR02	DMAPR01	DMAPR00
<R>	FFFFF172H	DMAIC1[Note]	DMAIF1	DMAMK1	0	0	0	DMAPR12	DMAPR11	DMAPR10
<R>	FFFFF174H	DMAIC2[Note]	DMAIF2	DMAMK2	0	0	0	DMAPR22	DMAPR21	DMAPR20
<R>	FFFFF176H	DMAIC3[Note]	DMAIF3	DMAMK3	0	0	0	DMAPR32	DMAPR31	DMAPR30

<R> **Note** μPD703253Y only

Table 19-2. Interrupt Control Register (xxICn) (2/2)

Address	Register	Bit							
		<7>	<6>	5	4	3	2	1	0
FFFFF178H	KRIC	KRIF	KRMK	0	0	0	KRPR2	KRPR1	KRPR0
FFFFF17AH	WTIIC	WTIIF	WTIMK	0	0	0	WTIPR2	WTIPR1	WTIPR0
FFFFF17CH	WTIC	WTIF	WTMK	0	0	0	WTPR2	WTPR1	WTPR0
<R> FFFFF17EH	ERRIC0[Note 1]/ ERRIC[Note 2]	ERRIF0/ ERRIF	ERRMK0/ ERRMK	0	0	0	ERRPR02/ ERRPR2	ERRPR01/ ERRPR1	ERRPR00/ ERRPR0
<R> FFFFF180H	WUPIC0[Note 1]/ STAIC[Note 2]	WUPIF0/ STAIF	WUPMK0/ STAMK	0	0	0	WUPPR02/ STAPR2	WUPPR01/ STAPR1	WUPPR00/ STAPR0
<R> FFFFF182H	RECIC0[Note 1]/ IEIC1[Note 2]	RECIF0/ IEIF1	RECMK0/ IEMK1	0	0	0	RECPR02/ IEPR12	RECPR01/ IEPR11	RECPR00/ IEPR10
<R> FFFFF184H	TRXIC0[Note 1]/ IEIC2[Note 2]	TRXIF0/ IEIF2	TRXMK0/ IEMK2	0	0	0	TRXPR02/ IEPR22	TRXPR01/ IEPR21	TRXPR00/ IEPR20

<R> **Notes 1.** μPD703253Y only

2. μPD703252Y only

19.3.5 Interrupt mask registers 0 to 3 (IMR0 to IMR3)

The IMR0 to IMR3 registers set the interrupt mask state for the maskable interrupts. The xxMKn bit of the IMR0 to IMR3 registers is equivalent to the xxICn.xxMKn bit.

The IMRm register can be read or written in 16-bit units (m = 0 to 3).

If the higher 8 bits of the IMRm register are used as an IMRmH register and the lower 8 bits as an IMRmL register, these registers can be read or written in 8-bit or 1-bit units (m = 0 to 3).

Reset input sets these registers to FFFFH.

Caution The device file defines the xxICn.xxMKn bit as a reserved word. If a bit is manipulated using the name of xxMKn, the contents of the xxICn register, instead of the IMRm register, are rewritten (as a result, the contents of the IMRm register are also rewritten).

After reset: FFFFH R/W Address: IMR3 FFFFF106H, IMR3L FFFFF106H, IMR3H FFFFF107H

<R> IMR3 (IMR3H[Note 1])

15	14	13	12	11	10	9	8
1	1	1	1	1	TRXMK0[Note 2]/IEMK2[Note 3]	RECMK0[Note 2]/IEMK1[Note 3]	WUPMK0[Note 2]/STAMK[Note 3]

<R> IMR3L

7	6	5	4	3	2	1	0
ERRMK0[Note 2]/ERRMK[Note 3]	WTMK	WTIMK	KRMK	DMAMK3[Note 4]	DMAMK2[Note 4]	DMAMK1[Note 4]	DMAMK0[Note 4]

After reset: FFFFH R/W Address: IMR2 FFFFF104H, IMR2L FFFFF104H, IMR2H FFFFF105H

IMR2 (IMR2H[Note 1])

15	14	13	12	11	10	9	8
ADMK	1	IICMK0	UA1TMK	UA1RMK	UA0TMK	UA0RMK	CB3TMK

IMR2L

7	6	5	4	3	2	1	0
CB3RMK	1	1	CB1TMK	CB1RMK	CB0TMK	CB0RMK/IICMK1	TM0EQMK0

After reset: FFFFH R/W Address: IMR1 FFFFF102H, IMR1L FFFFF102H, IMR1H FFFFF103H

<R> IMR1 (IMR1H[Note 1])

15	14	13	12	11	10	9	8
TP5CCMK1[Note 4]	TP5CCMK0[Note 4]	TP5OVMK[Note 4]	TP4CCMK1	TP4CCMK0	TP4OVMK	TP3CCMK1	TP3CCMK0

IMR1L

7	6	5	4	3	2	1	0
TP3OVMK	TP2CCMK1	TP2CCMK0	TP2OVMK	TP1CCMK1	TP1CCMK0	TP1OVMK	TP0CCMK1

After reset: FFFFH R/W Address: IMR0 FFFFF100H, IMR0L FFFFF100H, IMR0H FFFFF101H

IMR0 (IMR0H[Note 1])

15	14	13	12	11	10	9	8
TP0CCMK0	TP0OVMK	1	1	1	1	1	PMK7

IMR0L

7	6	5	4	3	2	1	0
PMK6	PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	1

xxMKn	Setting of interrupt mask flag
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Notes 1. To read bits 8 to 15 of the IMR0 to IMR3 registers in 8-bit or 1-bit units, specify them as bits 0 to 7 of IMR0H to IMR3H registers.

<R> **2.** These bits are valid only in the μPD703253Y.

3. These bits are valid only in the μPD703252Y.

<R> **4.** These bits are valid only in the μPD703253Y.
Be sure to set these bits to 1 in the μPD703249Y and μPD703252Y.

Caution Set bits 0 and 9 to 13 of the IMR0 register, bits 5, 6, and 14 of the IMR2 register, and bits 11 to 15 of the IMR3 register to "1". If the setting of these bits is changed, the operation is not guaranteed.

Remark xx: Identification name of each peripheral unit (see **Table 19-2 Interrupt Control Register (xxICn)**).
n: Peripheral unit number (see **Table 19-2 Interrupt Control Register (xxICn)**)

19.3.6 In-service priority register (ISPR)

The ISPR register holds the priority level of the maskable interrupt currently acknowledged. When an interrupt request signal is acknowledged, the bit of this register corresponding to the priority level of that interrupt request signal is set to 1 and remains set while the interrupt is serviced.

When the RETI instruction is executed, the bit corresponding to the interrupt request signal having the highest priority is automatically cleared to 0 by hardware. However, it is not cleared to 0 when execution is returned from non-maskable interrupt servicing or exception processing.

This register is read-only, in 8-bit or 1-bit units.

Reset input clears this register to 00H.

Caution If an interrupt is acknowledged while the ISPR register is being read in the interrupt enabled (EI) status, the value of the ISPR register after the bits of the register have been set by acknowledging the interrupt may be read. To accurately read the value of the ISPR register before an interrupt is acknowledged, read the register while interrupts are disabled (DI).

After reset: 00H R Address: FFFFF1FAH

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
ISPR	ISPR7	ISPR6	ISPR5	ISPR4	ISPR3	ISPR2	ISPR1	ISPR0

ISPRn	Priority of interrupt currently acknowledged
0	Interrupt request signal with priority n not acknowledged
1	Interrupt request signal with priority n acknowledged

Remark n = 0 to 7 (priority level)

19.3.7 ID flag

This flag controls the maskable interrupt's operating state, and stores control information regarding enabling or disabling of interrupt request signals. An interrupt disable flag (ID) is assigned to the PSW.

Reset sets this flag to 00000020H.

After reset: 00000020H

	31	8	7	6	5	4	3	2	1	0
PSW	0		NP	EP	**ID**	SAT	CY	OV	S	Z

ID	Specification of maskable interrupt servicing[Note]
0	Maskable interrupt request signal acknowledgment enabled
1	Maskable interrupt request signal acknowledgment disabled (pending)

Note Interrupt disable flag (ID) function

This bit is set to 1 by the DI instruction and cleared to 0 by the EI instruction. Its value is also modified by the RETI instruction or LDSR instruction when referencing the PSW.

Non-maskable interrupt request signals and exceptions are acknowledged regardless of this flag. When a maskable interrupt request signal is acknowledged, the ID flag is automatically set to 1 by hardware.

The interrupt request signal generated during the acknowledgment disabled period (ID flag = 1) is acknowledged when the xxICn.xxIFn bit is set to 1, and the ID flag is cleared to 0.

19.3.8 Watchdog timer mode register 2 (WDTM2)

This register can be read or written in 8-bit units (for details, see **CHAPTER 10 FUNCTIONS OF WATCHDOG TIMER 2**).

Reset sets this register to 67H.

After reset: 67H R/W Address: FFFFF6D0H

	7	6	5	4	3	2	1	0
WDTM2	0	WDM21	WDM20	WDCS24	WDCS23	WDCS22	WDCS21	WDCS20

WDM21	WDM20	Selection of watchdog timer operation mode
0	0	Stops operation
0	1	Non-maskable interrupt request mode (INTWDT2 signal generated)
1	×	Reset mode (WDT2RES signal generated)

Remark For details on the WDCS24 to WDCS20 bits, see **Table 10-2 Watchdog Timer 2 Clock Selection**.

19.4 Software Exception

A software exception is generated when the CPU executes the TRAP instruction, and can always be acknowledged.

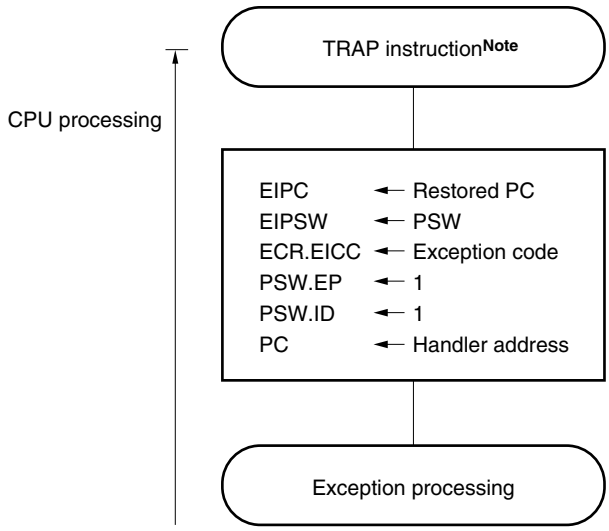
19.4.1 Operation

If a software exception occurs, the CPU performs the following processing, and transfers control to the handler routine.

<1> Saves the restored PC to EIPC.
<2> Saves the current PSW to EIPSW.
<3> Writes an exception code to the lower 16 bits (EICC) of ECR (interrupt source).
<4> Sets the PSW.EP and PSW.ID bits to 1.
<5> Sets the handler address (00000040H or 00000050H) corresponding to the software exception to the PC, and transfers control.

Figure 19-9 illustrates the processing of a software exception.

Figure 19-9. Software Exception Processing



Note TRAP instruction format: TRAP vector (the vector is a value from 00H to 1FH.)

The handler address is determined by the TRAP instruction's operand (vector). If the vector is 00H to 0FH, it becomes 00000040H, and if the vector is 10H to 1FH, it becomes 00000050H.

19.4.2 Restoration

Restoration from software exception processing is carried out by the RETI instruction.

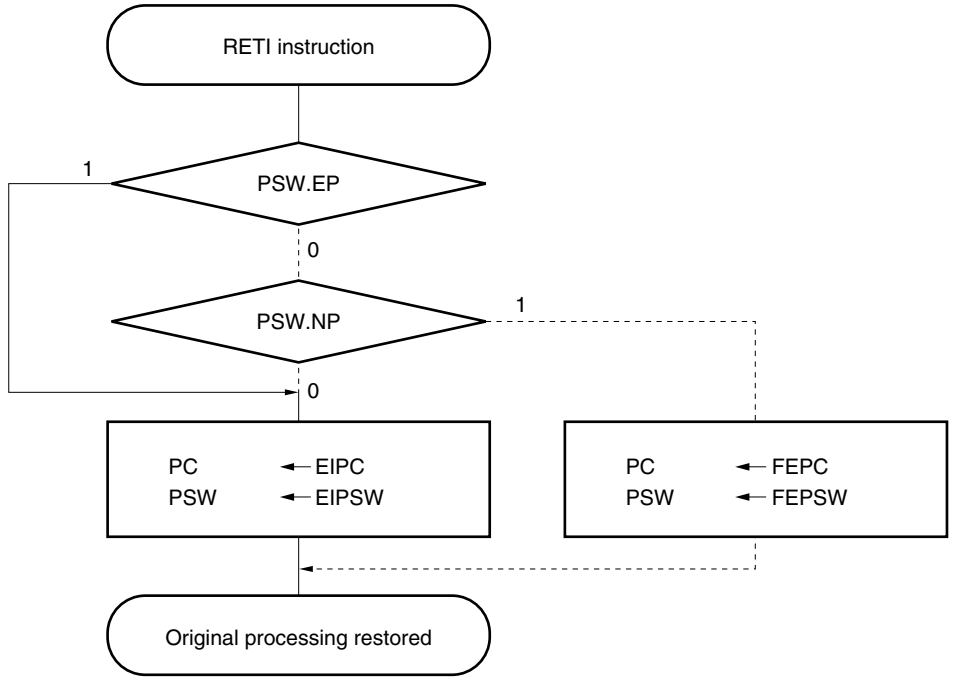
By executing the RETI instruction, the CPU carries out the following processing and shifts control to the restored PC's address.

<1> Loads the restored PC and PSW from EIPC and EIPSW because the PSW.EP bit is 1.

<2> Transfers control to the address of the restored PC and PSW.

Figure 19-10 illustrates the processing of the RETI instruction.

Figure 19-10. RETI Instruction Processing



Caution When the EP and NP bits are changed by the LDSR instruction during the software exception processing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 1 and the NP bit back to 0 using the LDSR instruction immediately before the RETI instruction.

Remark The solid line shows the CPU processing flow.

19.4.3 EP flag

The EP flag is a status flag used to indicate that exception processing is in progress. It is set when an exception occurs.

After reset: 00000020H

	31	8	7	6	5	4	3	2	1	0
PSW	0		NP	**EP**	ID	SAT	CY	OV	S	Z

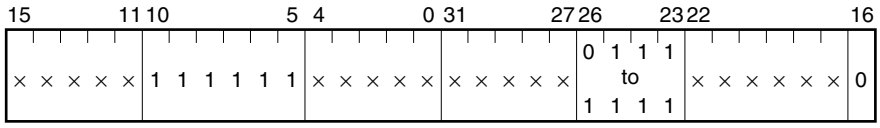
EP	Exception processing status
0	Exception processing not in progress.
1	Exception processing in progress.

19.5 Exception Trap

An exception trap is an interrupt that is requested when the illegal execution of an instruction takes place. In the V850ES/SG1, an illegal opcode exception (ILGOP: Illegal Opcode Trap) is considered as an exception trap.

19.5.1 Illegal opcode definition

The illegal instruction has an opcode (bits 10 to 5) of 111111B, a sub-opcode (bits 26 to 23) of 0111B to 1111B, and a sub-opcode (bit 16) of 0B. An exception trap is generated when an instruction applicable to this illegal instruction is executed.



×: Arbitrary

Caution Since it is possible to assign this instruction to an illegal opcode in the future, it is recommended that it not be used.

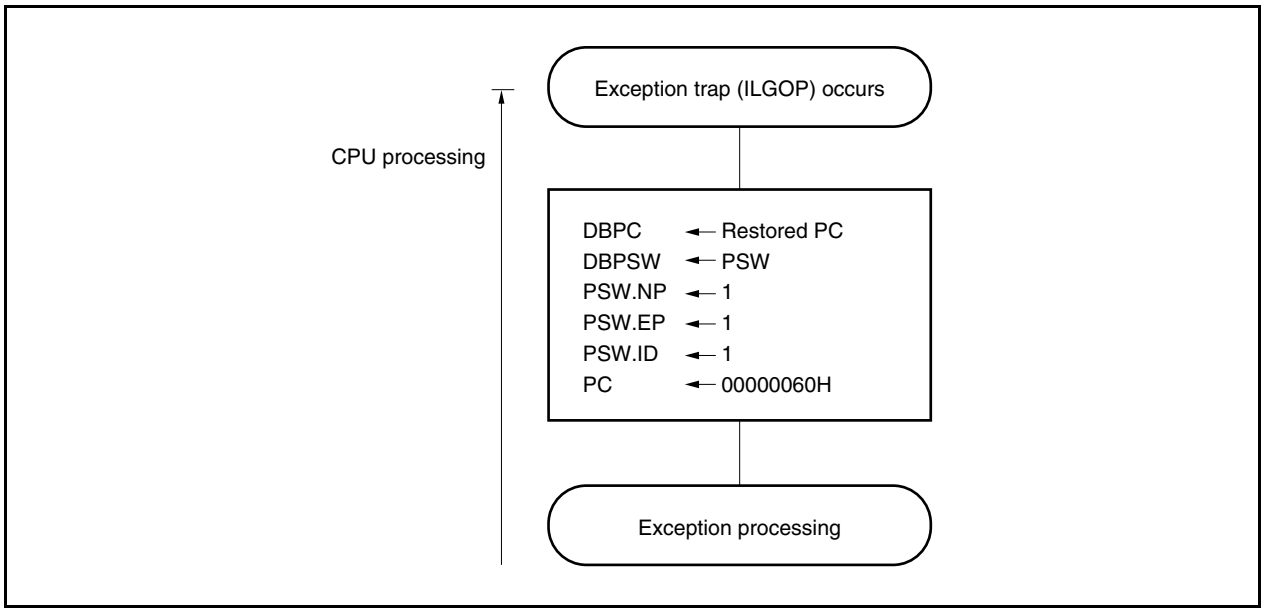
(1) Operation

If an exception trap occurs, the CPU performs the following processing, and transfers control to the handler routine.

<1> Saves the restored PC to DBPC.
<2> Saves the current PSW to DBPSW.
<3> Sets the PSW.NP, PSW.EP, and PSW.ID bits to 1.
<4> Sets the handler address (00000060H) corresponding to the exception trap to the PC, and transfers control.

Figure 19-11 illustrates the processing of the exception trap.

Figure 19-11. Exception Trap Processing



(2) Restoration

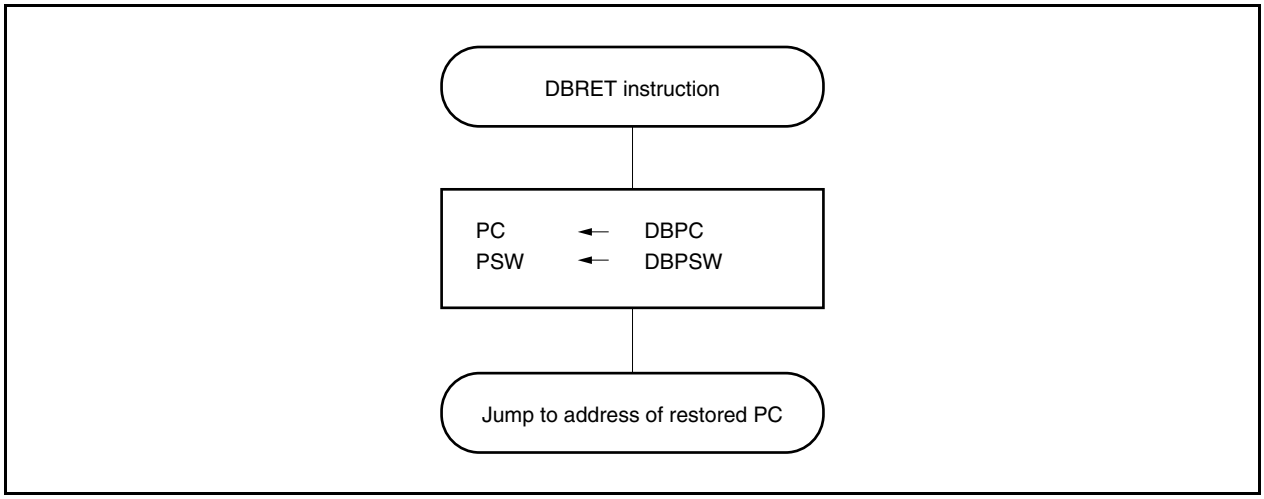
Restoration from an exception trap is carried out by the DBRET instruction. By executing the DBRET instruction, the CPU carries out the following processing and controls the address of the restored PC.

<1> Loads the restored PC and PSW from DBPC and DBPSW.
<2> Transfers control to the address indicated by the restored PC and PSW.

<R> **Caution DBPC and DBPSW can be accessed only during the interval between the execution of the illegal opcode and DBRET instruction.**

Figure 19-12 illustrates the processing for restoring from an exception trap.

Figure 19-12. Processing for Restoring from Exception Trap



19.5.2 Debug trap

A debug trap is an exception that is generated when the DBTRAP instruction is executed and is always acknowledged.

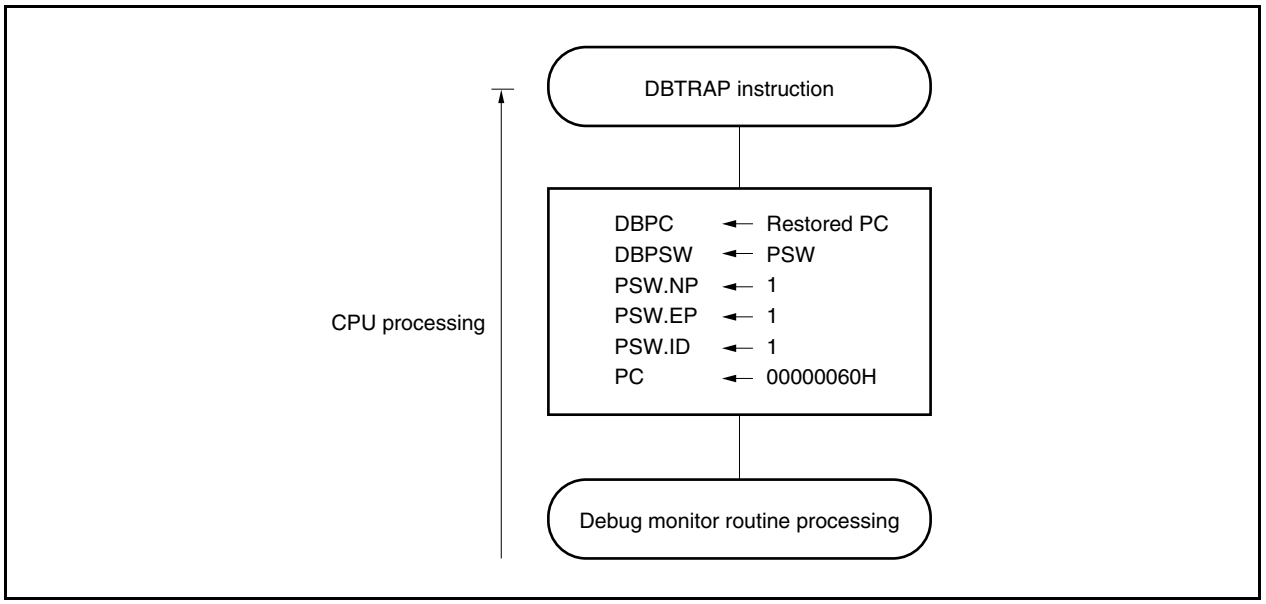
(1) Operation

Upon occurrence of a debug trap, the CPU performs the following processing.

<1> Saves restored PC to DBPC.
<2> Saves current PSW to DBPSW.
<3> Sets the PSW.NP, PSW.EP, and PSW.ID bits to 1.
<4> Sets handler address (00000060H) for debug trap to PC and transfers control.

Figure 19-13 shows the debug trap processing format.

Figure 19-13. Debug Trap Processing Format



(2) Restoration

Restoration from a debug trap is executed with the DBRET instruction.

With the DBRET instruction, the CPU performs the following steps and transfers control to the address of the restored PC.

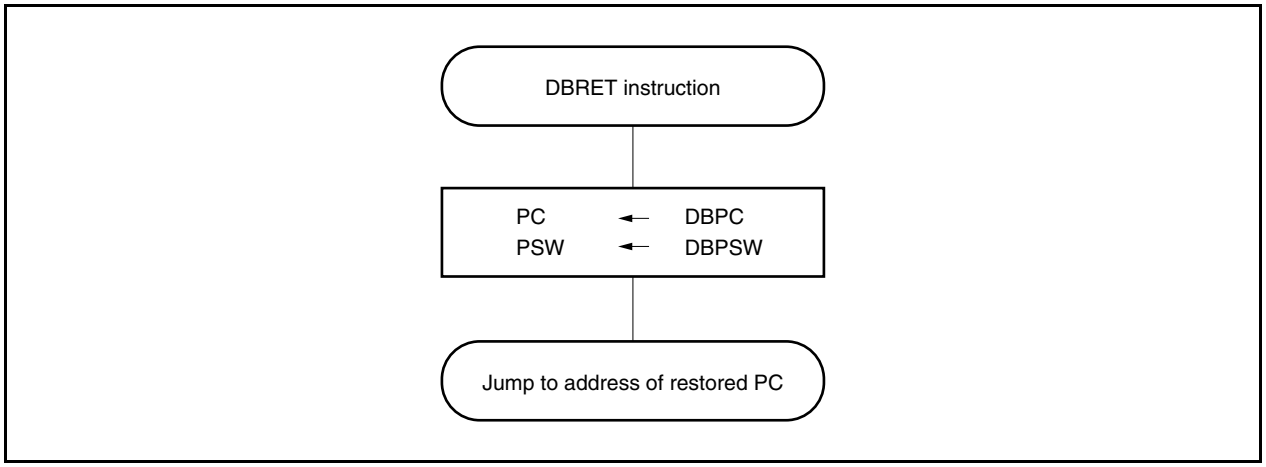
<1> The restored PC and PSW are read from DBPC and DBPSW.

<2> Control is transferred to the fetched address of the restored PC and PSW.

Caution DBPC and DBPSW can be accessed after the DBTRAP instruction is executed and before the DBRET instruction is executed.

Table 19-14 shows the processing format for restoration from a debug trap.

Figure 19-14. Processing Format of Restoration from Debug Trap



19.6 External Interrupt Request Input Pins (NMI and INTP0 to INTP7)

19.6.1 Noise elimination

(1) Eliminating noise on NMI pin

The NMI pin has an internal noise elimination circuit that uses analog delay. Therefore, the input level of the NMI pin is not detected as an edge unless it is maintained for a specific time or longer. Therefore, an edge is detected after specific time.

The NMI pin can be used to release the STOP mode. In the STOP mode, noise is not eliminated by using the system clock because the internal system clock is stopped.

(2) Eliminating noise on INTP0 to INTP7 pins

The INTP0 to INTP7 pins have an internal noise elimination circuit that uses analog delay. Therefore, the input level of the NMI pin is not detected as an edge unless it is maintained for a specific time or longer. Therefore, an edge is detected after specific time.

19.6.2 Edge detection

The valid edge of each of the NMI and INTP0 to INTP7 pins can be selected from the following four.

- Rising edge
- Falling edge
- Both rising and falling edges
- No edge detected

The edge of the NMI pin is not detected after reset. Therefore, the interrupt request signal is not acknowledged unless a valid edge is enabled by using the INTF0 and INTR0 register (the NMI pin functions as a normal port pin).

(1) External interrupt falling, rising edge specification register 0 (INTF0, INTR0)

The INTF0 and INTR0 registers are 8-bit registers that specify detection of the falling and rising edges of the NMI pin via bit 2 and the external interrupt pins (INTP0 to INTP3) via bits 3 to 6.

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF0n and INTR0n bits to 00, and then set the port mode.

After reset: 00H R/W Address: INTF0 FFFFFC00H, INTR0 FFFFFC20H

	7	6	5	4	3	2	1	0
INTF0	0	INTF06	INTF05	INTF04	INTF03	INTF02	0	0
		INTP3	INTP2	INTP1	INTP0	NMI		
	7	6	5	4	3	2	1	0
INTR0	0	INTR06	INTR05	INTR04	INTR03	INTR02	0	0
		INTP3	INTP2	INTP1	INTP0	NMI		

Remark For how to specify a valid edge, see **Table 19-3**.

Table 19-3. Valid Edge Specification

INTF0n	INTR0n	Valid Edge Specification (n = 2 to 6)
0	0	No edge detected
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Caution Be sure to clear the INTF0n and INTR0n bits to 00 when these registers are not used as the NMI or INTP0 to INTP3 pins.

Remark n = 2: Control of NMI pin

n = 3 to 6: Control of INTP0 to INTP3 pins

(2) External interrupt falling, rising edge specification register 3 (INTF3, INTR3)

The INTF3 and INTR3 registers are 8-bit registers that specify detection of the falling and rising edges of the external interrupt pin (INTP7).

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

Cautions 1. When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF31 and INTR31 bits to 00, and then set the port mode.

2. The INTP7 pin and RXDA0 pin are alternate-function pins. When using the pin as the RXDA0 pin, disable edge detection for the INTP7 alternate-function pin (clear the INTF3.INTF31 bit and the INRT3.INTR31 bit to 0). When using the pin as the INTP7 pin, stop UARTA0 reception (clear the UA0CTL0.UA0RXE bit to 0).

After reset: 00H R/W Address: INTF3 FFFFFC06H, INTR3 FFFFFC26H

	7	6	5	4	3	2	1	0
INTF3	0	0	0	0	0	0	INTF31	0
							INTP7	
	7	6	5	4	3	2	1	0
INTR3	0	0	0	0	0	0	INTR31	0
							INTP7	

Remark For how to specify a valid edge, see **Table 19-4**.

Table 19-4. Valid Edge Specification

INTF31	INTR31	Valid Edge Specification
0	0	No edge detected
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Caution Be sure to clear the INTF31 and INTR31 bits to 00 when these registers are not used as INTP7 pin.

(3) External interrupt falling, rising edge specification register 9H (INTF9H, INTR9H)

The INTF9H and INTR9H registers are 8-bit registers that specify detection of the falling and rising edges of the external interrupt pins (INTP4 to INTP6).

These registers can be read or written in 8-bit or 1-bit units.

Reset sets these registers to 00H.

Caution When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF9n and INTR9n bits to 0, and then set the port mode.

After reset: 00H R/W Address: INTF9H FFFFFC13H, INTR9H FFFFFC33H

	15	14	13	12	11	10	9	8
INTF9H	INTF915	INTF914	INTF913	0	0	0	0	0
	INTP6	INTP5	INTP4					
	15	14	13	12	11	10	9	8
INTR9H	INTR915	INTR914	INTR913	0	0	0	0	0
	INTP6	INTP5	INTP4					

Remark For how to specify a valid edge, see **Table 19-5**.

Table 19-5. Valid Edge Specification

INTF9n	INTR9n	Valid Edge Specification (n = 13 to 15)
0	0	No edge detected
0	1	Rising edge
1	0	Falling edge
1	1	Both rising and falling edges

Caution Be sure to clear the INTF9n and INTR9n bits to 00 when these registers are not used as INTP4 to INTP6 pins.

Remark n = 13 to 15: Control of INTP4 to INTP6 pins

(4) Noise elimination control register (NFC)

Digital noise elimination can be selected for the INTP3 pin. The noise elimination settings are performed using the NFC register.

When digital noise elimination is selected, the sampling clock for digital sampling can be selected from among $f_{XX}/64$, $f_{XX}/128$, $f_{XX}/256$, $f_{XX}/512$, $f_{XX}/1{,}024$, and f_{XT}. Sampling is performed three times.

<R> When digital noise elimination is selected, if the clock that performs sampling in the standby mode is stopped, then the INTP3 interrupt request signal cannot be used for releasing the standby mode. When f_{XT} is used as the sampling clock, the INTP3 interrupt request signal can be used for releasing either the subclock operating mode or the IDLE1/IDLE2/STOP/sub-IDLE mode.

This register can be read or written in 8-bit units.

Reset input clears this register to 00H.

Caution After the sampling clock has been changed, it takes 3 sampling clocks to initialize the digital noise eliminator. Therefore, if an INTP3 valid edge is input within these 3 sampling clocks after the sampling clock has been changed, an interrupt request signal may be generated. Therefore, be careful about the following points when using the interrupt and DMA functions <R> **(μPD703253Y only).**

- **When using the interrupt function, after the 3 sampling clocks have elapsed, enable interrupts after the interrupt request flag (PIC3.PIF3 bit) has been cleared.**
- <R> **When using the DMA function (started by INTP3) in the μPD703253Y, enable DMA after 3 sampling clocks have elapsed.**

After reset: 00H R/W Address: FFFFF318H

	7	6	5	4	3	2	1	0
NFC	NFEN	0	0	0	0	NFC2	NFC1	NFC0

NFEN	Settings of INTP3 pin noise elimination
0	Analog noise elimination (60 ns (TYP.))
1	Digital noise elimination

NFC2	NFC1	NFC0	Digital sampling clock
0	0	0	$f_{XX}/64$
0	0	1	$f_{XX}/128$
0	1	0	$f_{XX}/256$
0	1	1	$f_{XX}/512$
1	0	0	$f_{XX}/1{,}024$
1	0	1	f_{XT} (subclock)
Other than above			Setting prohibited

Remarks 1. Since sampling is performed three times, the reliably eliminated noise width is 2 sampling clocks.

2. In the case of noise with a width smaller than 2 sampling clocks, an interrupt request signal is generated if noise synchronized with the sampling clock is input.

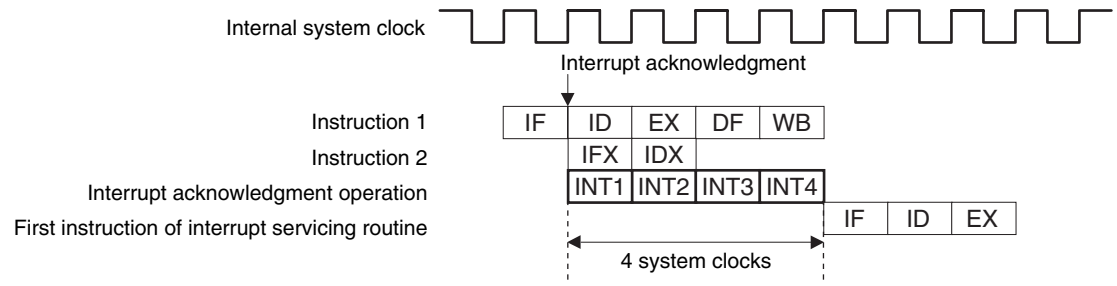
19.7 Interrupt Acknowledge Time of CPU

Except the following cases, the interrupt acknowledge time of the CPU is 4 clocks minimum. To input interrupt request signals successively, input the next interrupt request signal at least 5 clocks after the preceding interrupt.

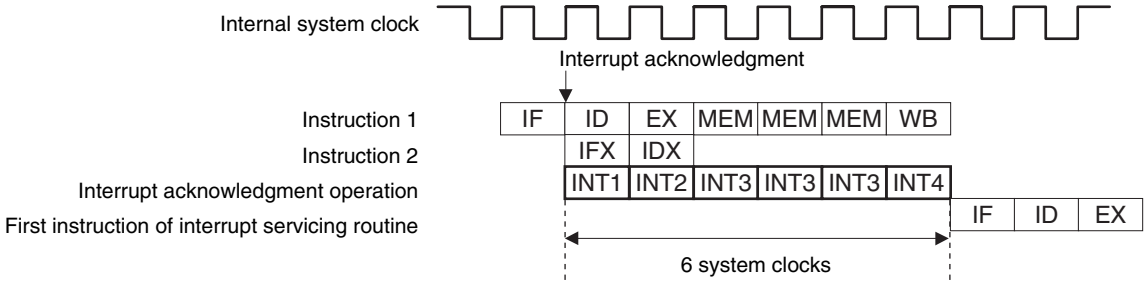
- In IDLE1/IDLE2/STOP mode
- When the external bus is accessed
- When interrupt request non-sampling instructions are successively executed (see **19.8 Periods in Which Interrupts Are Not Acknowledged by CPU**.)
- When the interrupt control register is accessed
- <R> When an on-chip peripheral I/O register is accessed
- <R> When a programmable peripheral I/O register is accessed (μPD703253Y only)

Figure 19-15. Pipeline Operation at Interrupt Request Signal Acknowledgment (Outline)

(1) Minimum interrupt response time



(2) Maximum interrupt response time



Remarks 1. INT1 to INT4: Interrupt acknowledgment processing
IFX: Invalid instruction fetch
IDX: Invalid instruction decode

<R> **2.** If the same interrupt request signal is generated while an interrupt of four cycles is being acknowledged, the new interrupt request signal is discarded. The next interrupt request signal from the same source is registered four cycles later.

Interrupt response time (internal system clock)			Condition
	Internal interrupt	External interrupt	
Minimum	4	4 + Analog delay time	The following cases are exceptions. • In IDLE1/IDLE2/STOP mode • External bus access • Two or more interrupt request non-sample instructions are executed in succession • Access to interrupt control register • Access to on-chip peripheral I/O register • Access to programmable peripheral I/O register (μPD703253Y only)
Maximum	6	6 + Analog delay time	

<R>

19.8 Periods in Which Interrupts Are Not Acknowledged by CPU

An interrupt is acknowledged by the CPU while an instruction is being executed. However, no interrupt will be acknowledged between an interrupt request non-sample instruction and the next instruction (interrupt is held pending).

The interrupt request non-sample instructions are as follows.

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (for PSW)
- The store instruction for the PRCMD register
- The store, SET1, NOT1, or CLR1 instructions for the following registers.
  - Interrupt-related registers:
    Interrupt control register (xxICn), interrupt mask registers 0 to 3 (IMR0 to IMR3)
  - Power save control register (PSC)
  - On-chip debug mode register (OCDM)

Remark xx: Identification name of each peripheral unit (see **Table 19-2 Interrupt Control Register (xxICn)**)
n: Peripheral unit number (see **Table 19-2 Interrupt Control Register (xxICn)**).

19.9 Cautions

(1) NMI pin

The NMI pin and P02 pin share a pin, and function as a normal port pin after being reset. To enable the NMI pin, validate the NMI pin with the PMC0 register. The initial setting of the NMI pin is “No edge detected”. Select the NMI pin valid edge using the INTF0 and INTR0 registers.

<R> (2) Interrupt control register (xxICn)

When manipulating the xxICn.xxMKn bit while interrupt requests may occur (including the state in which interrupts are disabled (DI)), be sure to use a bit manipulation instruction or use the IMRm.xxMKn bit (m = 0 to 3).

<R> (3) In-service priority register (ISPR)

If an interrupt is acknowledged while the ISPR register is being read in the interrupt enabled (EI) status, the value of the ISPR register after the bits of the register have been set by acknowledging the interrupt may be read. To accurately read the value of the ISPR register before an interrupt is acknowledged, read the register while interrupts are disabled (DI).

CHAPTER 20 KEY INTERRUPT FUNCTION

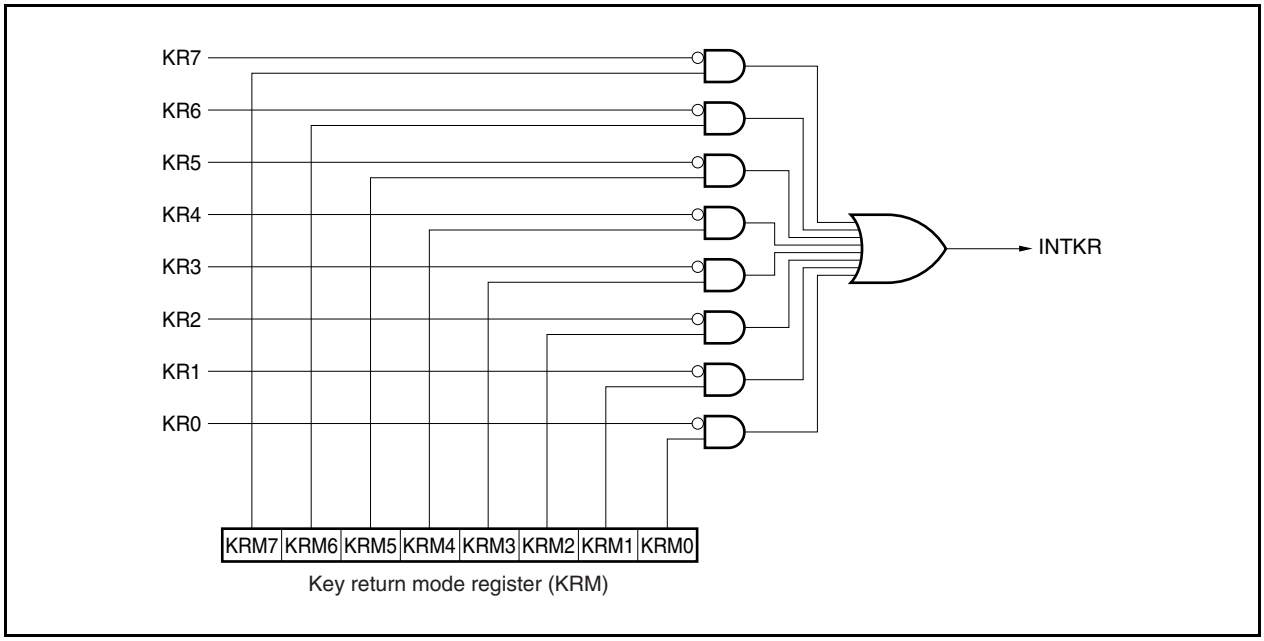
20.1 Function

A key interrupt request signal (INTKR) can be generated by inputting a falling edge to the eight key input pins (KR0 to KR7) by setting the KRM register.

Table 20-1. Assignment of Key Return Detection Pins

Flag	Pin Description
KRM0	Controls KR0 signal in 1-bit units
KRM1	Controls KR1 signal in 1-bit units
KRM2	Controls KR2 signal in 1-bit units
KRM3	Controls KR3 signal in 1-bit units
KRM4	Controls KR4 signal in 1-bit units
KRM5	Controls KR5 signal in 1-bit units
KRM6	Controls KR6 signal in 1-bit units
KRM7	Controls KR7 signal in 1-bit units

Figure 20-1. Key Return Block Diagram



20.2 Register

(1) Key return mode register (KRM)

The KRM register controls the KRM0 to KRM7 bits using the KR0 to KR7 signals.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF300H

	7	6	5	4	3	2	1	0
KRM	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	Control of key return mode
0	Does not detect key return signal
1	Detects key return signal

Caution Rewrite the KRM register after once clearing the KRM register to 00H.

Remark For the alternate-function pin settings, see **Table 4-15 Using Port Pin as Alternate Function Pin**.

20.3 Cautions

(1) If a low level is input to any of the KR0 to KR7 pins, the INTKR signal is not generated even if the falling edge of another pin is input.

(2) The RXDA1 and KR7 pins must not be used at the same time. To use the RXDA1 pin, do not use the KR7 pin. To use the KR7 pin, do not use the RXDA1 pin (it is recommended to set the PFC91 bit to 1 and clear PFCE91 bit to 0).

(3) If the KRM register is changed, an interrupt request signal (INTKR) may be generated. To prevent this, change the KRM register after disabling interrupts (DI) or masking, then clear the interrupt request flag (KRIC.KRIF bit) to 0, and enable interrupts (EI) or clear the mask.

(4) To use the key interrupt function, be sure to set the port pin to the key return pin and then enable the operation with the KRM register. To switch from the key return pin to the port pin, disable the operation with the KRM register and then set the port pin.

CHAPTER 21 STANDBY FUNCTION

21.1 Overview

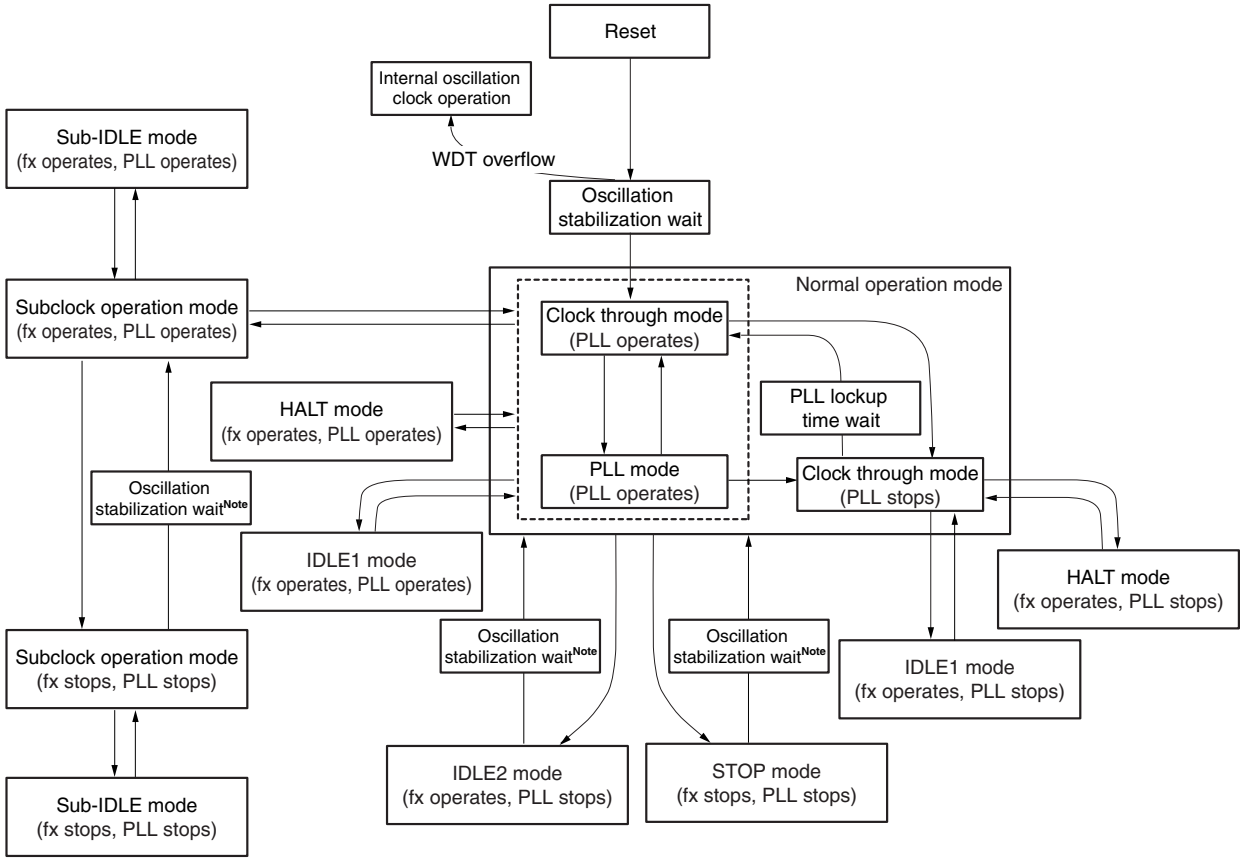
The power consumption of the system can be effectively reduced by using the standby modes in combination and selecting the appropriate mode for the application. The available standby modes are listed in Table 21-1.

Table 21-1. Standby Modes

Mode	Functional Outline
HALT mode	Mode in which only the operating clock of the CPU is stopped
IDLE1 mode	Mode in which all the operations of the internal circuits except the oscillator and PLL[Note] are stopped
IDLE2 mode	Mode in which all the operations of the internal circuits except the oscillator are stopped
STOP mode	Mode in which all the operations of the internal circuits except the subclock oscillator are stopped
Subclock operation mode	Mode in which the subclock is used as the internal system clock
Sub-IDLE mode	Mode in which all the operations of the internal circuits except the oscillator are stopped, in the subclock operation mode

Note The PLL holds the previous operating status.

Figure 21-1. Status Transition



Note If a WDT overflow occurs during an oscillation stabilization time, the CPU operates on the internal oscillation clock.

Remark fx: Main clock oscillation frequency

21.2 Registers

(1) Power save control register (PSC)

The PSC register is an 8-bit register that controls the standby function. The STP bit of this register is used to specify the standby mode. This register is a special register that can be written only by the special sequence combinations (see **3.4.8 Special registers**).

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF1FEH

	7	<6>	<5>	<4>	3	2	<1>	0
PSC	0	NMI1M	NMI0M	INTM	0	0	STP	0

NMI1M	Standby mode release control upon occurrence of INTWDT2 signal
0	Standby mode release by INTWDT2 signal enabled
1	Standby mode release by INTWDT2 signal disabled

NMI0M	Standby mode release control by NMI pin input
0	Standby mode release by NMI pin input enabled
1	Standby mode release by NMI pin input disabled

INTM	Standby mode release control via maskable interrupt request signal
0	Standby mode release by maskable interrupt request signal enabled
1	Standby mode release by maskable interrupt request signal disabled

STP	Standby mode[Note] setting
0	Normal mode
1	Standby mode

Note Standby mode set by STP bit: IDLE1, IDLE2, STOP, or sub-IDLE mode

Cautions 1. Before setting the IDLE1, IDLE2, STOP, or sub-IDLE mode, set the PSMR.PSM1 and PSMR.PSM0 bits and then set the STP bit.

2. Settings of the NMI1M, NMI0M, and INTM bits are invalid when HALT mode is released.

<R> **3. If the NMI1M, NMI0M, or INTM bit is set to 1 at the same time the STP bit is set to 1, the setting of NMI1M, NMI0M, or INTM bit becomes invalid. If there is an unmasked interrupt request signal being held pending when the IDLE1/IDLE2/STOP mode is set, set the bit corresponding to the interrupt request signal (NMI1M, NMI0M, or INTM) to 1, and then set the STP bit to 1.**

(2) Power save mode register (PSMR)

The PSMR register is an 8-bit register that controls the operation in the power save mode.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF820H

	7	6	5	4	3	2	<1>	<0>
PSMR	0	0	0	0	0	0	PSM1	PSM0

PSM1	PSM0	Specification of operation in software standby mode
0	0	IDLE1, sub-IDLE modes
0	1	STOP mode
1	0	IDLE2, sub-IDLE modes
1	1	STOP mode

Cautions 1. Be sure to clear bits 2 to 7 to "0".

2. The PSM0 and PSM1 bits are valid only when the PSC.STP bit is 1.

Remark IDLE1: In this mode, all operations except the oscillator operation and some other circuits (PLL) are stopped.

After the IDLE1 mode is released, the normal operation mode is restored without needing to secure the oscillation stabilization time, like the HALT mode.

IDLE2: In this mode, all operations except the oscillator operation are stopped.

After the IDLE2 mode is released, the normal operation mode is restored following the lapse of the setup time specified by the OSTS register (PLL).

STOP: In this mode, all operations except the subclock oscillator operation are stopped.

After the STOP mode is released, the normal operation mode is restored following the lapse of the oscillation stabilization time specified by the OSTS register.

Sub-IDLE: In this mode, all other operations are halted except for the oscillator. After the IDLE mode has been released by the interrupt request signal, the subclock operation mode will be restored after 12 cycles of the subclock have been secured.

(3) Oscillation stabilization time select register (OSTS)

The wait time until the oscillation stabilizes after the STOP mode is released or the wait time until the PLL stabilizes after the IDLE2 mode is released is controlled by the OSTS register.

The OSTS register can be read or written 8-bit units.

Reset sets this register to 06H.

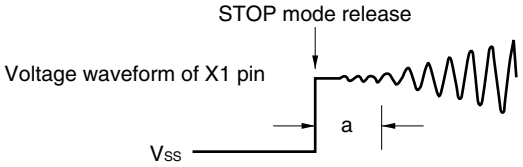
After reset: 06H R/W Address: FFFFF6C0H

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

OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time/setup time[Note]	fx	
				4 MHz	5 MHz
0	0	0	$2^{10}/f_x$	0.256 ms	0.205 ms
0	0	1	$2^{11}/f_x$	0.512 ms	0.410 ms
0	1	0	$2^{12}/f_x$	1.024 ms	0.819 ms
0	1	1	$2^{13}/f_x$	2.048 ms	1.638 ms
1	0	0	$2^{14}/f_x$	4.096 ms	3.277 ms
1	0	1	$2^{15}/f_x$	8.192 ms	6.554 ms
1	1	0	$2^{16}/f_x$	16.38 ms	13.107 ms
1	1	1	Setting prohibited		

Note The oscillation stabilization time and setup time are required when the STOP mode and IDLE2 mode are released, respectively.

Cautions 1. The wait time following release of the STOP mode does not include the time until the clock oscillation starts (“a” in the figure below) following release of the STOP mode, regardless of whether the STOP mode is released by reset input or the occurrence of an interrupt request signal.



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

3. The oscillation stabilization time following reset release is $2^{16}/f_x$ (because the initial value of the OSTS register = 06H).

Remark fx: Main clock oscillation frequency

21.3 HALT Mode

21.3.1 Setting and operation status

The HALT mode is set when a dedicated instruction (HALT) is executed in the normal operation mode.

In the HALT mode, the clock oscillator continues operating. Only clock supply to the CPU is stopped; clock supply to the other on-chip peripheral functions continues.

As a result, program execution is stopped, and the internal RAM retains the contents before the HALT mode was set. The on-chip peripheral functions that are independent of instruction processing by the CPU continue operating.

Table 21-3 shows the operating status in the HALT mode.

The average current consumption of the system can be reduced by using the HALT mode in combination with the normal operation mode for intermittent operation.

Cautions 1. Insert five or more NOP instructions after the HALT instruction.

2. If the HALT instruction is executed while an unmasked interrupt request signal is being held pending, the status shifts to HALT mode, but the HALT mode is then released immediately by the pending interrupt request.

21.3.2 Releasing HALT mode

The HALT mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from a peripheral function operable in the HALT mode, or reset signal ($\overline{\text{RESET}}$ pin input, reset signal (WDT2RES) generation by overflow of watchdog timer, or reset signal (CLMRES) generation by clock monitor (CLM)).

After the HALT mode has been released, the normal operation mode is restored.

(1) Releasing HALT mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The HALT mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal. If the HALT mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the HALT mode is released, but that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the HALT mode is released and that interrupt request signal is acknowledged.

Table 21-2. Operation After Releasing HALT Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed.	The next instruction is executed.

(2) Releasing HALT mode by reset

The same operation as the normal reset operation is performed.

Table 21-3. Operating Status in HALT Mode

Item \ Setting of HALT Mode		Operating Status	
		When Subclock Is Not Used	When Subclock Is Used
Main clock oscillator		Oscillation enabled	
Subclock oscillator		–	Oscillation enabled
Internal oscillator		Oscillation enabled	
PLL		Operable	
CPU		Stops operation	
<R> DMA[Note 1]		Operable	
Interrupt controller		Operable	
ROM correction		Stops operation	
<R> Timer P (TMP0 to TMP5[Note 2])		Operable	
Timer M (TMM0)		Operable when a clock other than f_{XT} is selected as the count clock	Operable
Watch timer		Operable when f_X (divided BRG) is selected as the count clock	Operable
Watchdog timer 2		Operable when a clock other than f_{XT} is selected as the count clock	Operable
Serial interface	CSIB0, CSIB1, CSIB3	Operable	
	I^2C00, I^2C01	Operable	
	UARTA0, UARTA1	Operable	
<R> IEBus controller[Note 3]		Operable	
<R> CAN controller[Note 1]		Operable	
A/D converter		Operable	
<R> D/A converter[Note 1]		Operable	
Key interrupt function (KR)		Operable	
External bus interface		See **2.2 Pin States**.	
Port function		Retains status before HALT mode was set	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the HALT mode was set.	

<R> **Notes 1.** μPD703253Y only

<R> **2.** μPD703249Y, 703252Y: TMP0 to TMP4
μPD703253Y: TMP0 to TMP5

<R> **3.** μPD703252Y only

21.4 IDLE1 Mode

21.4.1 Setting and operation status

The IDLE1 mode is set by clearing the PSMR.PSM1 and PSMR.PSM0 bits to 00 and setting the PSC.STP bit to 1 in the normal operation mode.

In the IDLE1 mode, the clock oscillator, and PLL continue operating but clock supply to the CPU and other on-chip peripheral functions stops.

As a result, program execution stops and the contents of the internal RAM before the IDLE1 mode was set are retained. The CPU and other on-chip peripheral functions stop operating. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Table 21-5 shows the operating status in the IDLE1 mode.

The IDLE1 mode can reduce the power consumption more than the HALT mode because it stops the operation of the on-chip peripheral functions. The main clock oscillator does not stop, so the normal operation mode can be restored without waiting for the oscillation stabilization time after the IDLE1 mode has been released, in the same manner as when the HALT mode is released.

Cautions 1. Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE1 mode.

2. If the IDLE1 mode is set while an unmasked interrupt request signal is being held pending, the IDLE1 mode is released immediately by the pending interrupt request.

21.4.2 Releasing IDLE1 mode

The IDLE1 mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from a peripheral function operable in the IDLE1 mode, or reset signal ($\overline{\text{RESET}}$ pin input, reset signal (WDT2RES) generation by overflow of watchdog timer, or reset signal (CLMRES) generation by clock monitor (CLM)).

After the IDLE1 mode has been released, the normal operation mode is restored.

(1) Releasing IDLE1 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The IDLE1 mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal. If the IDLE1 mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is processed as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the IDLE1 mode is released, but that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the IDLE1 mode is released and that interrupt request signal is acknowledged.

Caution An interrupt request signal that is disabled by setting the PSC.NMI1M, PSC.NMI0M, and PSC.INTM bits to 1 becomes invalid and IDLE1 mode is not released.

Table 21-4. Operation After Releasing IDLE1 Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed.	The next instruction is executed.

(2) Releasing IDLE1 mode by reset

The same operation as the normal reset operation is performed.

Table 21-5. Operating Status in IDLE1 Mode

	Setting of IDLE1 Mode / Item		Operating Status	
			When Subclock Is Not Used	When Subclock Is Used
	Main clock oscillator		Oscillation enabled	
	Subclock oscillator		–	Oscillation enabled
	Internal oscillator		Oscillation enabled	
	PLL		Operable	
	CPU		Stops operation	
<R>	DMA[Note 1]		Stops operation	
	Interrupt controller		Stops operation (but standby mode release enabled)	
	ROM correction		Stops operation	
<R>	Timer P (TMP0 to TMP5[Note 2])		Stops operation	
	Timer M (TMM0)		Operable when f_R/8 is selected as the count clock	Operable when f_R/8 or f_{XT} is selected as the count clock
	Watch timer		Operable when f_X (divided BRG) is selected as the count clock	Operable
	Watchdog timer 2		Operable when f_R is selected as the count clock	Operable when f_R or f_{XT} is selected as the count clock
	Serial interface	CSIB0, CSIB1, CSIB3	Operable when the $\overline{\text{SCKBn}}$ input clock is selected as the count clock (n = 0, 1, 3)	
		I^2C00, I^2C01	Stops operation	
		UARTA0, UARTA1	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)	
<R>	IEBus controller[Note 3]		Stops operation	
<R>	CAN controller[Note 1]		Stops operation	
	A/D converter		Holds operation (conversion result held)[Note 4]	
<R>	D/A converter[Note 1]		Holds operation (output held[Note 4])	
	Key interrupt function (KR)		Operable	
	External bus interface		See **2.2 Pin States**.	
	Port function		Retains status before IDLE1 mode was set	
	Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the IDLE1 mode was set.	

<R> **Notes 1.** μPD703253Y only

<R> **2.** μPD703249Y, 703252Y: TMP0 to TMP4
μPD703253Y: TMP0 to TMP5

<R> **3.** μPD703252Y only

<R> **4.** To realize low power consumption, stop the A/D converter and D/A converter before shifting to the IDLE1 mode.

21.5 IDLE2 Mode

21.5.1 Setting and operation status

The IDLE2 mode is set by setting the PSMR.PSM1 and PSMR.PSM0 bits to 10 and setting the PSC.STP bit to 1 in the normal operation mode.

In the IDLE2 mode, the clock oscillator continues operation but clock supply to the CPU, PLL, and other on-chip peripheral functions stops.

As a result, program execution stops and the contents of the internal RAM before the IDLE2 mode was set are retained. The CPU, PLL, and other on-chip peripheral functions stop operating. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Table 21-7 shows the operating status in the IDLE2 mode.

The IDLE2 mode can reduce the power consumption more than the IDLE1 mode because it stops the operations of the on-chip peripheral functions and PLL. However, because the PLL are stopped, a setup time for the PLL is required when the IDLE2 mode is released.

Cautions 1. Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE2 mode.

2. If the IDLE2 mode is set while an unmasked interrupt request signal is being held pending, the IDLE2 mode is released immediately by the pending interrupt request.

21.5.2 Releasing IDLE2 mode

The IDLE2 mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the IDLE2 mode, or reset signal ($\overline{\text{RESET}}$ pin input, reset signal (WDT2RES) generation by overflow of watchdog timer, or reset signal (CLMRES) generation by clock monitor (CLM)). The PLL returns to the operating status it was in before the IDLE2 mode was set.

After the IDLE2 mode has been released, the normal operation mode is restored.

(1) Releasing IDLE2 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The IDLE2 mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal. If the IDLE2 mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is processed as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the IDLE2 mode is released, but that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the IDLE2 mode is released and that interrupt request signal is acknowledged.

Caution The interrupt request signal that is disabled by setting the PSC.NMI1M, PSC.NMI0M, and PSC.INTM bits to 1 becomes invalid and IDLE2 mode is not released.

Table 21-6. Operation After Releasing IDLE2 Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address after securing the prescribed setup time.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed after securing the prescribed setup time.	The next instruction is executed after securing the prescribed setup time.

(2) Releasing IDLE2 mode by reset

The same operation as the normal reset operation is performed.

Table 21-7. Operating Status in IDLE2 Mode

	Setting of IDLE2 Mode / Item		Operating Status	
			When Subclock Is Not Used	When Subclock Is Used
	Main clock oscillator		Oscillation enabled	
	Subclock oscillator		–	Oscillation enabled
	Internal oscillator		Oscillation enabled	
	PLL		Stops operation	
	CPU		Stops operation	
<R>	DMA[Note 1]		Stops operation	
<R>	Interrupt controller		Stops operation (but standby release enabled)	
	ROM correction		Stops operation	
<R>	Timer P (TMP0 to TMP5[Note 2])		Stops operation	
	Timer M (TMM0)		Operable when f_R/8 is selected as the count clock	Operable when f_R/8 or f_{XT} is selected as the count clock
	Watch timer		Operable when f_X (divided BRG) is selected as the count clock	Operable
	Watchdog timer 2		Operable when f_R is selected as the count clock	Operable when f_R or f_{XT} is selected as the count clock
	Serial interface	CSIB0, CSIB1, CSIB3	Operable when the $\overline{\text{SCKBn}}$ input clock is selected as the count clock (n = 0, 1, 3)	
		I^2C00, I^2C01	Stops operation	
		UARTA0, UARTA1	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)	
<R>	IEBus controller[Note 3]		Stops operation	
<R>	CAN controller[Note 1]		Stops operation	
	A/D converter		Holds operation (conversion result held)[Note 4]	
<R>	D/A converter[Note 1]		Holds operation (output held[Note 4])	
	Key interrupt function (KR)		Operable	
	External bus interface		See **2.2 Pin States**.	
	Port function		Retains status before IDLE2 mode was set	
	Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the IDLE2 mode was set.	

<R> **Notes 1.** μPD703253Y only

<R> **2.** μPD703249Y, 703252Y: TMP0 to TMP4
μPD703253Y: TMP0 to TMP5

<R> **3.** μPD703252Y only

<R> **4.** To realize low power consumption, stop the A/D converter and D/A converter before shifting to the IDLE2 mode.

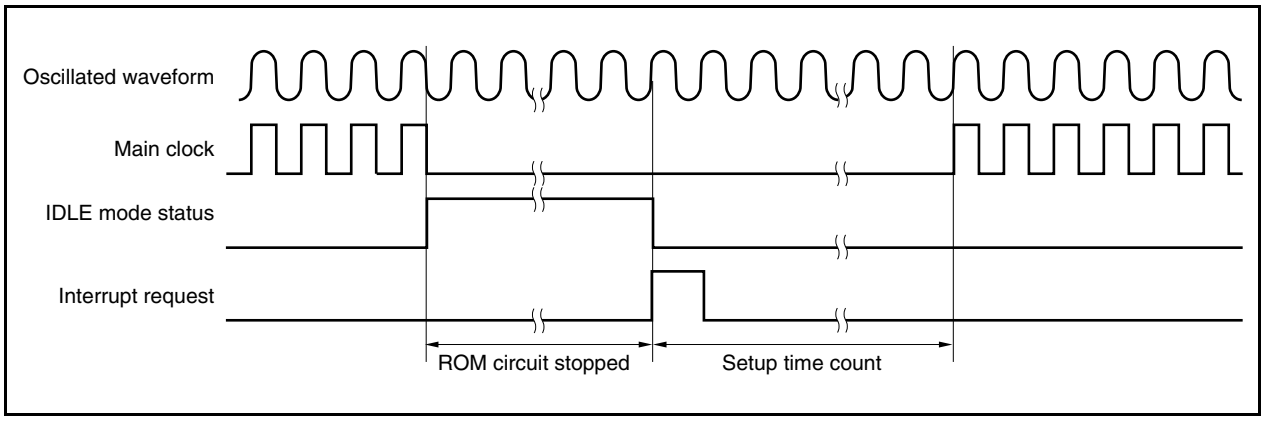
21.5.3 Securing setup time when releasing IDLE2 mode

Secure the setup time for the PLL after releasing the IDLE2 mode because the operation of the blocks other than the main clock oscillator stops after the IDLE2 mode is set.

(1) Releasing IDLE2 mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

Secure the specified setup time by setting the OSTS register.

When the releasing source is generated, the dedicated internal timer starts counting according to the OSTS register setting. When it overflows, the normal operation mode is restored.



(2) Release by reset ($\overline{\text{RESET}}$ pin input, WDT2RES generation)

This operation is the same as that of a normal reset.

The oscillation stabilization time is the initial value of the OSTS register, 2^{16}/fx.

21.6 STOP Mode

21.6.1 Setting and operation status

The STOP mode is set by setting the PSMR.PSM1 and PSMR.PSM0 bits to 01 or 11 and setting the PSC.STP bit to 1 in the normal operation mode.

In the STOP mode, the subclock oscillator continues operating but the main clock oscillator stops. Clock supply to the CPU and the on-chip peripheral functions is stopped.

As a result, program execution stops, and the contents of the internal RAM before the STOP mode was set are retained. The on-chip peripheral functions that operate with the clock oscillated by the subclock oscillator or an external clock continue operating.

Table 21-9 shows the operating status in the STOP mode.

Because the STOP mode stops operation of the main clock oscillator, it reduces the power consumption to a level lower than the IDLE2 mode. If the subclock oscillator, internal oscillator, and external clock are not used, the power consumption can be minimized with only leakage current flowing.

Cautions 1, Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the STOP mode.

2. If the STOP mode is set while an unmasked interrupt request signal is being held pending, the STOP mode is released immediately by the pending interrupt request.

21.6.2 Releasing STOP mode

The STOP mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the STOP mode, or reset signal ($\overline{\text{RESET}}$ pin input and reset signal (WDT2RES) generation by overflow of watchdog timer).

After the STOP mode has been released, the normal operation mode is restored after the oscillation stabilization time has been secured.

(1) Releasing STOP mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The STOP mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal. If the STOP mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the STOP mode is released, but that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the STOP mode is released and that interrupt request signal is acknowledged.

Caution The interrupt request that is disabled by setting the PSC.NMI1M, PSC.NMI0M, and PSC.INTM bits to 1 becomes invalid and STOP mode is not released.

Table 21-8. Operation After Releasing STOP Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address after securing the oscillation stabilization time.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed after securing the oscillation stabilization time.	The next instruction is executed after securing the oscillation stabilization time.

(2) Releasing STOP mode by reset

The same operation as the normal reset operation is performed.

Table 21-9. Operating Status in STOP Mode

	Setting of IDLE2 Mode / Item		Operating Status	
			When Subclock Is Not Used	When Subclock Is Used
	Main clock oscillator		Stops oscillation	
	Subclock oscillator		–	Oscillation enabled
	Internal oscillator		Oscillation enabled	
	PLL		Stops operation	
	CPU		Stops operation	
<R>	DMA[Note 1]		Stops operation	
<R>	Interrupt controller		Stops operation (but standby mode release enabled)	
	ROM correction		Stops operation	
<R>	Timer P (TMP0 to TMP5[Note 2])		Stops operation	
	Timer M (TMM0)		Operable when f_R/8 is selected as the count clock	Operable when f_R/8 or f_{XT} is selected as the count clock
	Watch timer		Stops operation	Operable when f_{XT} is selected as the count clock
	Watchdog timer 2		Operable when f_R is selected as the count clock	Operable when f_R or f_{XT} is selected as the count clock
	Serial interface	CSIB0, CSIB1, CSIB3	Operable when the $\overline{\text{SCKBn}}$ input clock is selected as the count clock (n = 0, 1, 3)	
		I^2C00, I^2C01	Stops operation	
		UARTA0, UARTA1	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)	
<R>	IEBus controller[Note 3]		Stops operation	
<R>	CAN controller[Note 1]		Stops operation	
	A/D converter		Stops operation (conversion result undefined)[Notes 4, 5]	
<R>	D/A converter[Note 1]		Stops operation[Notes 6, 7] (high impedance is output)	
	Key interrupt function (KR)		Operable	
	External bus interface		See **2.2 Pin States**.	
	Port function		Retains status before STOP mode was set	
	Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the STOP mode was set.	

<R> **Notes 1.** μPD703253Y only

<R> **2.** μPD703249Y, 703252Y: TMP0 to TMP4
μPD703253Y: TMP0 to TMP5

<R> **3.** μPD703252Y only

4. If the STOP mode is set while the A/D converter is operating, the A/D converter is automatically stopped and starts operating again after the STOP mode is released. However, in that case, the A/D conversion results after the STOP mode is released are invalid. All the A/D conversion results before the STOP mode is set are invalid.

5. Even if the STOP mode is set while the A/D converter is operating, the power consumption is reduced equivalently to when the A/D converter is stopped before the STOP mode is set.

<R> **6.** If the STOP mode is set while the D/A converter is operating, the D/A converter is automatically stopped and the pin status becomes high impedance. After the STOP mode is released, D/A conversion resumes, the setting time elapses, and the status returns to the output level before the STOP mode was set.

<R> **7.** Even if the STOP mode is set while the D/A converter is operating, the power consumption is reduced equivalently to when the D/A converter is stopped before the STOP mode is set.

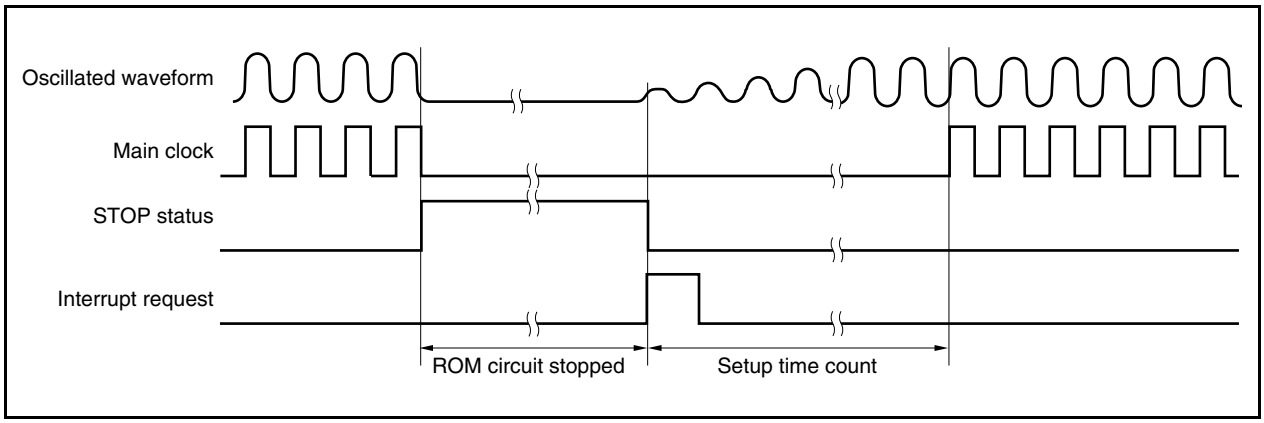
21.6.3 Securing oscillation stabilization time when releasing STOP mode

Secure the oscillation stabilization time for the main clock oscillator after releasing the STOP mode because the operation of the main clock oscillator stops after STOP mode is set.

(1) Releasing STOP mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

Secure the oscillation stabilization time by setting the OSTS register.

When the releasing source is generated, the dedicated internal timer starts counting according to the OSTS register setting. When it overflows, the normal operation mode is restored.



(2) Release by reset

This operation is the same as that of a normal reset.

The oscillation stabilization time is the initial value of the OSTS register, 2^{16}/fx.

21.7 Subclock Operation Mode

21.7.1 Setting and operation status

The subclock operation mode is set by setting the PCC.CK3 bit to 1 in the normal operation mode.

When the subclock operation mode is set, the internal system clock is changed from the main clock to the subclock. Check whether the clock has been switched by using the PCC.CLS bit.

When the PCC.MCK bit is set to 1, the operation of the main clock oscillator is stopped. As a result, the system operates only on the subclock.

In the subclock operation mode, the power consumption can be reduced to a level lower than in the normal operation mode because the subclock is used as the internal system clock. In addition, the power consumption can be further reduced to the level of the STOP mode by stopping the operation of the main clock oscillator.

Table 21-10 shows the operating status in subclock operation mode.

Cautions 1. When manipulating the CK3 bit, do not change the set values of the PCC.CK2 to PCC.CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended). For details of the PCC register, see 6.3 (1) Processor clock control register (PCC).

2. If the following conditions are not satisfied, change the CK2 to CK0 bits so that the conditions are satisfied and set the subclock operation mode.

Internal system clock (f_{CLK}) > Subclock (f_{XT} = 32.768 kHz) × 4

Remark Internal system clock: Clock generated from the main clock (f_{XX}) in accordance with the settings of the CK2 to CK0 bits

21.7.2 Releasing subclock operation mode

The subclock operation mode is released by a reset signal ($\overline{\text{RESET}}$ pin input, reset signal (WDT2RES) generation by overflow of watchdog timer, or reset signal (CLMRES) generation by clock monitor (CLM)) or by clearing the CK3 bit to 0.

If the main clock is stopped (MCK bit = 1), set the MCK bit to 1, secure the oscillation stabilization time of the main clock by software, and clear the CK3 bit to 0.

The normal operation mode is restored when the subclock operation mode is released.

Caution When manipulating the CK3 bit, do not change the set values of the CK2 to CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended).
For details of the PCC register, see 6.3 (1) Processor clock control register (PCC).

Table 21-10. Operating Status in Subclock Operation Mode

Setting of Subclock Operation Mode / Item		Operating Status	
		When Main Clock Is Oscillating	When Main Clock Is Stopped
Subclock oscillator		Oscillation enabled	
Internal oscillator		Oscillation enabled	
<R> PLL		Operable	Stops operation[Note 4]
CPU		Operable	
<R> DMA[Note 1]		Operable	
Interrupt controller		Operable	
ROM correction		Operable	
<R> Timer P (TMP0 to TMP5[Note 2])		Operable	Stops operation[Note 4]
Timer M (TMM0)		Operable	Operable when f_R/8 or f_{XT} is selected as the count clock
Watch timer		Operable	Operable when f_{XT} is selected as the count clock
Watchdog timer 2		Operable	Operable when f_R or f_{XT} is selected as the count clock
Serial interface	CSIB0, CSIB1, CSIB3	Operable	Operable when the $\overline{\text{SCKBn}}$ input clock is selected as the count clock (n = 0, 1, 3)
<R>	I^2C00, I^2C01	Operable	Stops operation[Note 4]
	UARTA0, UARTA1	Operable	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)
<R> IEBus controller[Note 3]		Operable	Stops operation[Note 4]
<R> CAN controller[Note 1]		Operable	Stops operation[Note 4]
<R> A/D converter		Operable	Stops operation[Note 4]
<R> D/A converter[Note 1]		Operable	
Key interrupt function (KR)		Operable	
<R> External bus interface		Operable	
Port function		Settable	
Internal data		Settable	

<R> **Notes 1.** μPD703253Y only

<R> **2.** μPD703249Y, 703252Y: TMP0 to TMP4
μPD703253Y: TMP0 to TMP5

<R> **3.** μPD703252Y only

<R> **4.** To stop the main clock, be sure to stop PLL (PLLCTL.PLLON bit = 0).
Also stop the internal peripheral functions that are operating on the main clock.

Caution When the CPU is operating on the subclock and main clock oscillation is stopped, accessing a register in which a wait occurs is disabled. If a wait is generated, it can be released only by reset (see 3.4.9 (2)).

21.8 Sub-IDLE Mode

21.8.1 Setting and operation status

The sub-IDLE mode is set by setting the PSMR.PSM1 and PSMR.PSM0 bits to 00 or 10 and setting the PSC.STP bit to 1 in the subclock operation mode.

In this mode, the clock oscillator continues operating but clock supply to the CPU and the other on-chip peripheral functions is stopped.

As a result, program execution stops and the contents of the internal RAM before the sub-IDLE mode was set are retained. The CPU and the other on-chip peripheral functions are stopped. However, the on-chip peripheral functions that can operate with the subclock or an external clock continue operating.

Because the sub-IDLE mode stops operation of the CPU and other on-chip peripheral functions, it can reduce the power consumption more than the subclock operation mode. If the sub-IDLE mode is set after the main clock has been stopped, the current consumption can be reduced to a level as low as that in the STOP mode.

Table 21-12 shows the operating status in the sub-IDLE mode.

Cautions 1. Following the store instruction to the PSC register for setting the sub-IDLE mode, insert the five or more NOP instructions.

2. If the sub-IDLE mode is set while an unmasked interrupt request signal is being held pending, the sub-IDLE mode is then released immediately by the pending interrupt request.

21.8.2 Releasing sub-IDLE mode

The sub-IDLE mode is released by a non-maskable interrupt request signal (NMI pin input, INTWDT2 signal), unmasked external interrupt request signal (INTP0 to INTP7 pin input), unmasked internal interrupt request signal from the peripheral functions operable in the sub-IDLE mode, or reset signal ($\overline{\text{RESET}}$ pin input, reset signal (WDT2RES) generation by overflow of watchdog timer, or reset signal (CLMRES) generation by clock monitor (CLM)). The PLL returns to the operating status it was in before the sub-IDLE mode was set.

When the sub-IDLE mode is released by an interrupt request signal, the subclock operation mode is set.

(1) Releasing sub-IDLE mode by non-maskable interrupt request signal or unmasked maskable interrupt request signal

The sub-IDLE mode is released by a non-maskable interrupt request signal or an unmasked maskable interrupt request signal, regardless of the priority of the interrupt request signal.

If the sub-IDLE mode is set in an interrupt servicing routine, however, an interrupt request signal that is issued later is serviced as follows.

(a) If an interrupt request signal with a priority lower than that of the interrupt request currently being serviced is issued, the sub-IDLE mode is released, but that interrupt request signal is not acknowledged. The interrupt request signal itself is retained.

(b) If an interrupt request signal with a priority higher than that of the interrupt request currently being serviced is issued (including a non-maskable interrupt request signal), the sub-IDLE mode is released and that interrupt request signal is acknowledged.

Cautions 1. The interrupt request signal that is disabled by setting the PSC.NMI1M, PSC.NMI0M, and PSC.INTM bits to 1 becomes invalid and sub-IDLE mode is not released.

2. When the sub-IDLE mode is released, 12 cycles of the subclock (about 366 μs) elapse from when the interrupt request signal that releases the sub-IDLE mode is generated to when the mode is released.

Table 21-11. Operation After Releasing Sub-IDLE Mode by Interrupt Request Signal

Release Source	Interrupt Enabled (EI) Status	Interrupt Disabled (DI) Status
Non-maskable interrupt request signal	Execution branches to the handler address.	
Maskable interrupt request signal	Execution branches to the handler address or the next instruction is executed.	The next instruction is executed.

(2) Releasing sub-IDLE mode by reset

The same operation as the normal reset operation is performed.

Table 21-12. Operating Status in Sub-IDLE Mode

Setting of Sub-IDLE Mode / Item		Operating Status	
		When Main Clock Is Oscillating	When Main Clock Is Stopped
Subclock oscillator		Oscillation enabled	
Internal oscillator		Oscillation enabled	
PLL		Operable	Stops operation[Note 4]
CPU		Stops operation	
<R> DMA[Note 1]		Stops operation	
<R> Interrupt controller		Stops operation (but standby mode release enabled)	
ROM correction		Stops operation	
<R> Timer P (TMP0 to TMP5[Note 2])		Stops operation	
Timer M (TMM0)		Operable when $f_R/8$ or f_{XT} is selected as the count clock	
Watch timer		Stops operation	Operable when f_{XT} is selected as the count clock
Watchdog timer 2		Operable when f_R or f_{XT} is selected as the count clock	
Serial interface	CSIB0, CSIB1, CSIB3	Operable when the $\overline{\text{SCKBn}}$ input clock is selected as the count clock (n = 0, 1, 3)	
	I^2C00, I^2C01	Stops operation	
	UARTA0, UARTA1	Stops operation (but UARTA0 is operable when the ASCKA0 input clock is selected)	
<R> IEBus controller[Note 3]		Stops operation	
<R> CAN controller[Note 1]		Stops operation	
A/D converter		Holds operation (conversion result held)[Note 5]	
<R> D/A converter[Note 1]		Holds operation (output held[Note 5])	
Key interrupt function (KR)		Operable	
External bus interface		See **2.2 Pin States** (same operation status as IDLE1, IDLE2 mode).	
Port function		Retains status before sub-IDLE mode was set	
Internal data		The CPU registers, statuses, data, and all other internal data such as the contents of the internal RAM are retained as they were before the sub-IDLE mode was set.	

Notes 1. <R> μPD703253Y only

2. <R> μPD703249Y, 703252Y: TMP0 to TMP4
μPD703253Y: TMP0 to TMP5

3. <R> μPD703252Y only

4. Be sure to stop the PLL (PLLCTL.PLLON bit = 0) before stopping the main clock.

5. <R> To realize low power consumption, stop the A/D and D/A converters before shifting to the sub-IDLE mode.

CHAPTER 22 RESET FUNCTIONS

22.1 Overview

The following reset functions are available.

(1) Three kinds of reset sources
- External reset input via the $\overline{\text{RESET}}$ pin
- Reset via the watchdog timer 2 (WDT2) overflow (WDT2RES)
- System reset via the detecting clock monitor (CLM) oscillation stop

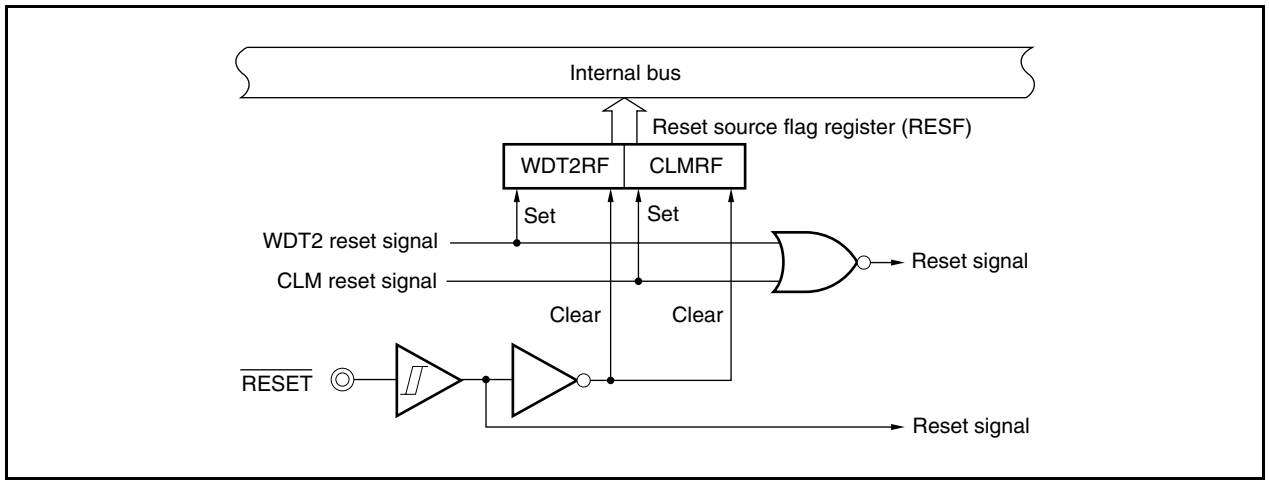
After a reset is released, the source of the reset can be confirmed with the reset source flag register (RESF).

(2) Emergency operation mode

<R> If the WDT2 overflows during the main clock oscillation stabilization time inserted after reset release or STOP mode release, a main clock oscillation anomaly is judged and the CPU starts operating on the internal oscillation clock.

Caution When the CPU operates on the internal oscillation clock, access to the register in which a wait state is generated is prohibited. For the register in which a wait state is generated, see 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.

Figure 22-1. Block Diagram of Reset Function



22.2 Registers to Check Reset Source

The V850ES/SG1 has three kinds of reset sources. After a reset has been released, the source of the reset that occurred can be checked with the reset source flag register (RESF).

(1) Reset source flag register (RESF)

The RESF register is a special register that can be written only by a combination of specific sequences (see **3.4.8 Special registers**).

The RESF register indicates the source from which a reset signal is generated.

This register is read or written in 8-bit or 1-bit units.

$\overline{\text{RESET}}$ pin sets this register to 00H. The default value differs if the source of reset is other than the $\overline{\text{RESET}}$ pin signal.

After reset: 00H[Note] R/W Address: FFFFF888H

	7	6	5	4	3	2	1	0
RESF	0	0	0	WDT2RF	0	0	CLMRF	0

WDT2RF	Reset signal from WDT2
0	Not generated
1	Generated

CLMRF	Reset signal from CLM
0	Not generated
1	Generated

Note The value of the RESF register is cleared to 00H when a reset is executed via the $\overline{\text{RESET}}$ pin. When a reset is executed by the watchdog timer 2 (WDT2) or clock monitor (CLM), the reset flags of this register (WDT2RF bit and CLMRF bit) are set. However, other sources are retained.

Caution Only "0" can be written to each bit of this register. If writing "0" conflicts with setting the flag (occurrence of reset), setting the flag takes precedence.

22.3 Operation

22.3.1 Reset operation via $\overline{\text{RESET}}$ pin

When a low level is input to the $\overline{\text{RESET}}$ pin, the system is reset, and each hardware unit is initialized.

When the level of the $\overline{\text{RESET}}$ pin is changed from low to high, the reset status is released.

Table 22-1. Hardware Status on $\overline{\text{RESET}}$ Pin Input

Item	During Reset	After Reset
Main clock oscillator (f_X)	Oscillation stops	Oscillation starts
Subclock oscillator (f_{XT})	Oscillation continues	
Internal oscillator	Oscillation stops	Oscillation starts
Peripheral clock (f_{XX}/1,024)	Operation stops	Operation starts after securing oscillation stabilization time
Internal system clock (f_{CLK}), CPU clock (f_{CPU})	Operation stops	Operation starts after securing oscillation stabilization time (initialized to f_{XX}/8)
CPU	Initialized	Program execution starts after securing oscillation stabilization time
Watchdog timer 2	Operation stops (initialized to 0)	Counts up from 0 with internal oscillation clock as source clock.
Internal RAM	Undefined if power-on reset or CPU access and reset input conflict (data is damaged). Otherwise value immediately after reset input is retained.	
I/O lines (ports/alternate-function pins)	High impedance[Note 1]	
On-chip peripheral I/O registers	Initialized to specified status, OCDM register is set (01H).	
Other on-chip peripheral functions	Operation stops	Operation can be started after securing oscillation stabilization time

<R> **Notes 1.** The following pins may momentarily output an undefined level upon power application.

<R> • P10/ANO0[Note 2] pin

<R> • P11/ANO1[Note 2] pin

• P53/KR3 pin

<R> **2.** μPD703253Y only

Caution The OCDM register is initialized by the $\overline{\text{RESET}}$ pin input. Therefore, if a high level is input to the P05/INTP2 pin after a reset release before the OCDM.OCDM0 bit is cleared, the pull-down resistor connected to the P05/INTP2 pin is controlled. For details, see CHAPTER 4 PORT FUNCTIONS.

Figure 22-2. Timing of Reset Operation by $\overline{\text{RESET}}$ Pin Input

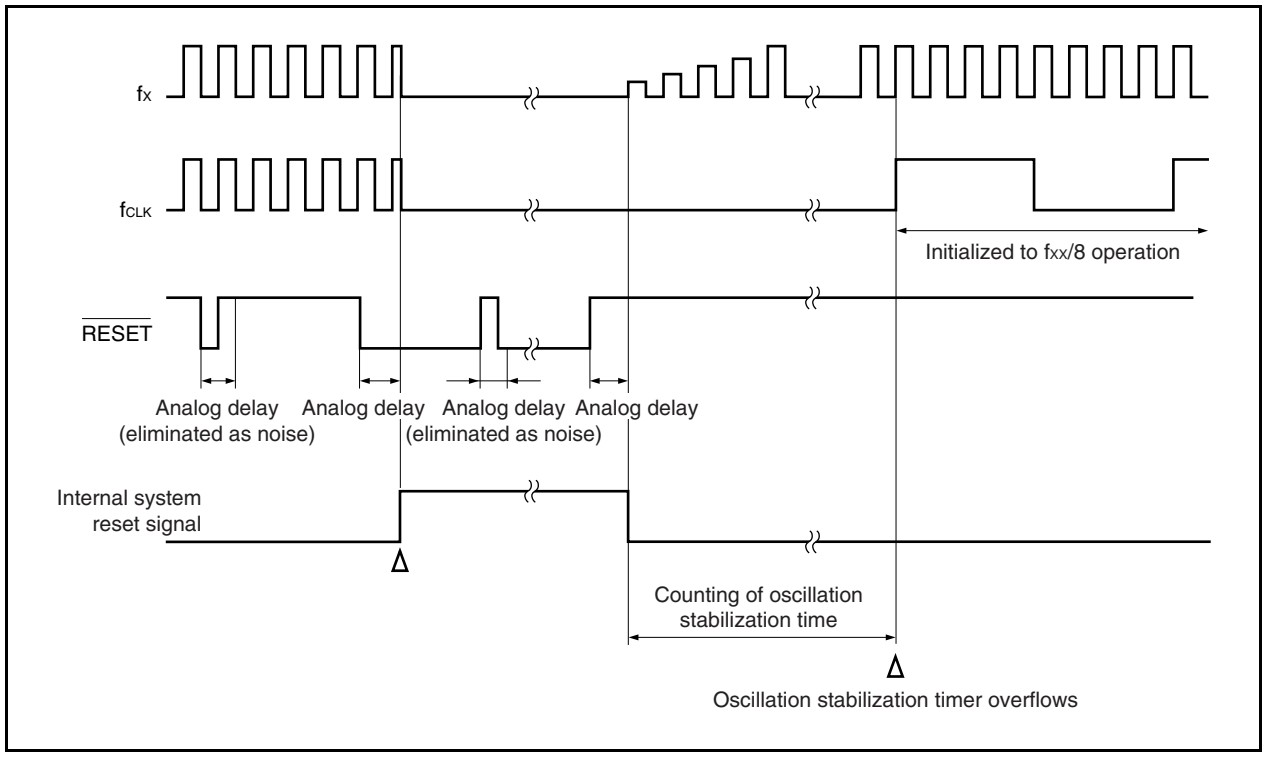
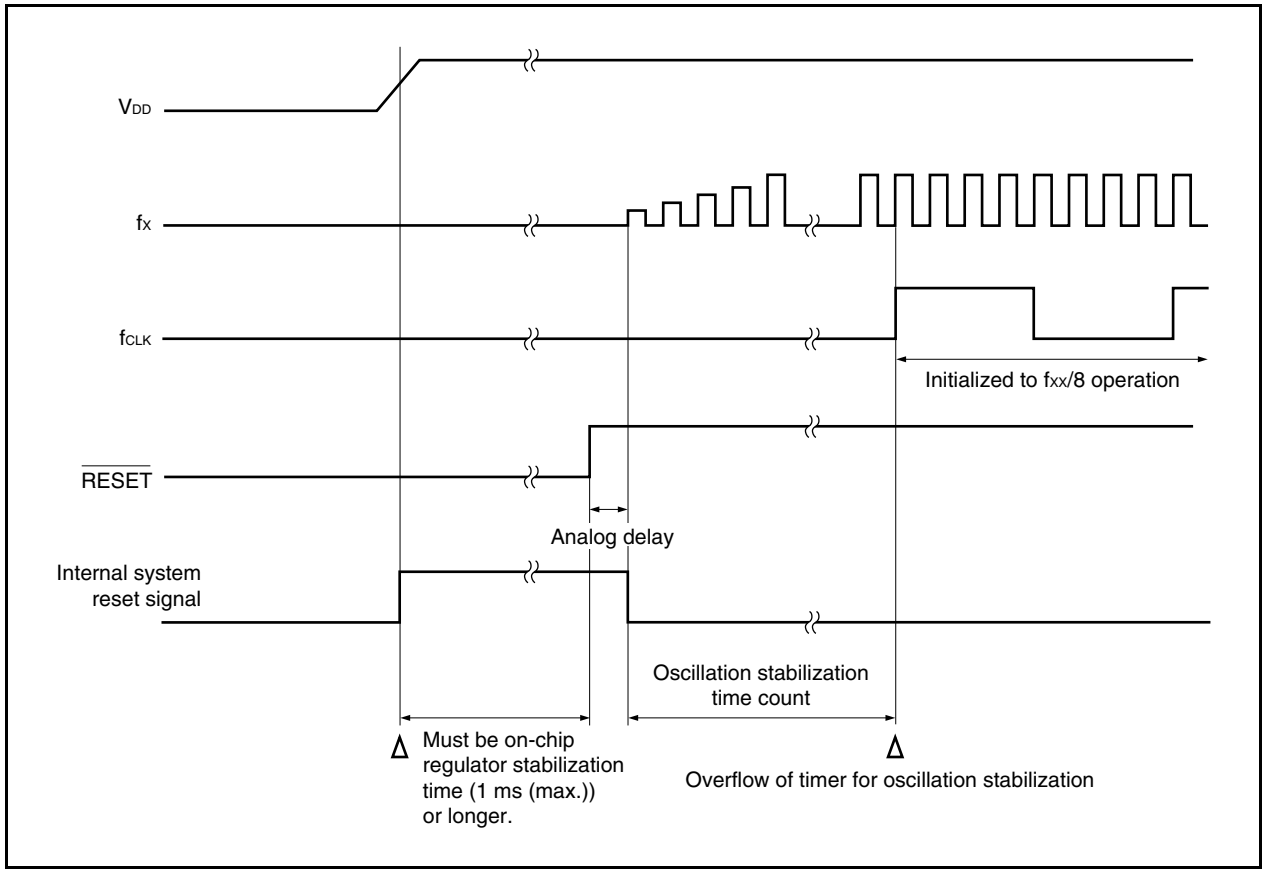


Figure 22-3. Timing of Power-on Reset Operation



22.3.2 Reset operation by watchdog timer 2 (WDT2RES)

When watchdog timer 2 is set to the reset operation mode due to overflow, upon watchdog timer 2 overflow (WDT2RES signal generation), a system reset is executed and the hardware is initialized to the initial status.

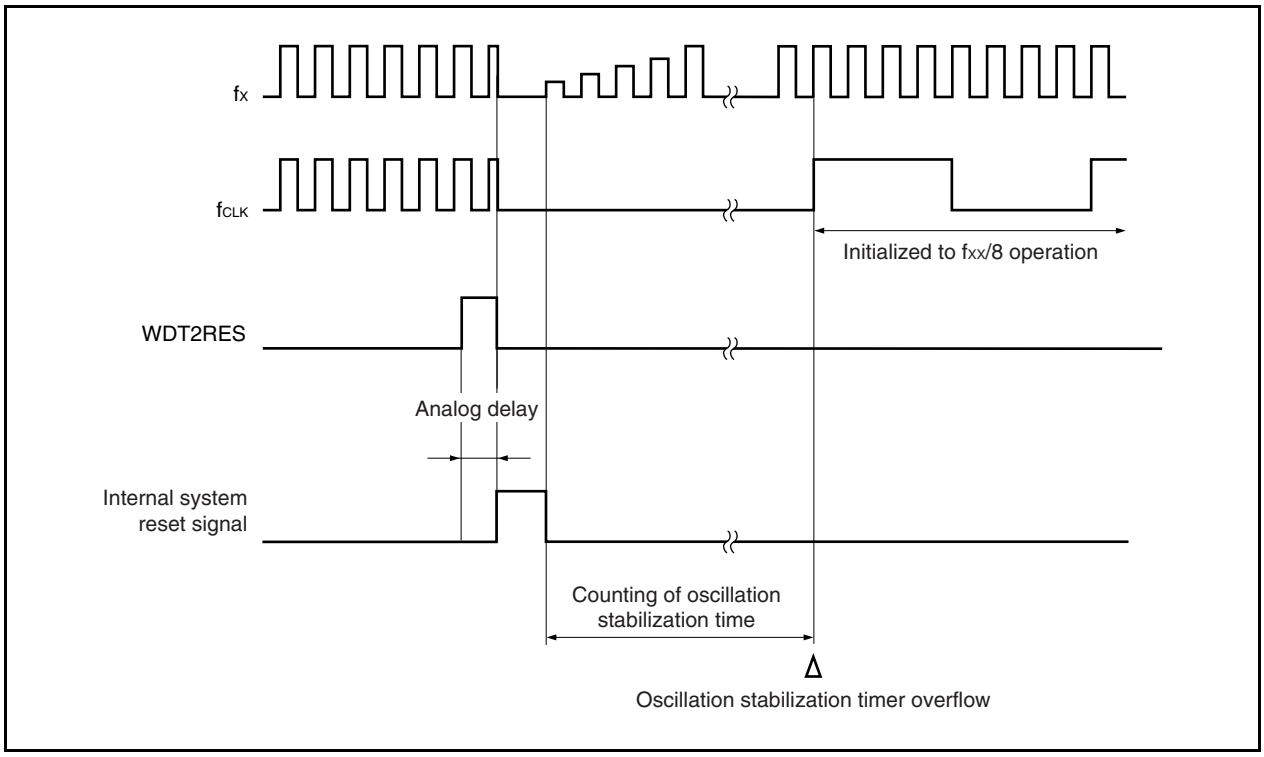
Following watchdog timer 2 overflow, the reset status is entered and lasts the predetermined time (analog delay), and the reset status is then automatically released.

The main clock oscillator is stopped during the reset period.

Table 22-2. Hardware Status During Watchdog Timer 2 Reset Operation

Item	During Reset	After Reset
Main clock oscillator (f_X)	Oscillation stops	Oscillation starts
Subclock oscillator (f_{XT})	Oscillation continues	
Internal oscillator	Oscillation stops	Oscillation starts
Peripheral clock (f_{XX} to f_{XX}/1,024)	Operation stops	Operation starts after securing oscillation stabilization time
Internal system clock (f_{CLK}), CPU clock (f_{CPU})	Operation stops	Operation starts after securing oscillation stabilization time (initialized to f_{XX}/8)
CPU	Initialized	Program execution after securing oscillation stabilization time
Watchdog timer 2	Operation stops (initialized to 0)	Counts up from 0 with internal oscillation clock as source clock.
Internal RAM	Undefined if power-on reset or CPU access and reset input conflict (data is damaged). Otherwise value immediately after reset input is retained.	
I/O lines (ports/alternate-function pins)	High impedance	
On-chip peripheral I/O register	Initialized to specified status, OCDM register retains its value.	
On-chip peripheral functions other than above	Operation stops	Operation can be started after securing oscillation stabilization time.

Figure 22-4. Timing of Reset Operation by WDT2RES Signal Generation



<R> 22.3.3 Reset operation by clock monitor (CLMRES)

When the clock monitor operation is enabled, the main clock is monitored by using the sampling clock (internally oscillated clock: f_R). If it is detected that the main clock is stopped, the system is reset and each hardware unit is initialized to a specific status.

After it has been detected that the main clock is stopped, the CPU is placed in the reset status for a specific time (of analog delay), and then it is automatically released from the reset status. After the reset status has been released, the timer for oscillation stabilization does not perform its counting operation, because the main clock is stopped. When watchdog timer 2 that is started by default overflows, the CPU starts program execution on an internally oscillated clock (f_R).

The status of each hardware unit during the reset period executed by the reset signal (CLMRES) of the clock monitor operation and after the reset status is released is shown below.

For the reset timing by the clock monitor operation, refer to **Figure 22-5**.

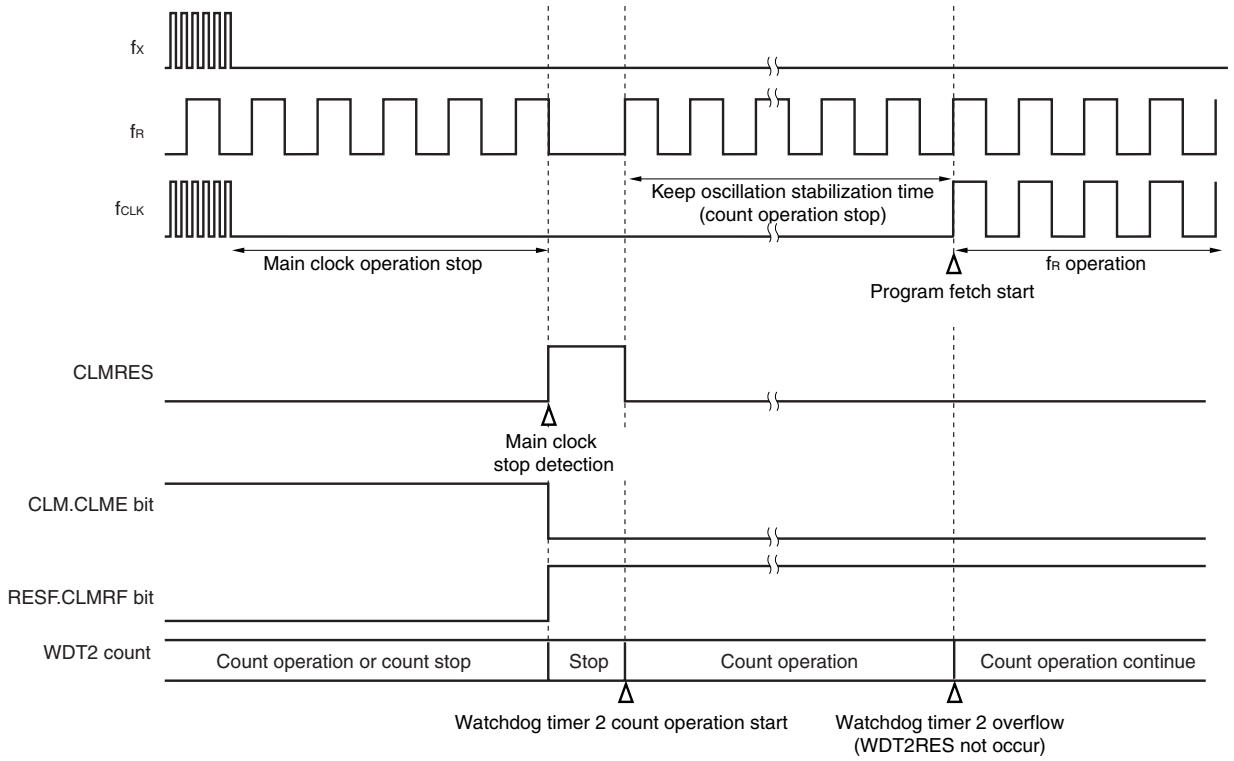
Table 22-3. Hardware Status During Reset Operation by Clock Monitor

Item	During Reset	After Reset
Main clock oscillator (f_X)	Oscillation stops	Oscillation starts[Note]
Subclock oscillator (f_{XT})	Oscillation continues	
Internal oscillator	Oscillation stops	Oscillation starts
Peripheral clock (f_{XX} to f_{XX}/1,024)	Operation stops	Operation starts after securing oscillation stabilization time[Note].
Internal system clock (f_{CLK}), CPU clock (f_{CPU})	Operation stops	Operation starts after securing oscillation stabilization time (initialized to f_{XX}/8). However, if watchdog timer 2 overflows before the CPU execution, operation starts with the internal oscillation clock (f_R).
CPU	Initialized	Program execution starts after securing oscillation stabilization time. However, if watchdog timer 2 overflows before the CPU execution, operation starts with the internal oscillation clock (f_R).
WDT2	Operation stops (initialized to 0)	Operation starts. WDT2RES is not generated, however, if only watchdog timer 2 overflows before CPU execution.
Internal RAM	Undefined	
I/O lines (ports/alternate-function pins)	High impedance	
On-chip peripheral I/O register	Initialized to specified status, OCDM register retains its value.	
On-chip peripheral functions other than above	Operation stops	Operation can be started after securing oscillation stabilization time[Note].

Note When the main clock starts oscillation after the reset operation by the clock monitor.

Remark For details of the clock monitor, refer to **CHAPTER 23 CLOCK MONITOR**.

Figure 22-5. Reset Timing by Clock Monitor



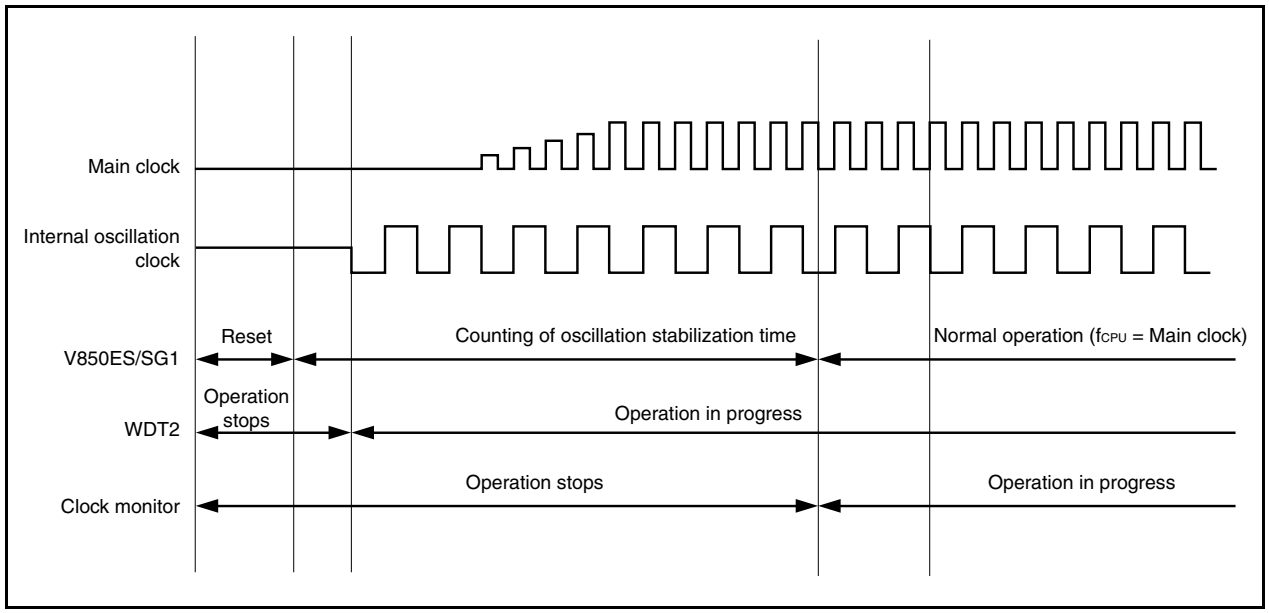
Remark The mode cannot be restored by software to the normal operation mode from the internally oscillated clock operation mode. The normal operation mode can be restored only when the main clock oscillation (fx) operates normally after reset (generation of $\overline{\text{RESET}}$, WDT2RES signals).

22.3.4 Operation after reset release

After the reset is released, the main clock starts oscillation and oscillation stabilization time (OSTS register initial value: 2^{16}/fx) is secured, and the CPU starts program execution.

WDT2 immediately begins to operate after a reset has been released using the internal oscillation clock as a source clock.

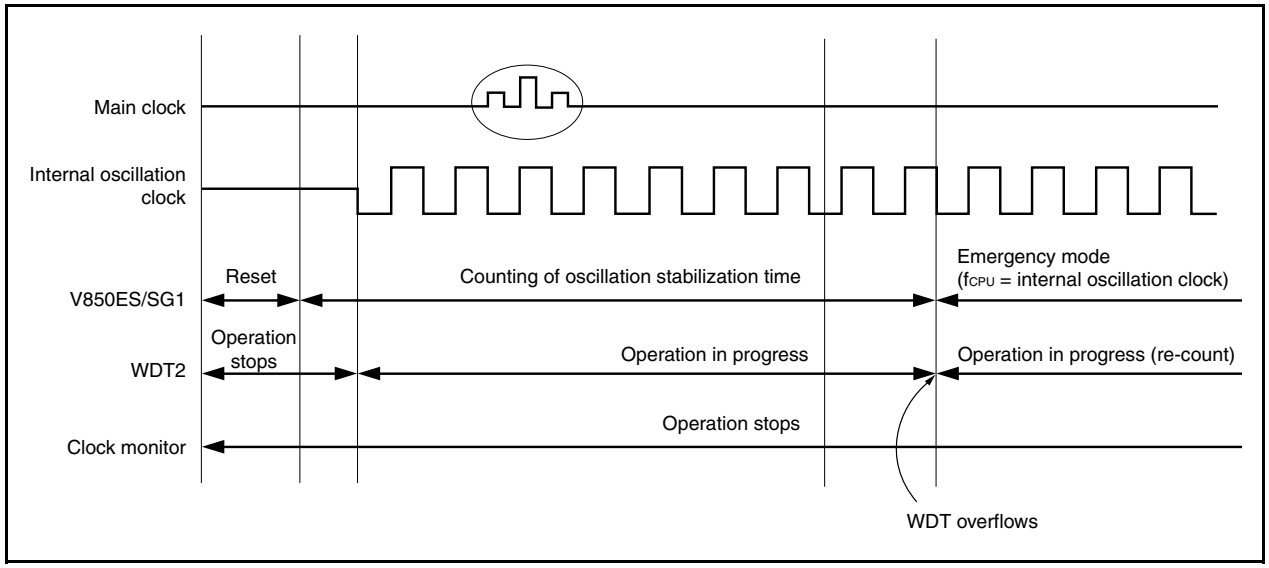
Figure 22-6. Operation After Reset Release



(1) Emergent operation mode

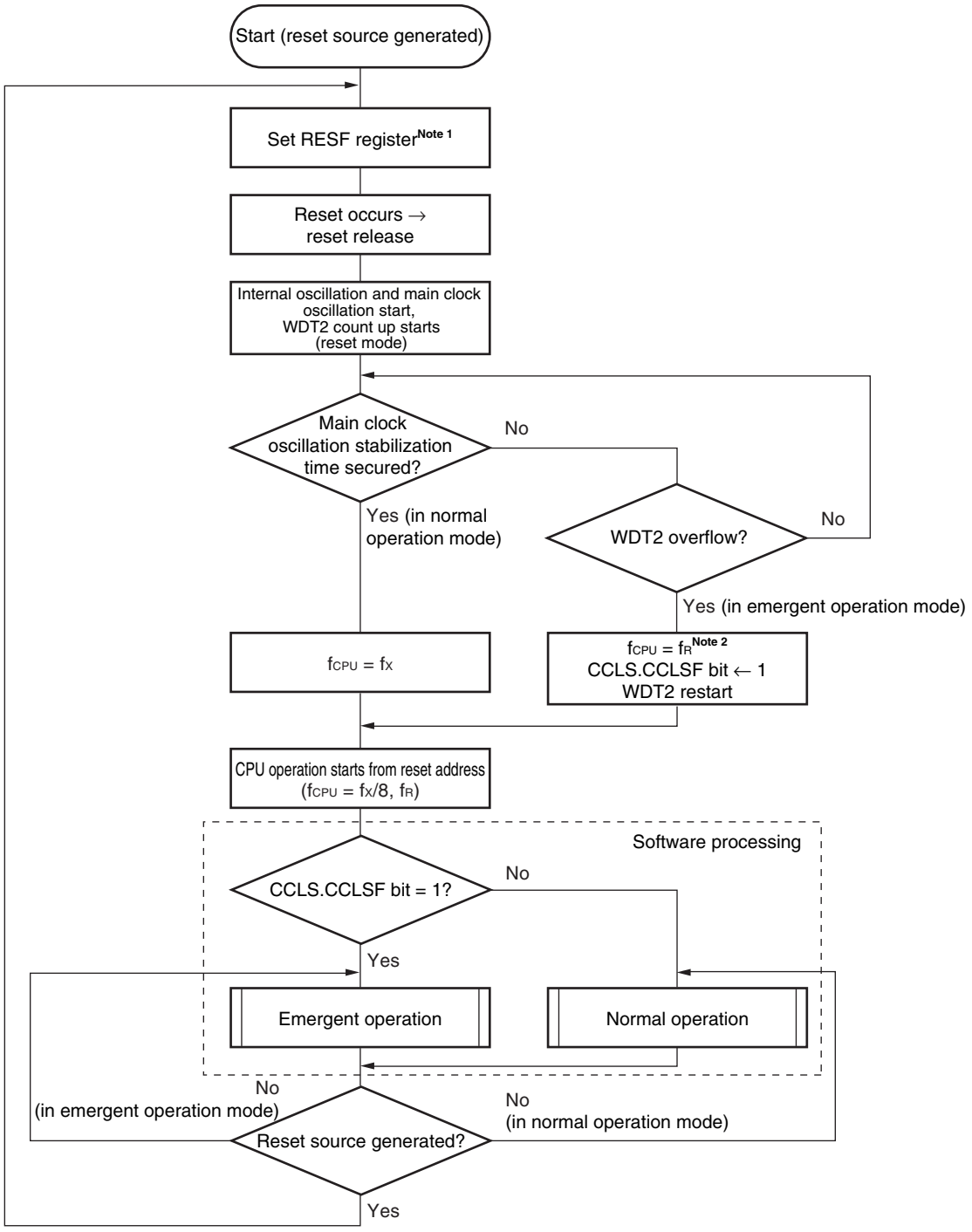
If an anomaly occurs in the main clock before oscillation stabilization time is secured, the WDT2 overflows before executing the CPU program. At this time, the CPU starts program execution by using the internal oscillation clock as the source clock.

Figure 22-7. Operation After Reset Release



The CPU operation clock states can be checked with the CPU operation clock status register (CCLS).

22.3.5 Reset function operation flow



Notes 1. Bit to be set differs depending on the reset source.

Reset Source	WDT2RF Bit	CRMRF Bit
$\overline{\text{RESET}}$ pin	0	0
WDT2	1	Value before reset is retained.
CLM	Value before reset is retained.	1

2. The internal oscillator cannot be stopped.

22.4 Valid/Invalid of Internal RAM Data

<R> A function to indicate whether the internal RAM data is valid is provided only in the μPD703253Y.

(1) Internal RAM data status register (RAMS)

The RAMS register is a special register. This can be written only in a special combination of sequences (see **3.4.8 Special registers**).

This register is a flag register that indicates whether the internal RAM is valid or not.

This register can be read or written in 8-bit or 1-bit units.

The set/clear conditions for the RAMF bit are shown below.

- Setting conditions: Detection of voltage lower than specified level
  Set by instruction
  Generation of reset signal by WDT2 or CLM
  Generation of reset signal via the $\overline{\text{RESET}}$ pin while internal RAM is being accessed.
- Clearing condition: Writing of 0 in specific sequence

After reset: 01H[Note] R/W Address: FFFFF892H

	7	6	5	4	3	2	1	<0>
RAMS	0	0	0	0	0	0	0	RAMF

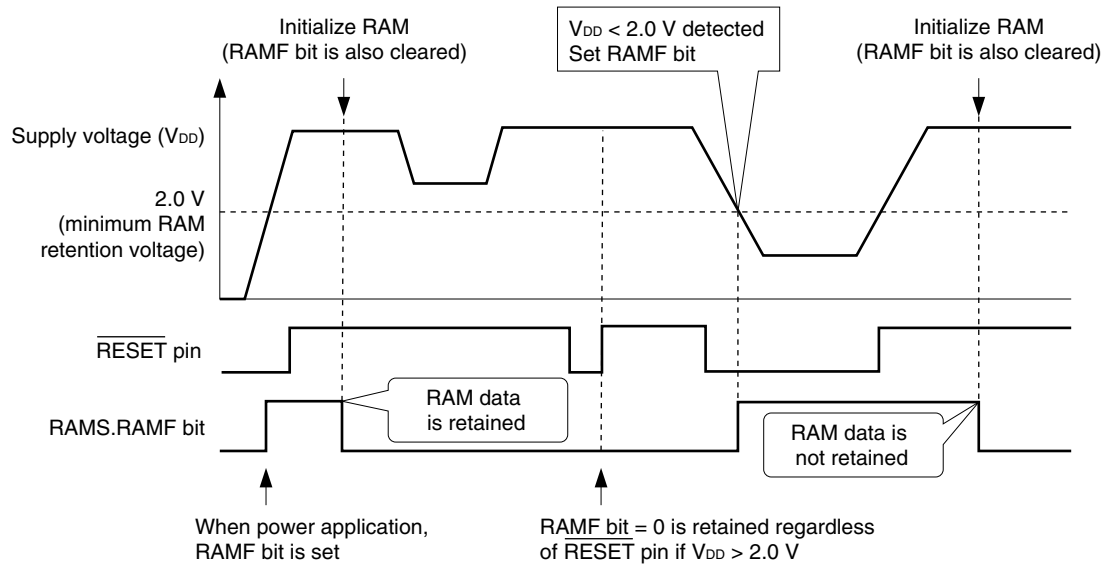
RAMF	Internal RAM data valid/invalid
0	Valid
1	Invalid

Note This register is set to 01H after reset by the $\overline{\text{RESET}}$ pin input (only for RAM access), watchdog timer 2 overflow, or clock monitor. After reset by other sources, the register value at that time will be retained.

22.4.1 RAM retention voltage detection operation

The supply voltage and detected voltage are compared. When the supply voltage drops below the detected voltage (including on power application), the RAMS.RAMF bit is set to 1.

Figure 22-8. Operation Timing of RAM Retention Voltage Detection Function



Remarks 1. The RAMF bit is set to 1 if the supply voltage drops under the minimum RAM retention voltage (2.0 V (TYP.)).

2. The RAMF bit operates regardless of the $\overline{\text{RESET}}$ pin status.

22.4.2 Emulation function

When an in-circuit emulator is used, the operation of the RAM retention flag (RAMS.RAMF bit) can be pseudo-controlled and emulated by manipulating the PEMU1 register on the debugger.

This register is valid only in the emulation mode. It is invalid in the normal mode.

(1) Peripheral emulation register 1 (PEMU1)

After reset: 00H R/W Address: FFFFF9FEH

	7	6	5	4	3	2	1	0
PEMU1	0	0	0	0	0	EVARAMIN	0	0

EVARAMIN	Pseudo specification of RAM retention voltage detection signal
0	Do not detect voltage lower than RAM retention voltage.
1	Detect voltage lower than RAM retention voltage (set RAMF flag).

Caution This bit is not automatically cleared.

[Usage]

When an in-circuit emulator is used, pseudo emulation of RAMF is implemented by rewriting this register on the debugger.

<1> CPU break (CPU operation stops.)

<2> Set the EVARAMIN bit to 1 by using a register write command.
By setting the EVARAMIN bit to 1, the RAMF bit is set to 1 on hardware (the internal RAM data is invalid).

<3> Clear the EVARAMIN bit to 0 by using a register write command again.
Unless this operation is performed (clearing the EVARAMIN bit to 0), the RAMF bit cannot be cleared to 0 by a CPU operation instruction.

<4> Run the CPU and resume emulation.

CHAPTER 23 CLOCK MONITOR

23.1 Functions

The clock monitor samples the main clock by using the internal oscillation clock and generates a reset request signal (CLMRES) when oscillation of the main clock is stopped.

Once the operation of the clock monitor has been enabled by an operation enable flag, it cannot be cleared to 0 by any means other than reset.

When a reset by the clock monitor (CLMRES) occurs, the RESF.CLMRF bit is set. For details on the RESF register, see **22.2 Registers to Check Reset Source**.

The clock monitor automatically stops under the following conditions.

- During oscillation stabilization time after STOP mode is released
- When the main clock is stopped (from when the PCC.MCK bit = 1 during subclock operation, until the PCC.CLS bit = 0 during main clock operation)
- When the sampling clock (internal oscillation clock) is stopped
- When the CPU operates with the internal oscillation clock

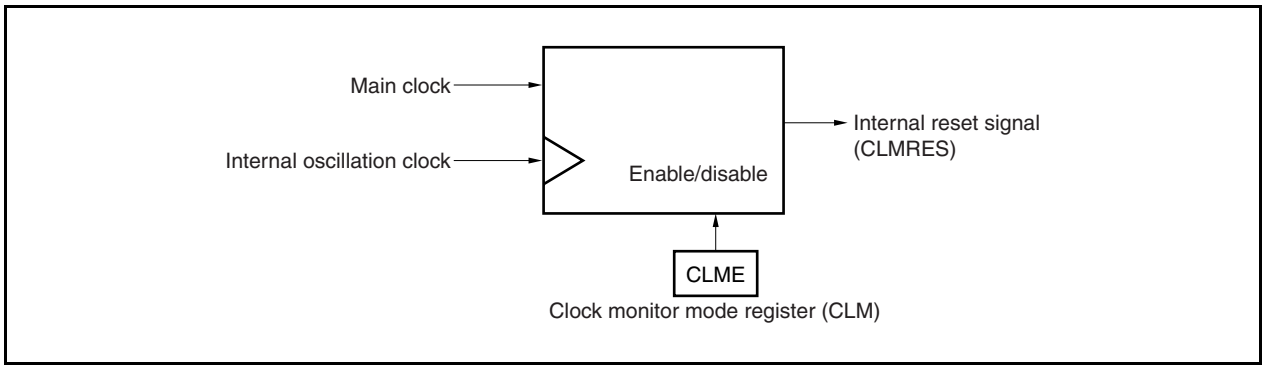
23.2 Configuration

The clock monitor includes the following hardware.

Table 23-1. Configuration of Clock Monitor

Item	Configuration
Control register	Clock monitor mode register (CLM)

Figure 23-1. Timing of Reset via Clock Monitor



23.3 Register

The clock monitor is controlled by the clock monitor mode register (CLM).

(1) Clock monitor mode register (CLM)

The CLM register is a special register. This can be written only in a special combination of sequences (see **3.4.8 Special registers**).

This register is used to set the operation mode of the clock monitor.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF870H

	7	6	5	4	3	2	1	<0>
CLM	0	0	0	0	0	0	0	CLME

CLME	Clock monitor operation enable or disable
0	Disable clock monitor operation.
1	Enable clock monitor operation.

Cautions 1. Once the CLME bit has been set to 1, it cannot be cleared to 0 by any means other than reset.

2. When a reset by the clock monitor occurs, the CLME bit is cleared to 0 and the RESF.CLMRF bit is set to 1.

23.4 Operation

This section explains the functions of the clock monitor. The start and stop conditions are as follows.

<Start condition>

Enabling operation by setting the CLM.CLME bit to 1

<Stop conditions>

- While oscillation stabilization time is being counted after STOP mode is released
- When the main clock is stopped (from when PCC.MCK bit = 1 during subclock operation to when PCC.CLS bit = 0 during main clock operation)
- When the sampling clock (internal oscillation clock) is stopped
- When the CPU operates with the internal oscillation clock

Table 23-2. Operation Status of Clock Monitor
(When CLM.CLME Bit = 1, During Internal Oscillation Clock Operation)

CPU Operating Clock	Operation Mode	Status of Main Clock	Status of Internal Oscillation Clock	Status of Clock Monitor
Main clock	HALT mode	Oscillates	Oscillates[Note 1]	Operates[Note 2]
	IDLE1, IDLE2 modes	Oscillates	Oscillates[Note 1]	Operates[Note 2]
	STOP mode	Stops	Oscillates[Note 1]	Stops
<R> Subclock (MCK bit of PCC register = 0)	Sub-IDLE mode	Oscillates	Oscillates[Note 1]	Operates[Note 2]
<R> Subclock (MCK bit of PCC register = 1)	Sub-IDLE mode	Stops	Oscillates[Note 1]	Stops
Internal oscillation clock	–	Stops	Oscillates[Note 3]	Stops
During reset	–	Stops	Stops	Stops

Notes 1. Internal oscillator can be stopped by setting the RCM.RSTOP bit to 1.

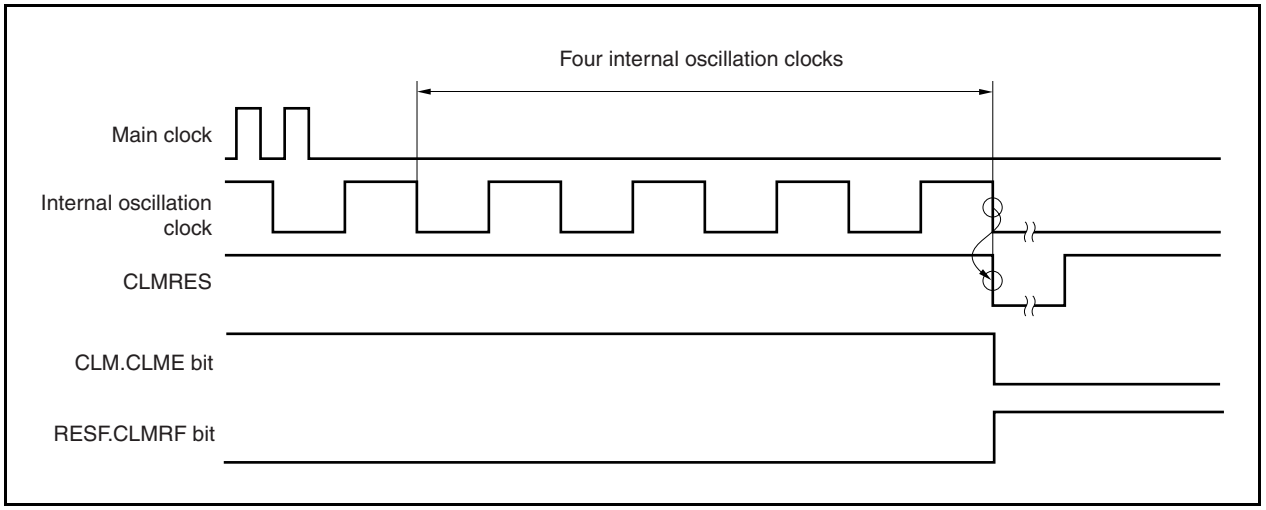
2. The clock monitor is stopped while internal oscillator is stopped.

3. Internal oscillator cannot be stopped by software.

(1) Operation when main clock oscillation is stopped (CLME bit = 1)

If oscillation of the main clock is stopped when the CLME bit = 1, an internal reset signal (CLMRES) is generated as shown in Figure 23-2.

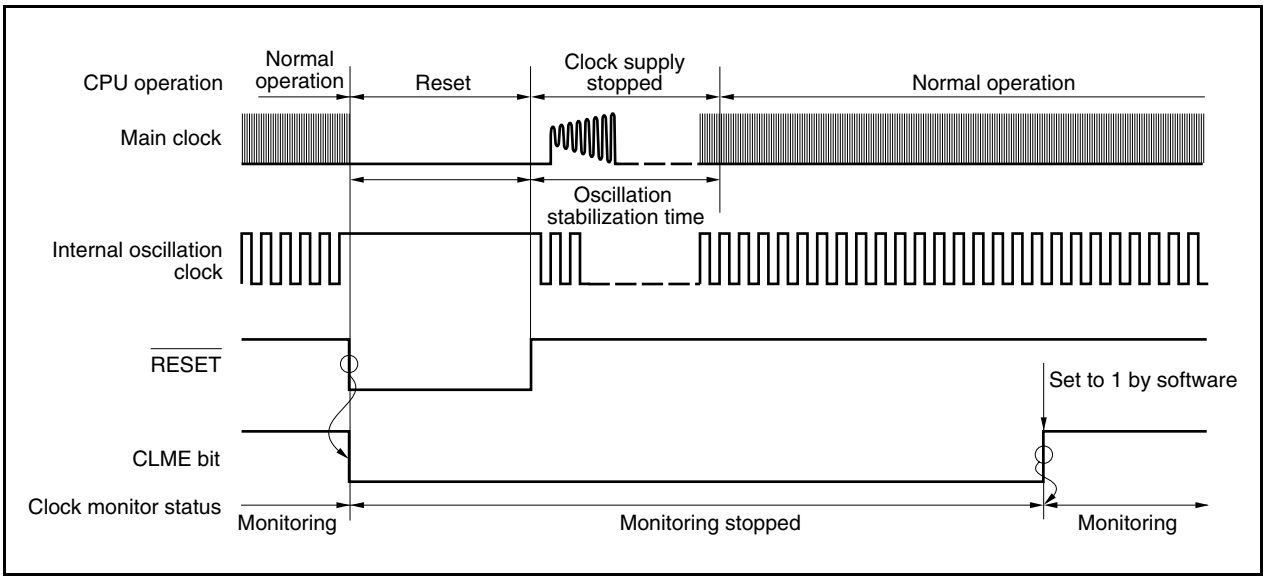
Figure 23-2. Reset Period Due to That Oscillation of Main Clock Is Stopped



(2) Clock monitor status after $\overline{\text{RESET}}$ input

$\overline{\text{RESET}}$ input clears the CLM.CLME bit to 0 and stops the clock monitor operation. When CLME bit is set to 1 by software at the end of the oscillation stabilization time of the main clock, monitoring is started.

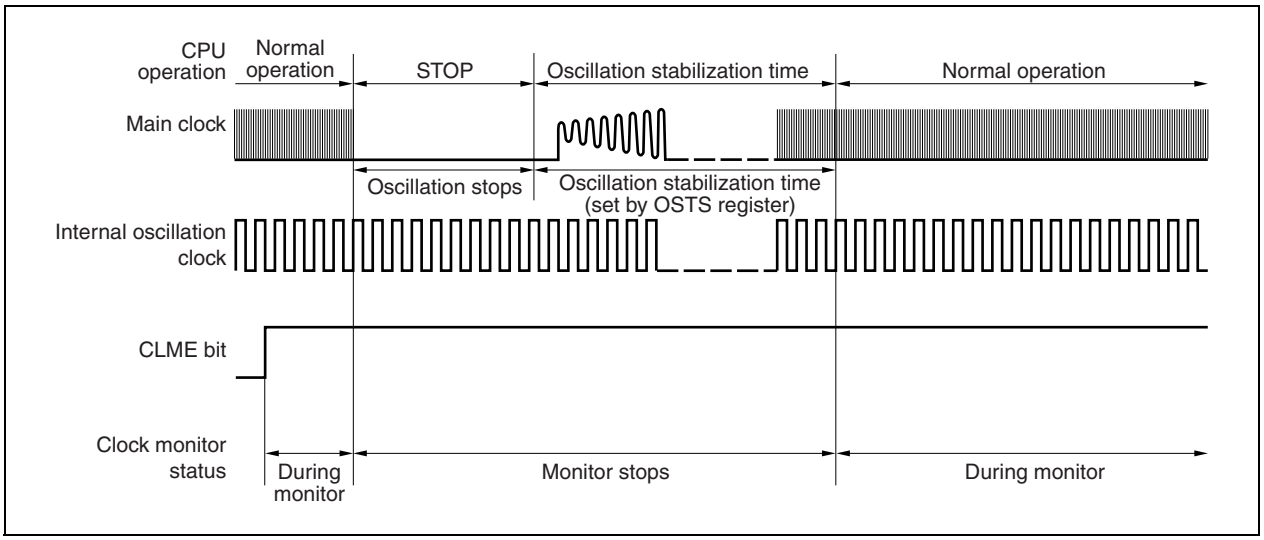
Figure 23-3. Clock Monitor Status After $\overline{\text{RESET}}$ Input
(CLM.CLME bit = 1 is set after $\overline{\text{RESET}}$ input and at the end of main clock oscillation stabilization time)



(3) Operation in STOP mode or after STOP mode is released

If the STOP mode is set with the CLM.CLME bit = 1, the monitor operation is stopped in the STOP mode and while the oscillation stabilization time is being counted. After the oscillation stabilization time, the monitor operation is automatically started.

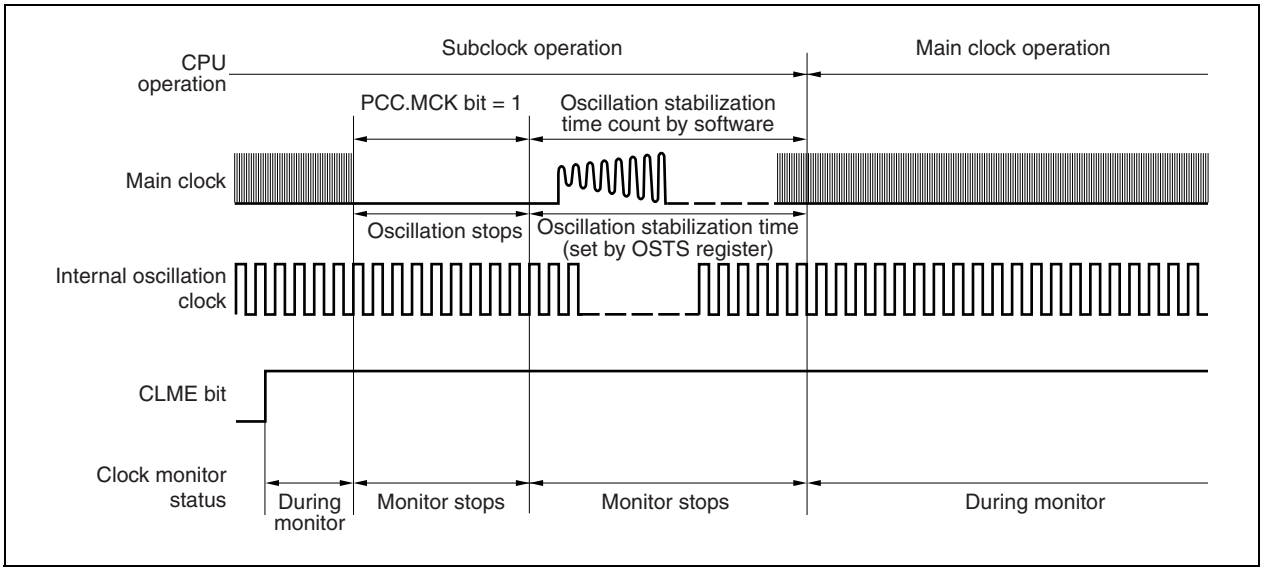
Figure 23-4. Operation in STOP Mode or After STOP Mode Is Released



(4) Operation when main clock is stopped (arbitrary)

During subclock operation (PCC.CLS bit = 1) or when the main clock is stopped by setting the PCC.MCK bit to 1, the monitor operation is stopped until the main clock operation is started (PCC.CLS bit = 0). The monitor operation is automatically started when the main clock operation is started.

Figure 23-5. Operation When Main Clock Is Stopped (Arbitrary)



(5) Operation while CPU is operating on internal oscillation clock (CCLS.CCLSF bit = 1)

The monitor operation is not stopped when the CCLSF bit is 1, even if the CLME bit is set to 1.

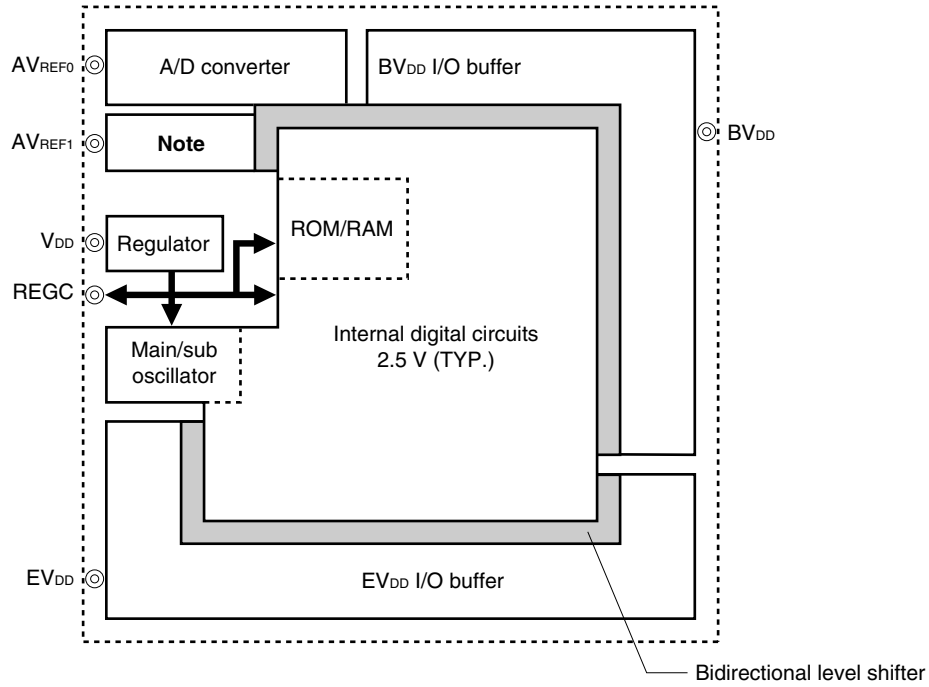
CHAPTER 24 REGULATOR

24.1 Outline

The V850ES/SG1 includes a regulator to reduce power consumption and noise.

<R> This regulator supplies a stepped-down V_{DD} power supply voltage to the oscillator block and internal logic circuits (except the A/D converter, D/A converter (μPD703253Y only), and output buffers). The regulator output voltage is set to 2.5 V (TYP.).

<R> **Figure 24-1. Regulator**



Note μPD703249Y, 703252Y: P1 I/O buffer
μPD703253Y: D/A converter

Caution Use the regulator with a setting of $V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1} \geq BV_{DD}$.

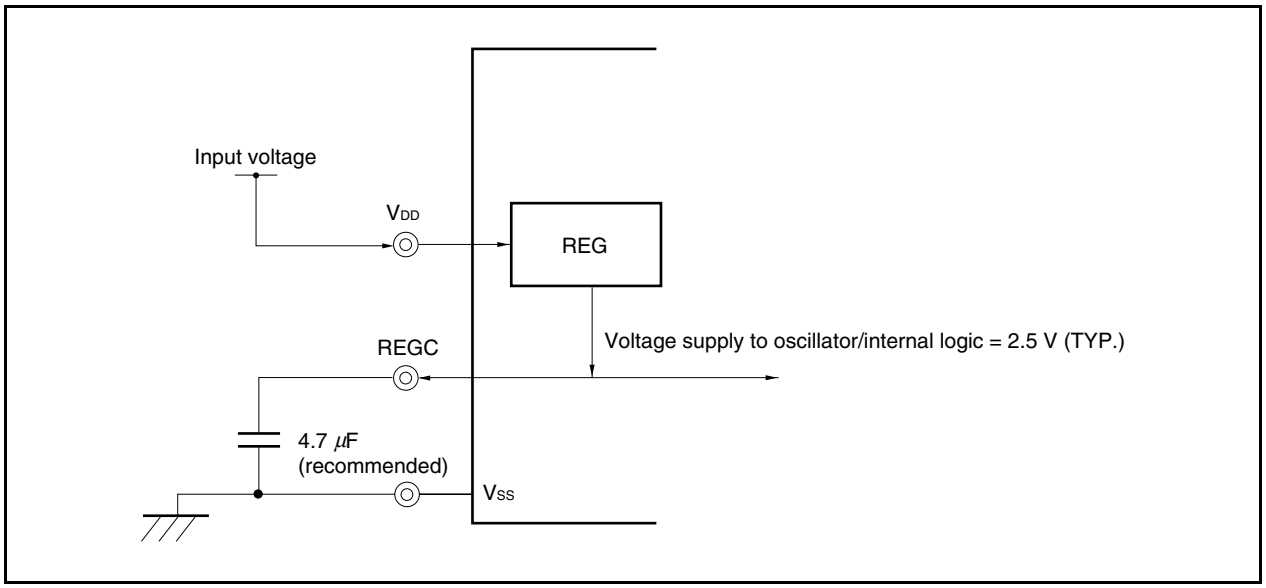
24.2 Operation

The regulator of this product always operates in any mode (normal operation mode, HALT mode, IDLE1 mode, IDLE2 mode, STOP mode, or during reset).

Be sure to connect a capacitor (4.7 μF (recommended value)) to the REGC pin to stabilize the regulator output.

A diagram of the regulator pin connection method is shown below.

Figure 24-2. REGC Pin Connection



CHAPTER 25 ROM CORRECTION FUNCTION

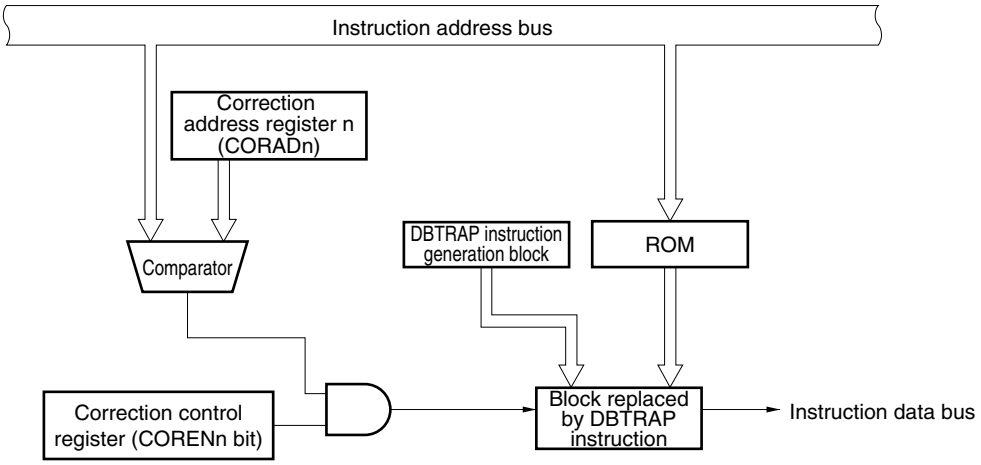
25.1 Overview

The ROM correction function is used to replace part of the program in the internal ROM with the program of an external memory or the internal RAM.

By using this function, program bugs found in the internal ROM can be corrected.

Up to four addresses can be specified for correction.

Figure 25-1. Block Diagram of ROM Correction



Remark n = 0 to 3

25.2 Registers

(1) Correction address registers 0 to 3 (CORAD0 to CORAD3)

The CORAD0 to CORAD3 registers set the first address of the program to be corrected in the ROM.

The program can be corrected at up to four places because four CORADn registers are provided (n = 0 to 3).

The CORADn register can be read or written in 32-bit units.

If the higher 16 bits of the CORADn register are used as the CORADnH register, and the lower 16 bits as the CORADnL register, these registers can be read or written in 16-bit units.

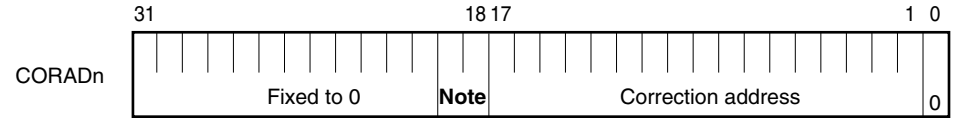
Reset input clears these registers to 00000000H.

Because the ROM capacity differs from one product to another, set the correction addresses in the following ranges.

- μPD703249Y, 703252Y (256 KB): 0000000H to 003FFFFH
- <R> μPD703253Y (128 KB): 0000000H to 001FFFFH

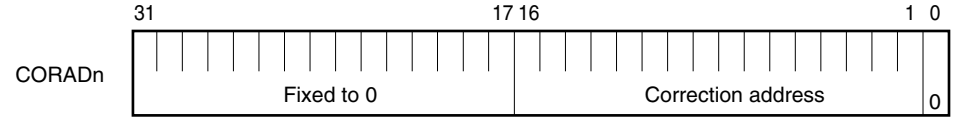
After reset: 00000000H R/W Address: CORAD0 FFFFF840H,
CORAD0L FFFFF840H, CORAD0H FFFFF842H,
CORAD1 FFFFF844H,
CORAD1L FFFFF844H, CORAD1H FFFFF846H,
CORAD2 FFFFF848H,
CORAD2L FFFFF848H, CORAD2H FFFFF84AH,
CORAD3 FFFFF84CH,
CORAD3L FFFFF84CH, CORAD3H FFFFF84EH

(a) 256 KB



(n = 0 to 3)

<R> (b) 128 KB



(n = 0 to 3)

Note Cleared to 0.

(2) Correction control register (CORCN)

The CORCN register disables or enables the correction operation at the addresses set in the CORADn register (n = 0 to 3).

Each channel can be enabled or disabled by this register.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

After reset: 00H R/W Address: FFFFF880H

	7	6	5	4	3	2	1	0
CORCN	0	0	0	0	COREN3	COREN2	COREN1	COREN0

CORENn	Enables/disables correction operation
0	Disabled
1	Enabled

Remark n = 0 to 3

Table 25-1. Correspondence Between CORCN Register Bits and CORADn Registers

CORCN Register Bit	Corresponding CORADn Register
COREN3	CORAD3
COREN2	CORAD2
COREN1	CORAD1
COREN0	CORAD0

25.3 ROM Correction Operation and Program Flow

<1> If the address to be corrected and the fetch address of the internal ROM match, the fetch code is replaced by the DBTRAP instruction.

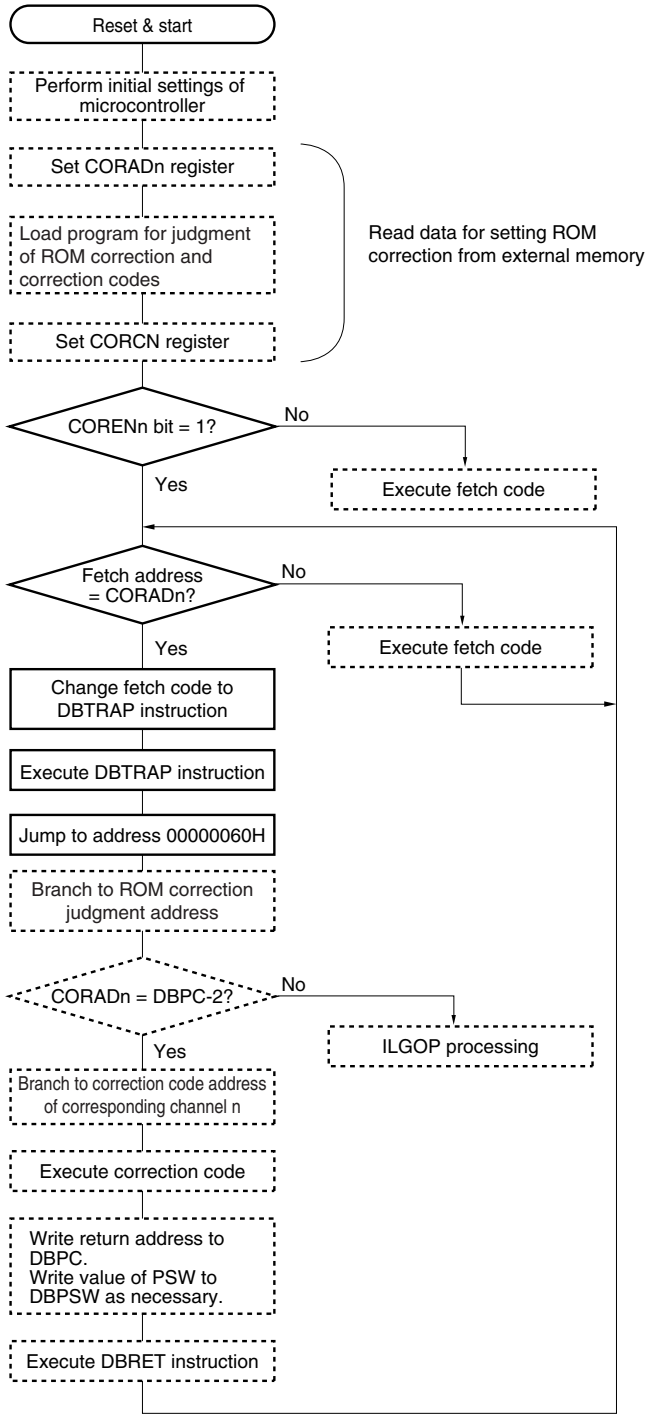
<2> When the DBTRAP instruction is executed, execution branches to address 00000060H.

<3> Software processing after branching causes the result of ROM correction to be judged (the fetch address and ROM correction operation are confirmed) and execution to branch to the correction software.

<4> After the correction software has been executed, the return address is set, and return processing is started by the DBRET instruction.

Caution The software that performs <3> and <4> must be executed in the internal ROM/RAM.

Figure 25-2. ROM Correction Operation and Program Flow



Remarks 1. : Processing by user program (software)

: Processing by ROM correction (hardware)

2. n = 0 to 3

25.4 Cautions

(1) When setting an address to be corrected in the CORADn register, clear the higher bits to 0 in accordance with the capacity of the internal ROM.

(2) The ROM correction function cannot be used to correct data in the internal ROM. It can only be used to correct instruction codes. If ROM correction is used to correct data, that data is replaced with a DBTRAP instruction code.

(3) ROM correction is not performed in regards to the ROM code before writing in the CORCNn register ends.

(4) After executing a DBTRAP instruction, the PSW.NP, EP, and DI bits are set to 111, and interrupt/exception cannot be acknowledged. After executing a DBTRAP instruction, change the PSW register value as required.

(5) The DBPC and DBPSW registers can be accessed while DBTRAP instructions are being executed.

(6) If the addresses of the instructions executed immediately after the CORCNn register setting (enabled) are set as the correction addresses, normal operation may not be obtained (DBTRAP is not generated).

CHAPTER 26 ELECTRICAL SPECIFICATIONS

26.1 Absolute Maximum Ratings

(T_A = 25°C)

Parameter	Symbol	Conditions	Ratings	Unit
Supply voltage	V_{DD}	$V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$	−0.5 to +4.6	V
	BV_{DD}		−0.5 to +4.6	V
	EV_{DD}	$V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$	−0.5 to +4.6	V
	AV_{REF0}	$V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$	−0.5 to +4.6	V
	AV_{REF1}	$V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$	−0.5 to +4.6	V
	V_{SS}	$V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$	−0.5 to +0.5	V
	AV_{SS}	$V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$	−0.5 to +0.5	V
	BV_{SS}	$V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$	−0.5 to +0.5	V
	EV_{SS}	$V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$	−0.5 to +0.5	V
Input voltage	V_{I1}	$\overline{\text{RESET}}$, PDH4, PDH5	−0.5 to EV_{DD} + 0.5[Note 1]	V
	V_{I2}	PCM0 to PCM3, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH3, PDL0 to PDL15	−0.5 to BV_{DD} + 0.5[Note 1]	V
	V_{I3}	P10, P11	−0.5 to AV_{REF1} + 0.5[Note 1]	V
	V_{I4}	X1, X2, XT1, XT2	−0.5 to V_{RO}[Note 2] + 0.5[Note 1]	V
	V_{I5}	P02 to P06, P30 to P39, P40 to P42, P50 to P55, P90 to P915	−0.5 to +6.0	V
Analog input voltage	V_{IAN}	P70 to P711	−0.5 to AV_{REF0} + 0.5[Note 1]	V

Notes 1. Be sure not to exceed the absolute maximum ratings (MAX. value) of each supply voltage.

2. On-chip regulator output voltage (2.5 V (TYP.))

(T_A = 25°C)

Parameter	Symbol	Conditions		Ratings	Unit
Output current, low	I_{OL}	P02 to P06, P30 to P39, P40 to P42, P50 to P55, P90 to P915, PDH4, PDH5	Per pin	4	mA
			Total of all pins	50	mA
		PCM0 to PCM3, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH3, PDL0 to PDL15	Per pin	4	mA
			Total of all pins	50	mA
		P10, P11	Per pin	4	mA
			Total of all pins	8	mA
		P70 to P711	Per pin	4	mA
			Total of all pins	20	mA
Output current, high	I_{OH}	P02 to P06, P30 to P39, P40 to P42, P50 to P55, P90 to P915, PDH4, PDH5	Per pin	−4	mA
			Total of all pins	−50	mA
		PCM0 to PCM3, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH3, PDL0 to PDL15	Per pin	−4	mA
			Total of all pins	−50	mA
		P10, P11	Per pin	−4	mA
			Total of all pins	−8	mA
		P70 to P711	Per pin	−4	mA
			Total of all pins	−20	mA
Operating ambient temperature	T_A			−40 to +85	°C
Storage temperature	T_{stg}			−65 to +150	°C

Cautions 1. Do not directly connect the output (or I/O) pins of IC products to each other, or to V_{DD}, V_{CC}, and GND. Open-drain pins or open-collector pins, however, can be directly connected to each other.

Direct connection of the output pins between an IC product and an external circuit is possible, if the output pins can be set to the high-impedance state and the output timing of the external circuit is designed to avoid output conflict.

2. 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. The ratings and conditions indicated for DC characteristics and AC characteristics represent the quality assurance range during normal operation.

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

26.2 Capacitance

<R> **(T_A = 25°C, V_{DD} = EV_{DD} = BV_{DD} = AV_{REF0} = AV_{REF1}, V_{SS} = EV_{SS} = BV_{SS} = AV_{SS} = 0 V)**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
I/O capacitance	C_{IO}	f_X = 1 MHz Unmeasured pins returned to 0 V			10	pF

26.3 Operating Conditions

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD}$ = EV_{DD} = AV_{REF0} = AV_{REF1}, V_{SS} = EV_{SS} = BV_{SS} = AV_{SS} = 0 V)

Internal System Clock Frequency	Conditions	Supply Voltage				Unit
		V_{DD}	EV_{DD}	BV_{DD}	AV_{REF0}, AV_{REF1}	
f_{XX} = 2.5 to 20 MHz	C = 4.7 μF, **Note 1**	2.85 to 3.6	2.85 to 3.6	2.7 to 3.6	2.85 to 3.6	V
<R>	C = 4.7 μF, **Note 2**	3.0 to 3.6	3.0 to 3.6	2.7 to 3.6	3.0 to 3.6	V
<R> f_{XT} = 32.768 kHz	C = 4.7 μF, **Note 1**	2.85 to 3.6	2.85 to 3.6	2.7 to 3.6	2.85 to 3.6	V

Notes 1. μPD703249Y, 703252Y: A/D converter stopped

<R> μPD703253Y: A/D converter stopped, D/A converter stopped

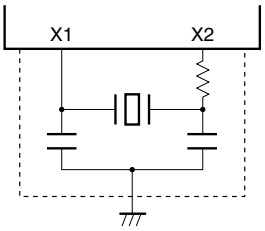
2. μPD703249Y, 703252Y: A/D converter operation

<R> μPD703253Y: A/D converter operation, D/A converter operation

26.4 Oscillator Characteristics

26.4.1 Main clock oscillator characteristics

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Resonator	Circuit Example	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Ceramic resonator/ Crystal resonator		Oscillation frequency (f_X)[Note 1]		2.5		10	MHz
		Oscillation stabilization time[Note 2]	After reset is released		$2^{16}/f_X$		s
			After STOP mode is released	1[Note 4]	**Note 3**		ms
			After IDLE2 mode is released	350[Note 4]	**Note 3**		μs

Notes 1. The oscillation frequency shown above indicates only oscillator characteristics. Use the μPD703253Y so that the internal operation conditions do not exceed the ratings shown in **AC Characteristics** and **DC Characteristics**.

2. Time required from start of oscillation until the resonator stabilizes.

3. The value varies depending on the setting of the OSTS register.

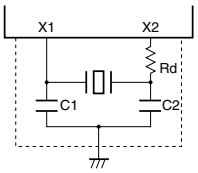
4. Time required to set up PLL. Secure the setup time using the OSTS register.

Cautions 1. When using the main clock oscillator, wire as follows in the area enclosed by the broken lines in the above figure 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. When the main clock is stopped and the device is operating on the subclock, wait until the oscillation stabilization time has been secured by the program before switching back to the main clock.

<R> **(i) KYOCERA KINSEKI CORPORATION: Crystal resonator (T_A = –40 to +85°C)**

Manufacturer (Part Number)	Circuit Example	Oscillation Frequency f_X (kHz)	Recommended Circuit Constant			Oscillation Voltage Range	
			C1 (pF)	C2 (pF)	Rd (kΩ)	MIN. (V)	MAX. (V)
KYOCERA KINSEKI CORPORATION - CX-5FD (capacitance : 8 pF) - CX-49G (capacitance : 8 pF) - HC-49/U-S (capacitance : 8 pF) About other resonator's type name, refer to the resonator manufacturer.		4,000	8	8	–	2.85	3.6
		5,000	8	8	–	2.85	3.6
		8,000	8	8	–	2.85	3.6
		10,000	8	8	–	2.85	3.6
		3,145.72	8	8	–	2.85	3.6
		4,718.592	8	8	–	2.85	3.6
		6,291.456	8	8	–	2.85	3.6

Caution This oscillator constant is a reference value based on evaluation under a specific environment by the resonator manufacturer.

If optimization of oscillator characteristics is necessary in the actual application, apply to the resonator manufacturer for evaluation on the implementation circuit.

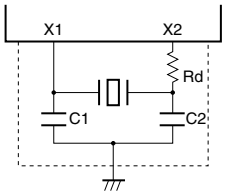
The oscillation voltage and oscillation frequency indicate only oscillator characteristics. Use the V850ES/SG1 so that the internal operating conditions are within the specifications of the DC and AC characteristics.

Remark Contact :

KYOCERA Electronic components & devices http://global.kyocera.com/prdct/electro/index.html

Resonator vs. IC matching search http://www3.kyocera.co.jp/electro/app/en/searchTopShow.do

<R> **(ii) Murata Mfg. Co. Ltd.: Ceramic resonator (T_A = –40 to +85°C)**

Manufacturer	Circuit Example	Oscillation Frequency f_X (MHz)	Part Number	Recommended Circuit Constant			Oscillation Voltage Range	
				C1 (pF)	C2 (pF)	Rd (kΩ)	MIN. (V)	MAX. (V)
Murata Mfg. Co. Ltd.		4.000	CSTCR4M00G55B-R0	on-chip (39)	on-chip (39)	0	2.85	3.6
			CSTCR4M00G15C**-R0	on-chip (39)	on-chip (39)	0	2.85	3.6
		5.000	CSTCR5M00G55B-R0	on-chip (39)	on-chip (39)	0	2.85	3.6
			CSTCR5M00G15C**-R0	on-chip (39)	on-chip (39)	0	2.85	3.6
		6.000	CSTCR6M00G55B-R0	on-chip (39)	on-chip (39)	0	2.85	3.6
			CSTCR6M00G15C**-R0	on-chip (39)	on-chip (39)	0	2.85	3.6
		8.000	CSTCE8M00G55A-R0	on-chip (33)	on-chip (33)	0	2.85	3.6
			CSTCE8M00G15C**-R0	on-chip (33)	on-chip (33)	0	2.85	3.6
		10.000	CSTCE10M0G55A-R0	on-chip (33)	on-chip (33)	0	2.85	3.6
			CSTCE10M0G15C**-R0	on-chip (33)	on-chip (33)	0	2.85	3.6

Caution This oscillator constant is a reference value based on evaluation under a specific environment by the resonator manufacturer.

If optimization of oscillator characteristics is necessary in the actual application, apply to the resonator manufacturer for evaluation on the implementation circuit.

The oscillation voltage and oscillation frequency indicate only oscillator characteristics. Use the V850ES/SG1 so that the internal operating conditions are within the specifications of the DC and AC characteristics.

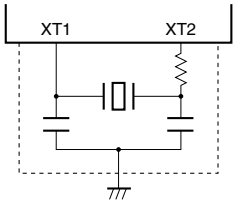
Remarks 1. The total tolerance of a product having "**" in its part number can be adjusted up to ±3000 ppm.

2. Contact:

Product Engineering Service Section
Piezoelectric Components Department I
Piezoelectric Components Division
Device Business Unit
Murata Mfg. Co., Ltd.
TEL: +81-75-955-6915
E-mail: piezo@murata.co.jp
IC Part Number -> Ceramic Resonator Search: http://search.murata.co.jp/Ceramy/CeMenu_en.do

26.4.2 Subclock oscillator characteristics

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Resonator	Circuit Example	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Crystal resonator		Oscillation frequency (f_{XT})[Note 1]		32	32.768	35	kHz
		Oscillation stabilization time[Note 2]				10	s

Notes 1. The oscillation frequency shown above indicates only oscillator characteristics. Use the μPD703253Y so that the internal operation conditions do not exceed the ratings shown in **AC Characteristics** and **DC Characteristics**.

2. Time required from when V_{DD} reaches the following oscillation voltage range to when the crystal resonator stabilizes.

Cautions 1. When using the subclock 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. The subclock oscillator is designed as a low-amplitude circuit for reducing power consumption, and is more prone to malfunction due to noise than the main clock oscillator.
Particular care is therefore required with the wiring method when the subclock is used.

3. For the resonator selection and oscillator constant, customers are requested to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.

26.4.3 PLL characteristics

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input frequency	f_X	×4 mode	2.5		5	MHz
		×8 mode	2.5		2.5	MHz
Output frequency	f_{XX}	×4 mode	10		20	MHz
		×8 mode	20		20	MHz
Lock time	t_{PLL}	After V_{DD} reaches 2.85 V (MIN.)			800	μs

26.4.4 Internal oscillator characteristics

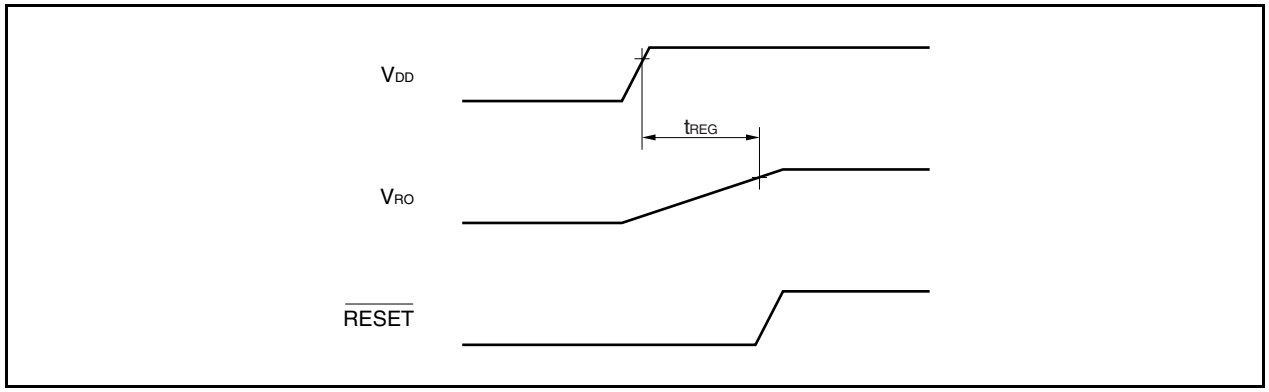
(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Output frequency	f_R		100	200	400	kHz

26.5 Regulator Characteristics

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input voltage	V_{DD}	f_{XX} = 20 MHz (MAX.)	2.85		3.6	V
Output voltage	V_{RO}			2.5		V
Regulator output stabilization time	t_{REG}	After V_{DD} reaches 2.85 V (MIN.), Stabilization capacitance C = 4.7 μF connected to REGC pin			1	ms



26.6 DC Characteristics

26.6.1 I/O level

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input voltage, high	V_{IH1}	PDH4, PDH5	0.7EV_{DD}		EV_{DD}	V
	V_{IH2}	$\overline{\text{RESET}}$	0.8EV_{DD}		EV_{DD}	V
	V_{IH3}	P02 to P06, P30 to P37, P42, P50 to P55, P92 to P915	0.8EV_{DD}		5.5	V
	V_{IH4}	P38, P39, P40, P41, P90, P91	0.7EV_{DD}		5.5	V
	V_{IH5}	PCM0 to PCM3, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH3, PDL0 to PDL15	0.7BV_{DD}		BV_{DD}	V
	V_{IH6}	P70 to P711	0.7AV_{REF0}		AV_{REF0}	V
	V_{IH7}	P10, P11	0.7AV_{REF1}		AV_{REF1}	V
Input voltage, low	V_{IL1}	PDH4, PDH5	EV_{SS}		0.3EV_{DD}	V
	V_{IL2}	$\overline{\text{RESET}}$	EV_{SS}		0.2EV_{DD}	V
	V_{IL3}	P02 to P06, P30 to P37, P42, P50 to P55, P92 to P915	EV_{SS}		0.2EV_{DD}	V
	V_{IL4}	P38, P39, P40, P41, P90, P91	EV_{SS}		0.3EV_{DD}	V
	V_{IL5}	PCM0 to PCM3, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH3, PDL0 to PDL15	BV_{SS}		0.3BV_{DD}	V
	V_{IL6}	P70 to P711	AV_{SS}		0.3AV_{REF0}	V
	V_{IL7}	P10, P11	AV_{SS}		0.3AV_{REF1}	V
Input leakage current, high	I_{LIH}	$V_I = V_{DD} = EV_{DD} = BV_{DD} = AV_{REF0} = AV_{REF1}$			5	μA
Input leakage current, low	I_{LIL}	V_I = 0 V			–5	μA
Output leakage current, high	I_{LOH}	$V_O = V_{DD} = EV_{DD} = BV_{DD} = AV_{REF0} = AV_{REF1}$			5	μA
Output leakage current, low	I_{LOL}	V_O = 0 V			–5	μA

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol	Conditions			MIN.	TYP.	MAX.	Unit
Output voltage, high	V_{OH1}	P02 to P06, P30 to P39, P40 to P42, P50 to P55, P90 to P915, PDH4, PDH5	Per pin I_{OH} = –1.0 mA	Total of all pins –20 mA	EV_{DD} – 1.0		EV_{DD}	V
			Per pin I_{OH} = –100 μA	Total of all pins –4.2 mA	EV_{DD} – 0.5		EV_{DD}	V
	V_{OH2}	PCM0 to PCM3, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH3, PDL0 to PDL15	Per pin I_{OH} = –1.0 mA	Total of all pins –20 mA	BV_{DD} – 1.0		BV_{DD}	V
			Per pin I_{OH} = –100 μA	Total of all pins –2.8 mA	BV_{DD} – 0.5		BV_{DD}	V
	V_{OH3}	P70 to P711	Per pin I_{OH} = –0.4 mA	Total of all pins –4.8 mA	AV_{REF0} – 1.0		AV_{REF0}	V
			Per pin I_{OH} = –100 μA	Total of all pins –1.2 mA	AV_{REF0} – 0.5		AV_{REF0}	V
	V_{OH4}	P10, P11	Per pin I_{OH} = –0.4 mA	Total of all pins –0.8 mA	AV_{REF1} – 1.0		AV_{REF1}	V
			Per pin I_{OH} = –100 μA	Total of all pins –0.2 mA	AV_{REF1} – 0.5		AV_{REF1}	V
Output voltage, low	V_{OL1}	P02 to P06, P30 to P37, P42, P50 to P55, P92 to P915, PDH4, PDH5	Per pin I_{OL} = 1.0 mA	Total of all pins 20 mA	0		0.4	V
	V_{OL2}	P38, P39, P40, P41, P90, P91	Per pin I_{OL} = 3.0 mA		0		0.4	V
	V_{OL3}	PCM0 to PCM3, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH3, PDL0 to PDL15	Per pin I_{OL} = 1.0 mA	Total of all pins 20 mA	0		0.4	V
	V_{OL4}	P10, P11, P70 to P711	Per pin I_{OL} = 0.4 mA	Total of all pins 5.6 mA	0		0.4	V
Software pull-down resistor	R_1	P05	$V_I = V_{DD}$		10	30	100	kΩ

Remarks 1. Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

2. When the I_{OH} and I_{OL} conditions are not satisfied for a pin but the total value of all pins is satisfied, only that pin does not satisfy the DC characteristics.

26.6.2 Supply current

(1) μPD703249Y, 703252Y

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Supply current[Note 1] (mask ROM version)	I_{DD1}	Normal operation	f_{XX} = 20 MHz (f_X = 5 MHz), peripheral function operating		21	32	mA
	I_{DD2}	HALT mode	f_{XX} = 20 MHz (f_X = 5 MHz), peripheral function operating		15	24	mA
	I_{DD3}	IDLE1 mode	f_{XX} = 5 MHz (f_X = 5 MHz), PLL off		0.3	0.8	mA
	I_{DD4}	IDLE2 mode	f_{XX} = 5 MHz (f_X = 5 MHz), PLL off		0.3	0.8	mA
	I_{DD5}	Subclock operating mode	f_{XT} = 32.768 kHz, main clock, internal oscillator stopped		50	100	μA
	I_{DD6}	Sub-IDLE mode	f_{XT} = 32.768 kHz, main clock, internal oscillator stopped		15	70	μA
	I_{DD7}	STOP mode	Subclock stopped, internal oscillator stopped		6	50	μA
			Subclock operating, internal oscillator stopped		10	60	μA
			Subclock stopped, internal oscillator operating		10	60	μA

Note Total of V_{DD}, EV_{DD}, and BV_{DD} currents. Current flowing through the output buffers, A/D converter, D/A converter, and on-chip pull-down resistor is not included.

<R> **(2) μPD703253Y**

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Supply current[Note]	I_{DD1}	Normal operation	f_{XX} = 20 MHz (f_X = 5 MHz) peripheral function operating		18	28	mA
	I_{DD2}	HALT mode	f_{XX} = 20 MHz (f_X = 5 MHz) peripheral function operating		12	21	mA
	I_{DD3}	IDLE1 mode	f_{XX} = 5 MHz (f_X = 5 MHz), PLL off		0.3	0.8	mA
	I_{DD4}	IDLE2 mode	f_{XX} = 5 MHz (f_X = 5 MHz), PLL off		0.3	0.8	mA
	I_{DD5}	Subclock operating mode	f_{XT} = 32.768 kHz, main clock, internal oscillator stopped		50	100	μA
	I_{DD6}	Sub-IDLE mode	f_{XT} = 32.768 kHz, main clock, internal oscillator stopped		15	70	μA
	I_{DD7}	STOP mode	Subclock stopped, internal oscillator stopped		5	50	μA
			Subclock operating, internal oscillator stopped		9	60	μA
			Subclock stopped, internal oscillator operating		9	60	μA

Note Total of V_{DD}, EV_{DD}, and BV_{DD} currents. Current flowing through the output buffers, A/D converter, D/A converter, and on-chip pull-down resistor is not included.

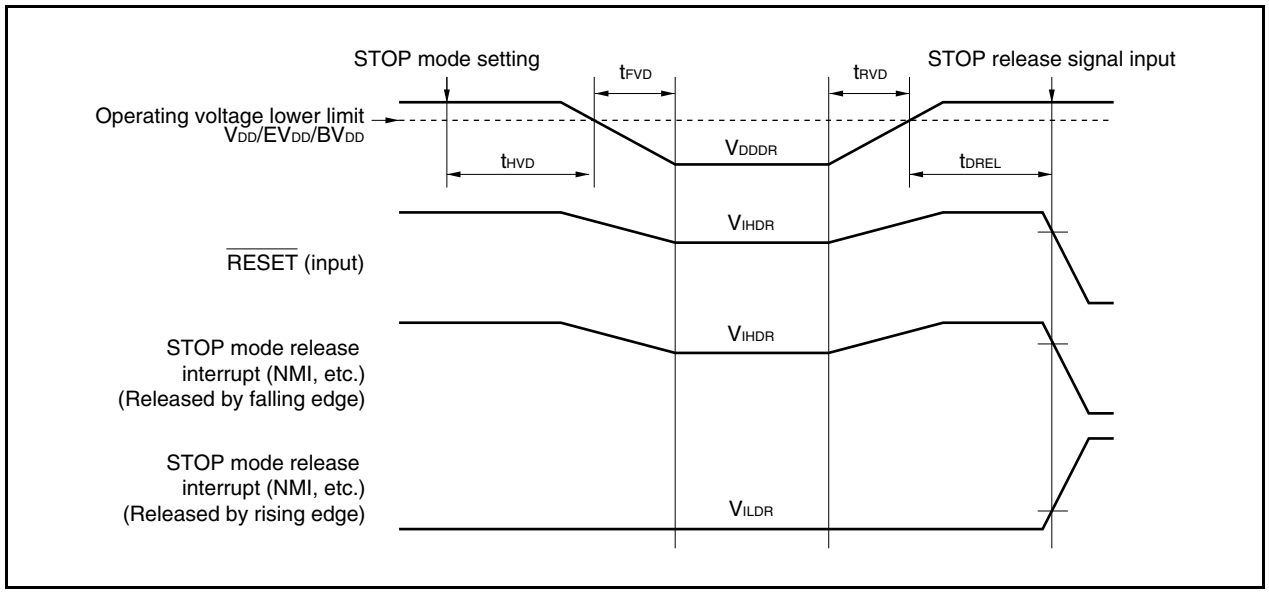
26.7 Data Retention Characteristics

(1) In STOP mode

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention voltage	V_{DDDR}	STOP mode (all functions stopped)	1.9		3.6	V
Data retention current	I_{DDDR}	STOP mode (all functions stopped)		6	50	μA
Supply voltage rise time	t_{RVD}		200			μs
Supply voltage fall time	t_{FVD}		200			μs
Supply voltage retention time	t_{HVD}	After STOP mode setting	0			ms
STOP release signal input time	t_{DREL}	After V_{DD} reaches 2.85 V (MIN.)	0			ms
Data retention input voltage, high	V_{IHDR}	$V_{DD} = EV_{DD} = BV_{DD} = V_{DDDR}$	0.9V_{DDDR}		V_{DDDR}	V
Data retention input voltage, low	V_{ILDR}	$V_{DD} = EV_{DD} = BV_{DD} = V_{DDDR}$	0		0.1V_{DDDR}	V

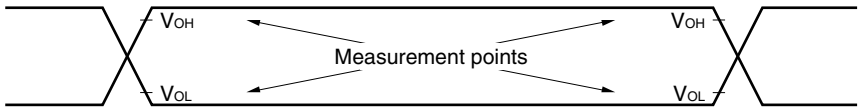
Caution Shifting to STOP mode and restoring from STOP mode must be performed within the rated operating range.



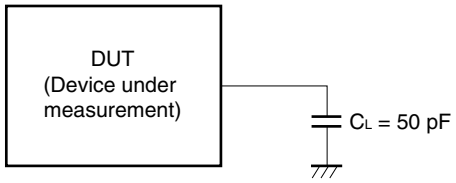
26.8 AC Characteristics

(1) AC test input measurement points (V_{DD}, AV_{REF0}, AV_{REF1}, EV_{DD}, BV_{DD})

(2) AC test output measurement points



(3) Load conditions



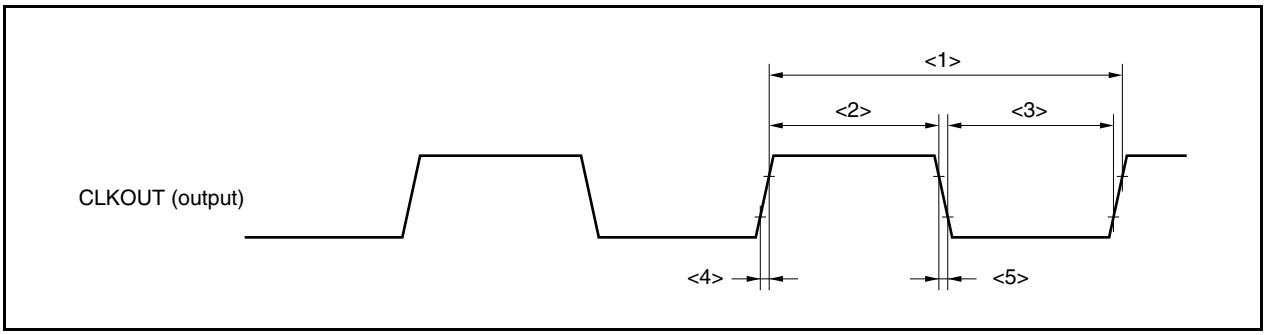
Caution If the load capacitance exceeds 50 pF due to the circuit configuration, bring the load capacitance of the device to 50 pF or less by inserting a buffer or by some other means.

26.8.1 CLKOUT output timing

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
Output cycle	t_{CYK}	<1>		50 ns	31.25 μs	
High-level width	t_{WKH}	<2>		$t_{CYK}/2 - 10$		ns
Low-level width	t_{WKL}	<3>		$t_{CYK}/2 - 10$		ns
Rise time	t_{KR}	<4>			10	ns
Fall time	t_{KF}	<5>			10	ns

Clock Timing



26.8.2 Bus timing

(1) In multiplexed bus mode

(a) Read/write cycle (CLKOUT asynchronous)

(T_A = –40 to +85°C, $BV_{DD} \le V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
Address setup time (to ASTB↓)	t_{SAST}	<6>		$(0.5 + t_{ASW})T - 20$		ns
<R> Address hold time (from ASTB↓)	t_{HSTA}	<7>		$(0.5 + t_{AHW})T - 15$		ns
Delay time from $\overline{RD}$↓ to address float	t_{FRDA}	<8>			16	ns
Data input setup time from address	t_{SAID}	<9>			$(2 + n + t_{ASW} + t_{AHW})T - 35$	ns
<R> Data input setup time from $\overline{RD}$↓	t_{SRID}	<10>			$(1 + n)T - 25$	ns
Delay time from ASTB↓ to $\overline{RD}$, $\overline{WRm}$↓	$t_{DSTRDWR}$	<11>		$(0.5 + t_{AHW})T - 15$		ns
Data input hold time (from $\overline{RD}$↑)	t_{HRDID}	<12>		0		ns
Address output time from $\overline{RD}$↑	t_{DRDA}	<13>		$(1 + i)T - 15$		ns
Delay time from $\overline{RD}$, $\overline{WRm}$↑ to ASTB↑	$t_{DRDWRST}$	<14>		$0.5T - 15$		ns
Delay time from $\overline{RD}$↑ to ASTB↓	t_{DRDST}	<15>		$(1.5 + i + t_{ASW})T - 15$		ns
$\overline{RD}$, $\overline{WRm}$ low-level width	t_{WRDWRL}	<16>		$(1 + n)T - 15$		ns
<R> ASTB high-level width	t_{WSTH}	<17>		$(1 + i + t_{ASW})T - 15$		ns
Data output time from $\overline{WRm}$↓	t_{DWROD}	<18>			15	ns
Data output setup time (to $\overline{WRm}$↑)	t_{SODWR}	<19>		$(1 + n)T - 20$		ns
Data output hold time (from $\overline{WRm}$↑)	t_{HWROD}	<20>		$T - 15$		ns
$\overline{WAIT}$ setup time (to address)	t_{SAWT1}	<21>	$n \ge 1$		$(1.5 + t_{ASW} + t_{AHW})T - 35$	ns
	t_{SAWT2}	<22>			$(1.5 + n + t_{ASW} + t_{AHW})T - 35$	ns
$\overline{WAIT}$ hold time (from address)	t_{HAWT1}	<23>	$n \ge 1$	$(0.5 + n + t_{ASW} + t_{AHW})T$		ns
	t_{HAWT2}	<24>		$(1.5 + n + t_{ASW} + t_{AHW})T$		ns
$\overline{WAIT}$ setup time (to ASTB↓)	t_{SSTWT1}	<25>	$n \ge 1$		$(1 + t_{AHW})T - 25$	ns
	t_{SSTWT2}	<26>			$(1 + n + t_{AHW})T - 25$	ns
$\overline{WAIT}$ hold time (from ASTB↓)	t_{HSTWT1}	<27>	$n \ge 1$	$(n + t_{AHW})T$		ns
	t_{HSTWT2}	<28>		$(1 + n + t_{AHW})T$		ns

Remarks 1. t_{ASW}: Number of address setup wait clocks
t_{AHW}: Number of address hold wait clocks

2. $T = 1/f_{CPU}$ (f_{CPU}: CPU operating clock frequency)

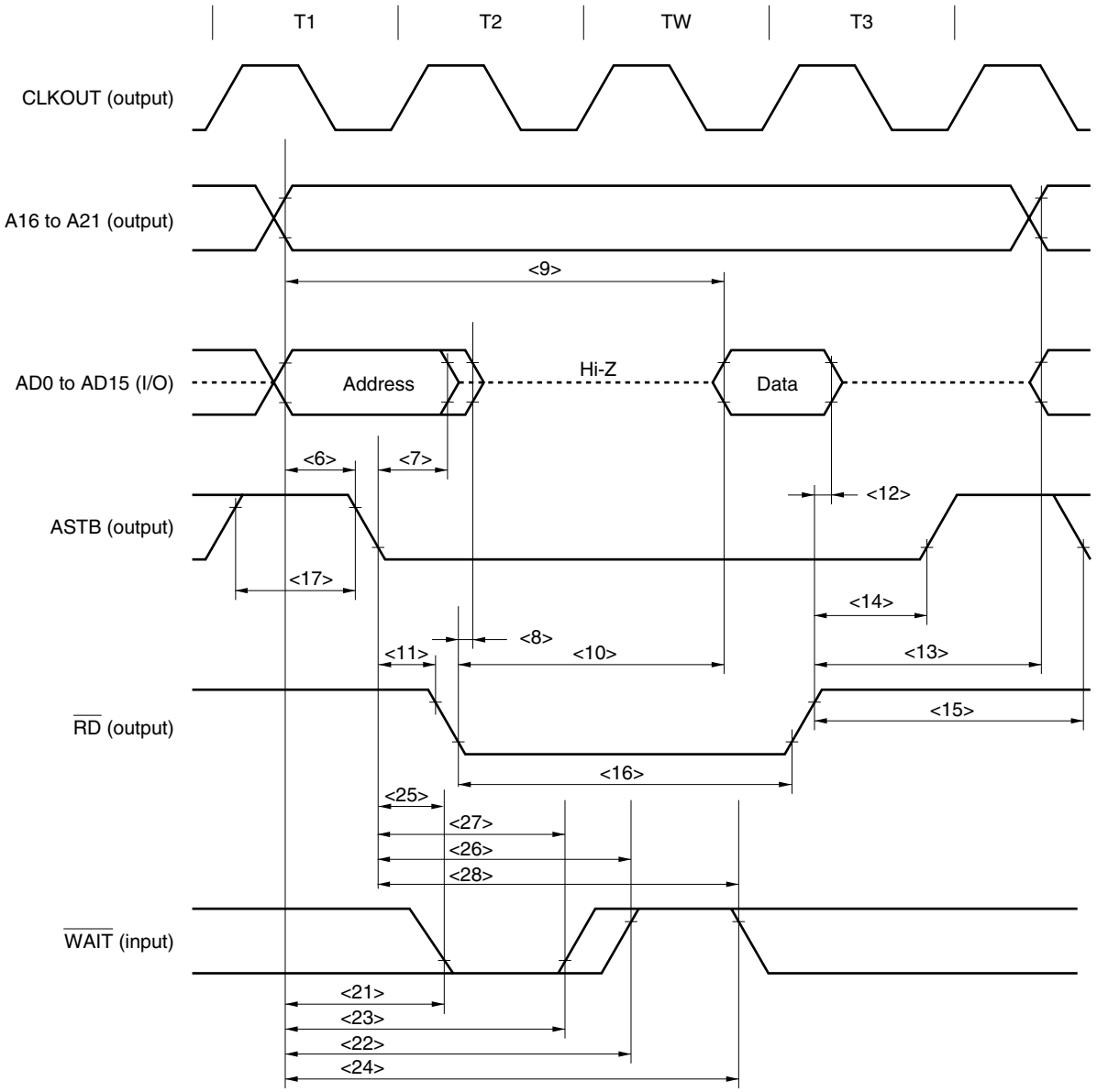
3. n: Number of wait clocks inserted in the bus cycle
The sampling timing changes when a programmable wait is inserted.

4. m = 0, 1

5. i: Number of idle states inserted after a read cycle (0 or 1)

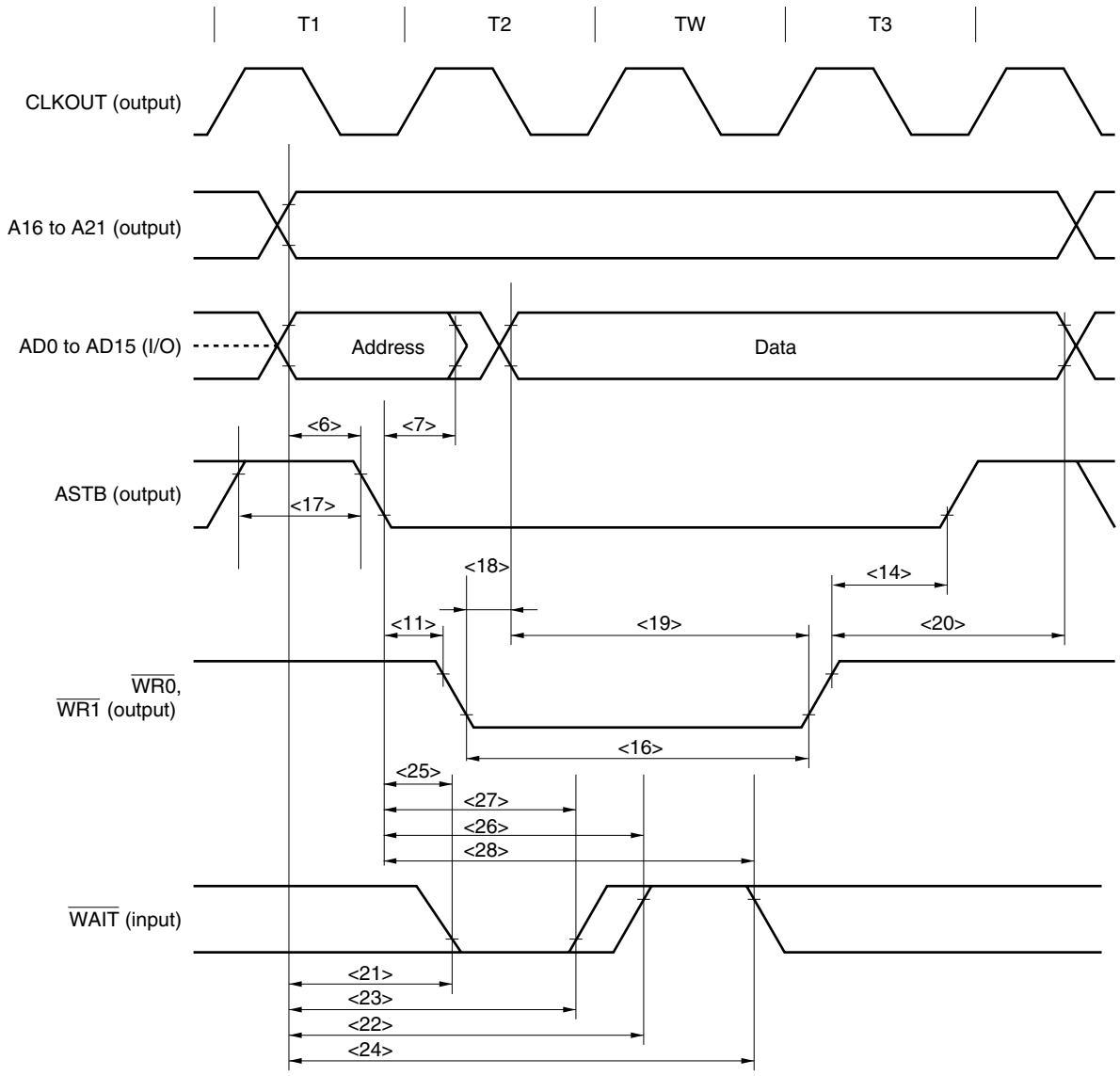
6. The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

Read Cycle (CLKOUT Asynchronous): In Multiplexed Bus Mode



Remark $\overline{WR0}$ and $\overline{WR1}$ are high level.

Write Cycle (CLKOUT Asynchronous): In Multiplexed Bus Mode



Remark $\overline{RD}$ is high level.

(b) Read/write cycle (CLKOUT synchronous): In multiplexed bus mode

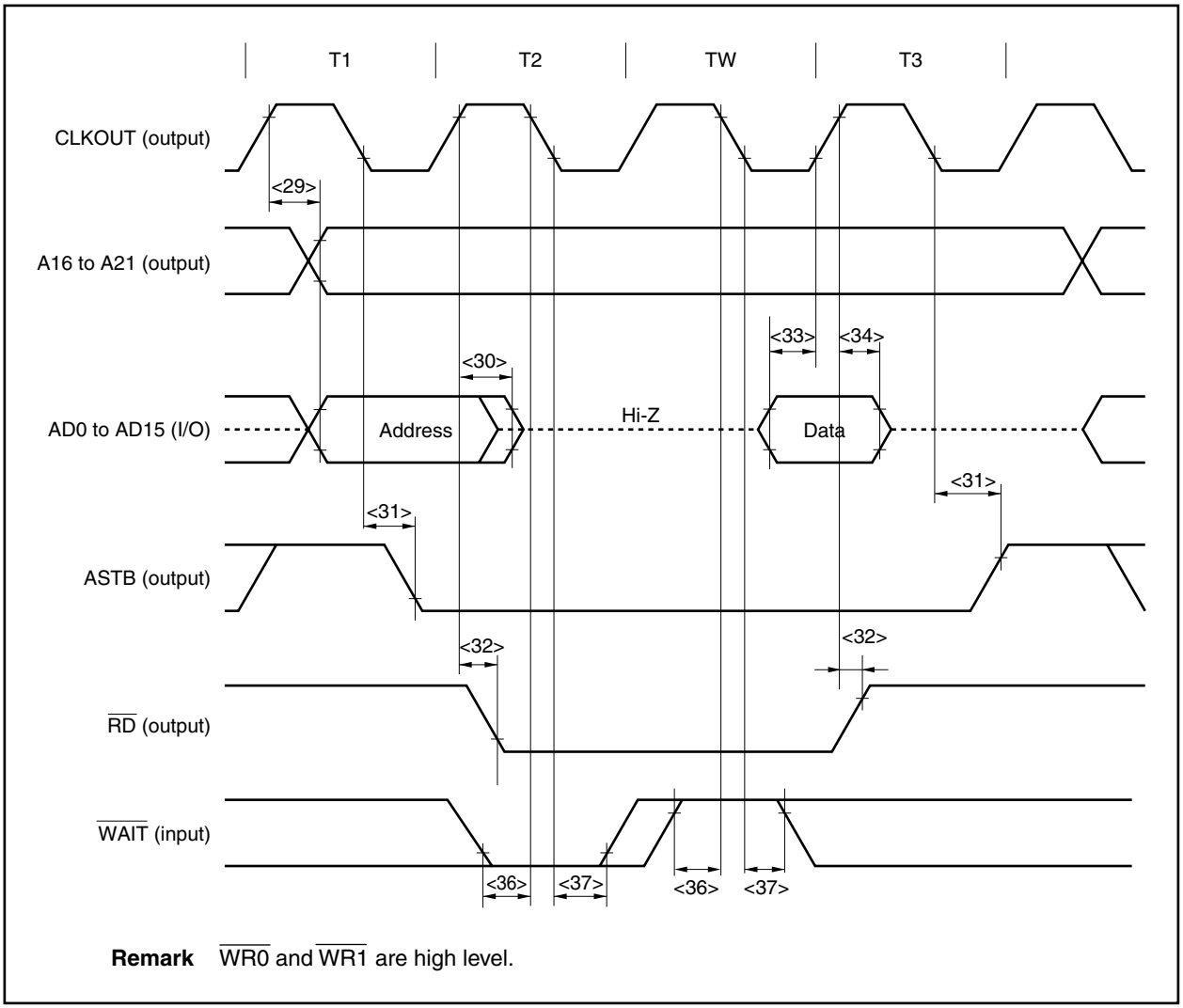
(T_A = –40 to +85°C, $BV_{DD} \le V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
Delay time from CLKOUT↑ to address	t_{DKA}	<29>		0	25	ns
Delay time from CLKOUT↑ to address float	t_{FKA}	<30>		0	19	ns
Delay time from CLKOUT↓ to ASTB	t_{DKST}	<31>		–12	7	ns
Delay time from CLKOUT↑ to $\overline{RD}$, $\overline{WRm}$	t_{DKRDWR}	<32>		–5	14	ns
Data input setup time (to CLKOUT↑)	t_{SIDK}	<33>		15		ns
Data input hold time (from CLKOUT↑)	t_{HKID}	<34>		5		ns
Data output delay time from CLKOUT↑	t_{DKOD}	<35>			19	ns
$\overline{WAIT}$ setup time (to CLKOUT↓)	t_{SWTK}	<36>		20		ns
$\overline{WAIT}$ hold time (from CLKOUT↓)	t_{HKWT}	<37>		5		ns

Remarks 1. m = 0, 1

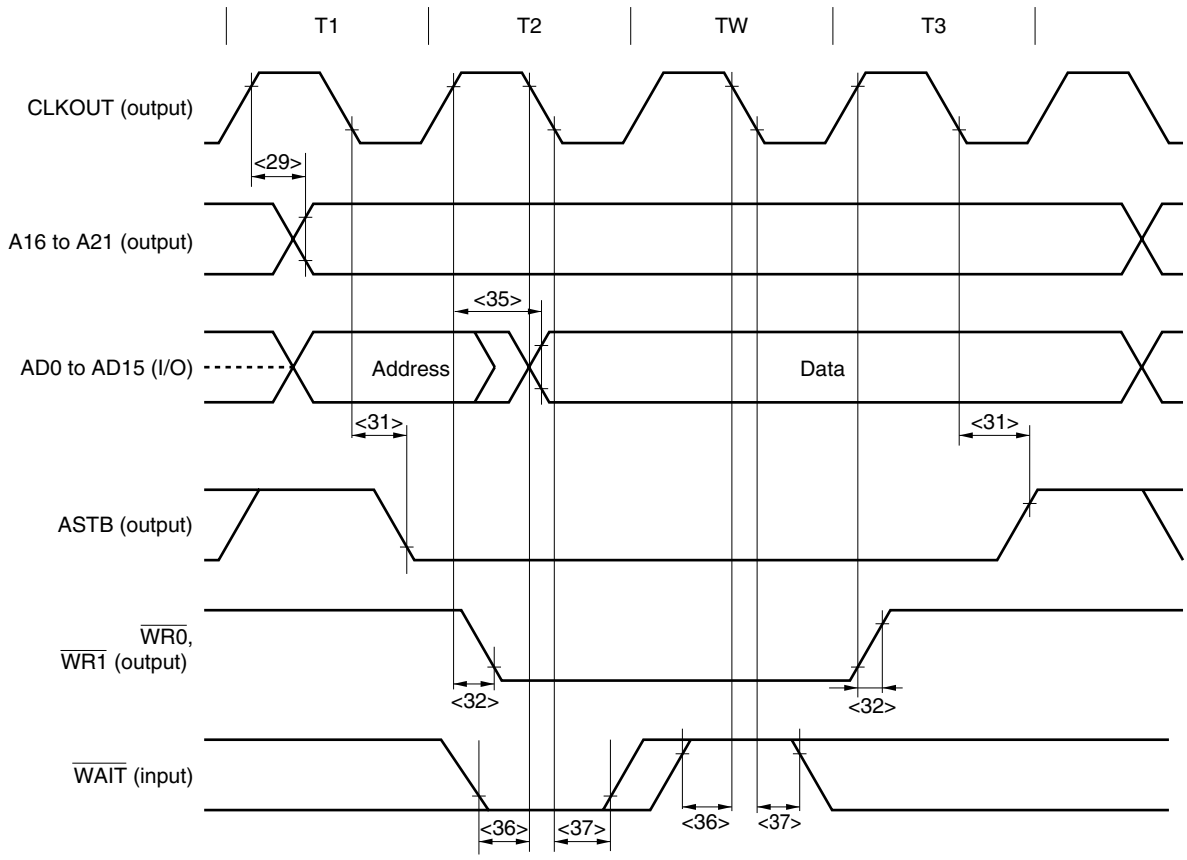
2. The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

Read Cycle (CLKOUT Synchronous): In Multiplexed Bus Mode



Remark $\overline{WR0}$ and $\overline{WR1}$ are high level.

Write Cycle (CLKOUT Synchronous): In Multiplexed Bus Mode



Remark RD is high level.

(2) In separate bus mode

(a) Read cycle (CLKOUT asynchronous): In separate bus mode

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
Address setup time (to $\overline{RD}\downarrow$)	t_{SARD}	<38>		$(0.5 + t_{ASW})T - 23$		ns
Address hold time (from $\overline{RD}\uparrow$)	t_{HARD}	<39>		$iT + 1$		ns
$\overline{RD}$ low-level width	t_{WRDL}	<40>		$(1.5 + n + t_{AHW})T - 10$		ns
Data setup time (to $\overline{RD}\uparrow$)	t_{SISD}	<41>		23		ns
Data hold time (from $\overline{RD}\uparrow$)	t_{HISD}	<42>		0		ns
Data setup time (to address)	t_{SAID}	<43>			$(2 + n + t_{ASW} + t_{AHW})T - 40$	ns
$\overline{WAIT}$ setup time (to $\overline{RD}\downarrow$)	t_{SRDWT1}	<44>			$(0.5 + t_{AHW})T - 25$	ns
	t_{SRDWT2}	<45>			$(0.5 + n + t_{AHW})T - 25$	ns
$\overline{WAIT}$ hold time (from $\overline{RD}\downarrow$)	t_{HRDWT1}	<46>		$(n - 0.5 + t_{AHW})T$		ns
	t_{HRDWT2}	<47>		$(n + 0.5 + t_{AHW})T$		ns
$\overline{WAIT}$ setup time (to address)	t_{SAWT1}	<48>			$(1 + t_{ASW} + t_{AHW})T - 45$	ns
	t_{SAWT2}	<49>			$(1 + n + t_{ASW} + t_{AHW})T - 45$	ns
<R> $\overline{WAIT}$ hold time (from address)	t_{HAWT1}	<50>		$(n + t_{ASW} + t_{AHW})T$		ns
	t_{HAWT2}	<51>		$(1 + n + t_{ASW} + t_{AHW})T$		ns

Remarks 1. t_{ASW}: Number of address setup wait clocks
t_{AHW}: Number of address hold wait clocks

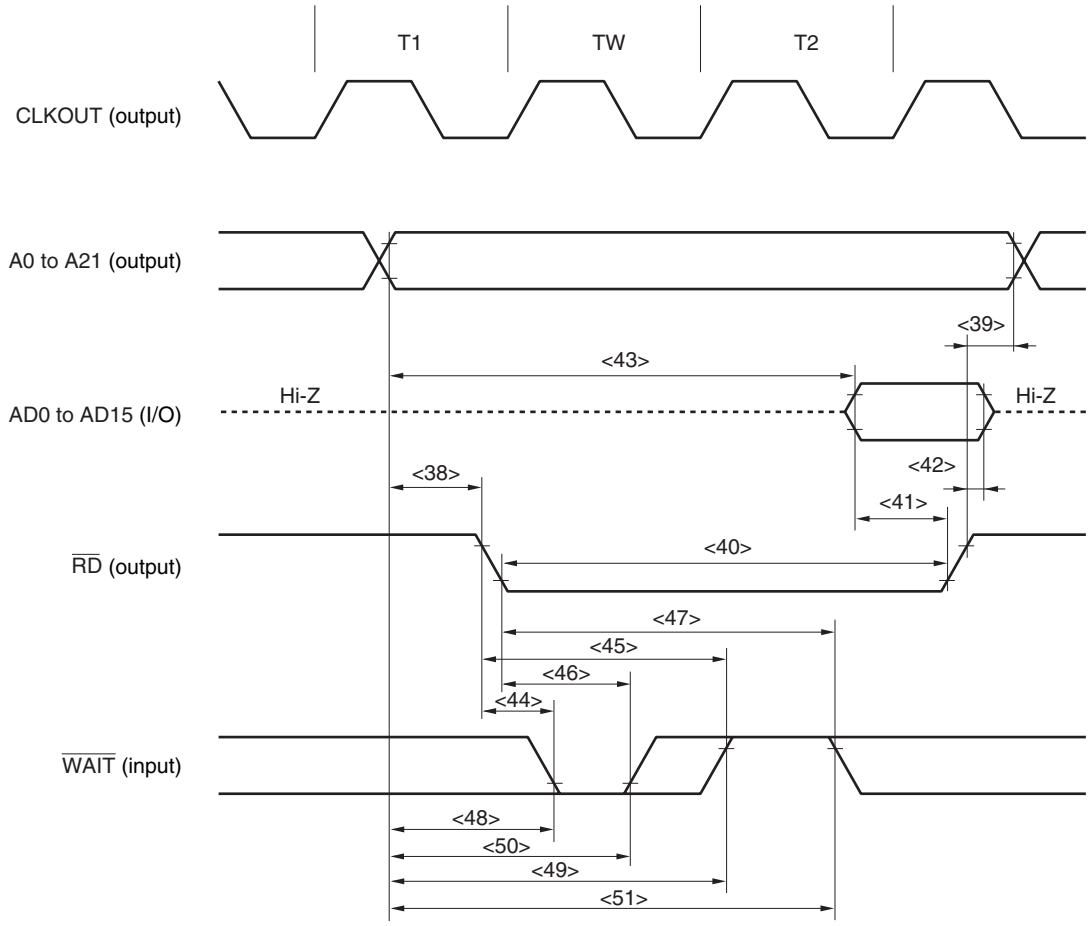
2. $T = 1/f_{CPU}$ (f_{CPU}: CPU operating clock frequency)

3. n: Number of wait clocks inserted in the bus cycle
The sampling timing changes when a programmable wait is inserted

4. i: Number of idle states inserted after a read cycle (0 or 1)

5. The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

Read Cycle (CLKOUT Asynchronous): In Separate Bus Mode



Remark $\overline{WR0}$ and $\overline{WR1}$ are high level.

(b) Write cycle (CLKOUT asynchronous): In separate bus mode

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
Address setup time (to $\overline{WRm}\downarrow$)	t_{SAWR}	<52>		$(1 + t_{ASW} + t_{AHW})T - 23$		ns
Address hold time (from $\overline{WRm}\uparrow$)	t_{HAWR}	<53>		$0.5T - 10$		ns
$\overline{WRm}$ low-level width	t_{WWRL}	<54>		$(0.5 + n)T - 10$		ns
Data output time from $\overline{WRm}\downarrow$	t_{DOSDW}	<55>		–5		ns
Data setup time (to $\overline{WRm}\uparrow$)	t_{SOSDW}	<56>		$(0.5 + n)T - 20$		ns
Data hold time (from $\overline{WRm}\uparrow$)	t_{HOSDW}	<57>		$0.5T - 10$		ns
Data setup time (to address)	t_{SAOD}	<58>		$(1 + t_{ASW} + t_{AHW})T - 25$		ns
$\overline{WAIT}$ setup time (to $\overline{WRm}\downarrow$)	t_{SWRWT1}	<59>		22		ns
	t_{SWRWT2}	<60>			$nT - 22$	ns
$\overline{WAIT}$ hold time (from $\overline{WRm}\downarrow$)	t_{HWRWT1}	<61>		0		ns
	t_{HWRWT2}	<62>		nT		ns
$\overline{WAIT}$ setup time (to address)	t_{SAWT1}	<63>			$(1 + t_{ASW} + t_{AHW})T - 45$	ns
	t_{SAWT2}	<64>			$(1 + n + t_{ASW} + t_{AHW})T - 45$	ns
$\overline{WAIT}$ hold time (from address)	t_{HAWT1}	<65>		$(n + t_{ASW} + t_{AHW})T$		ns
	t_{HAWT2}	<66>		$(1 + n + t_{ASW} + t_{AHW})T$		ns

Remarks 1. m = 0, 1

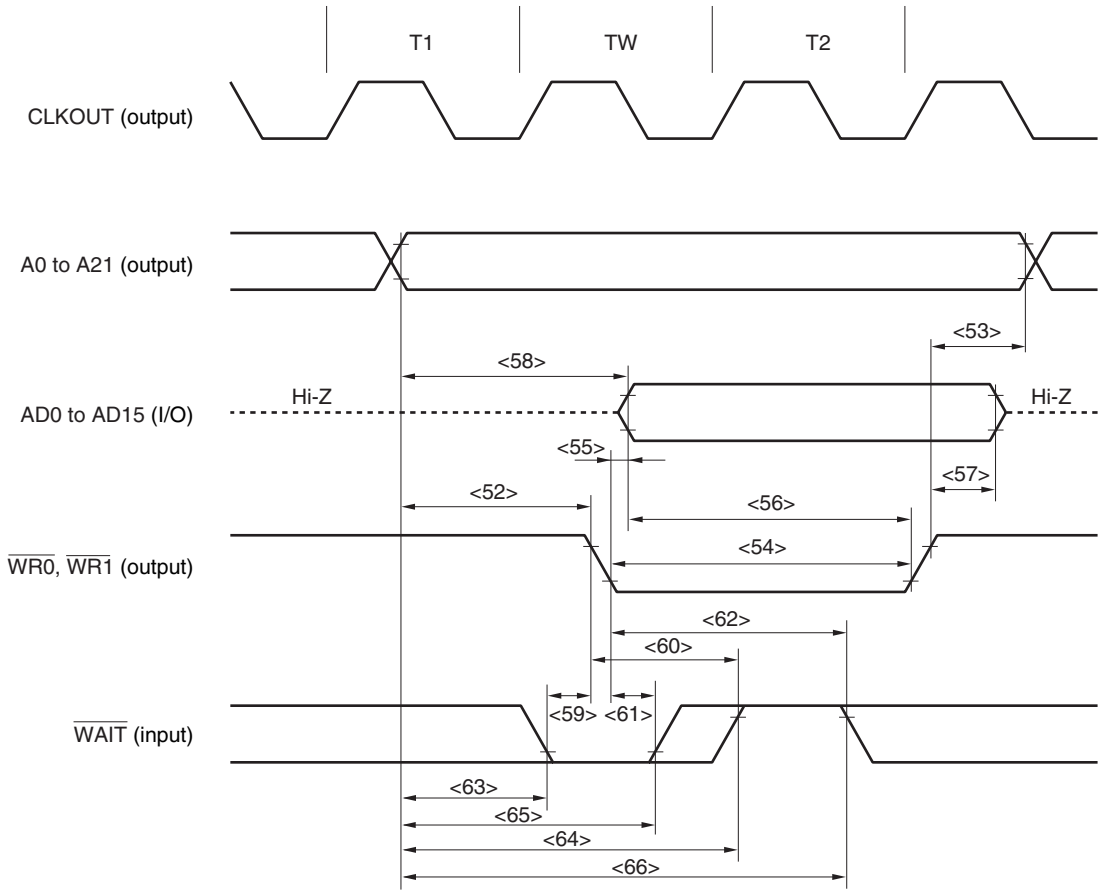
2. t_{ASW}: Number of address setup wait clocks
t_{AHW}: Number of address hold wait clocks

3. $T = 1/f_{CPU}$ (f_{CPU}: CPU operating clock frequency)

4. n: Number of wait clocks inserted in the bus cycle
The sampling timing changes when a programmable wait is inserted.

5. The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

Write Cycle (CLKOUT Asynchronous): In Separate Bus Mode



Remark $\overline{RD}$ is high level.

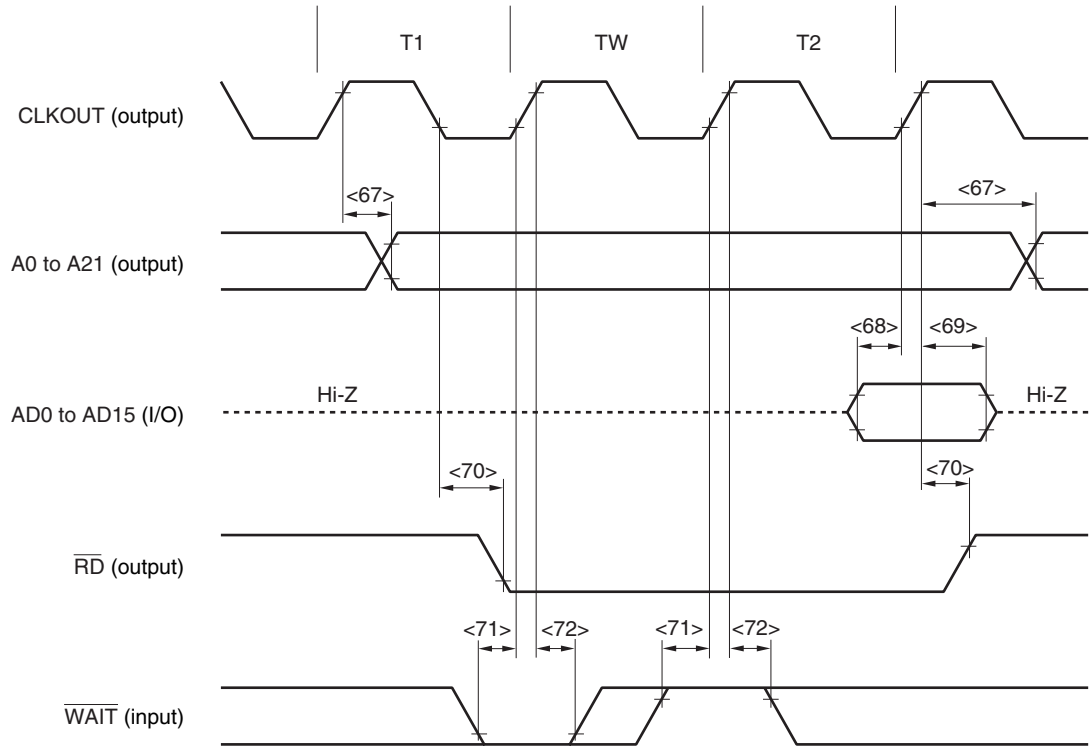
(c) Read cycle (CLKOUT synchronous): In separate bus mode

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
Delay time from CLKOUT↑ to address	t_{DKSA}	<67>		2	25	ns
Data input setup time (to CLKOUT↑)	t_{SISDK}	<68>		20		ns
Data input hold time (from CLKOUT↑)	t_{HKISD}	<69>		0		ns
Delay time from CLKOUT↓↑ to $\overline{RD}$	t_{DKSR}	<70>		–2	12	ns
$\overline{WAIT}$ setup time (to CLKOUT↑)	t_{SWTK}	<71>		20		ns
$\overline{WAIT}$ hold time (from CLKOUT↑)	t_{HKWT}	<72>		0		ns

Remark The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

Read Cycle (CLKOUT Synchronous, 1 Wait): In Separate Bus Mode



Remark $\overline{WR0}$ and $\overline{WR1}$ are high level.

(d) Write cycle (CLKOUT synchronous): In separate bus mode

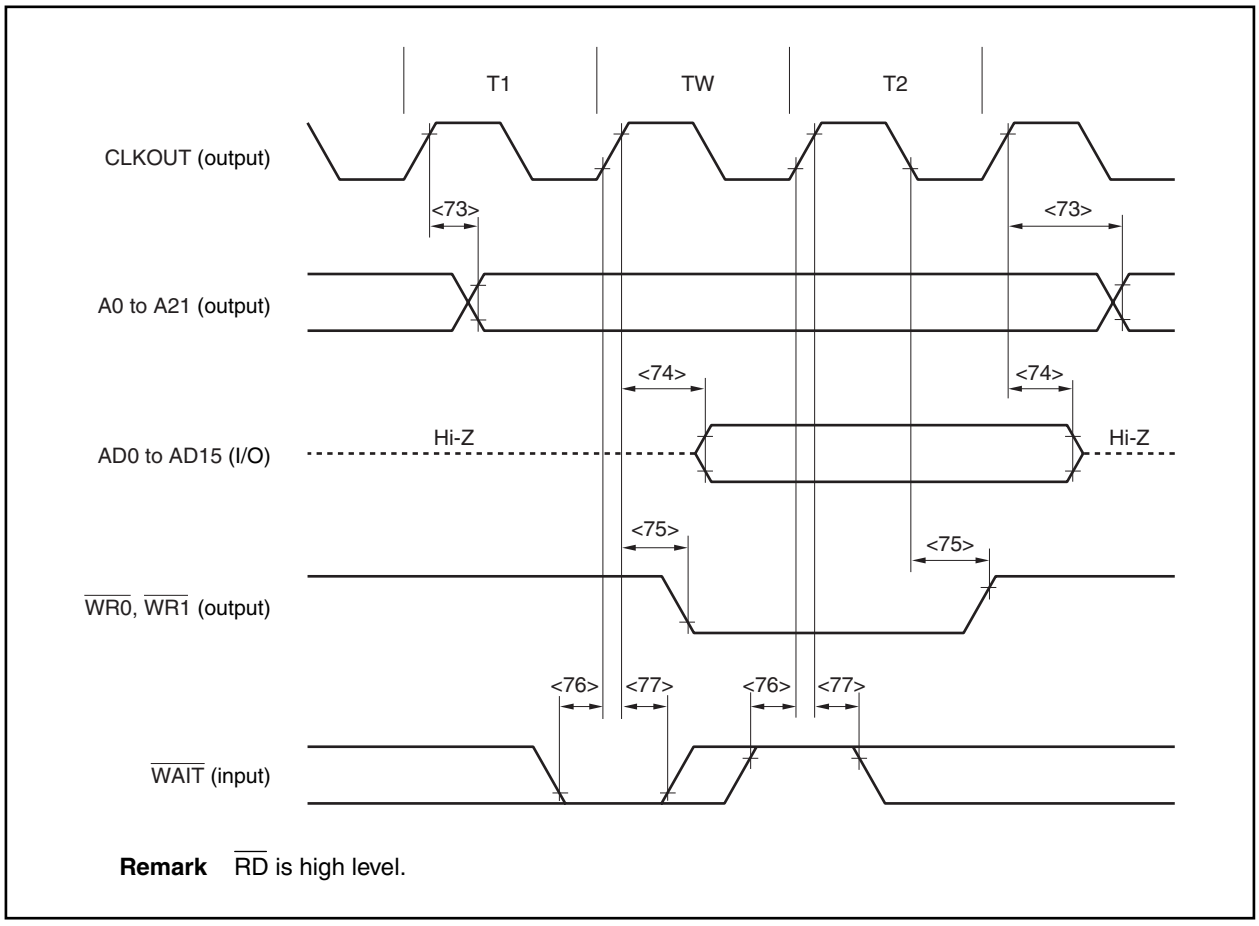
(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
Delay time from CLKOUT↑ to address	t_{DKSA}	<73>		2	25	ns
Delay time from CLKOUT↑ to data output	t_{DKSD}	<74>		2	15	ns
Delay time from CLKOUT↑↓ to $\overline{WRm}$	t_{DKSW}	<75>		–2	12	ns
$\overline{WAIT}$ setup time (to CLKOUT↑)	t_{SWTK}	<76>		20		ns
$\overline{WAIT}$ hold time (from CLKOUT↑)	t_{HKWT}	<77>		0		ns

Remarks 1. m = 0, 1

2. The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

Write Cycle (CLKOUT Synchronous): In Separate Bus Mode



Remark $\overline{RD}$ is high level.

(3) Bus hold

(a) CLKOUT asynchronous

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$\overline{\text{HLDRQ}}$ high-level width	t_{WHQH}	<78>		T + 10		ns
$\overline{\text{HLDAK}}$ low-level width	t_{WHAL}	<79>		T – 15		ns
Delay time from $\overline{\text{HLDAK}}\uparrow$ to bus output	t_{DHAC}	<80>		–3		ns
Delay time from $\overline{\text{HLDRQ}}\downarrow$ to $\overline{\text{HLDAK}}\downarrow$	t_{DHQHA1}	<81>			(2n + 7.5)T + 25	ns
Delay time from $\overline{\text{HLDRQ}}\uparrow$ to $\overline{\text{HLDAK}}\uparrow$	t_{DHQHA2}	<82>		0.5T	1.5T + 25	ns

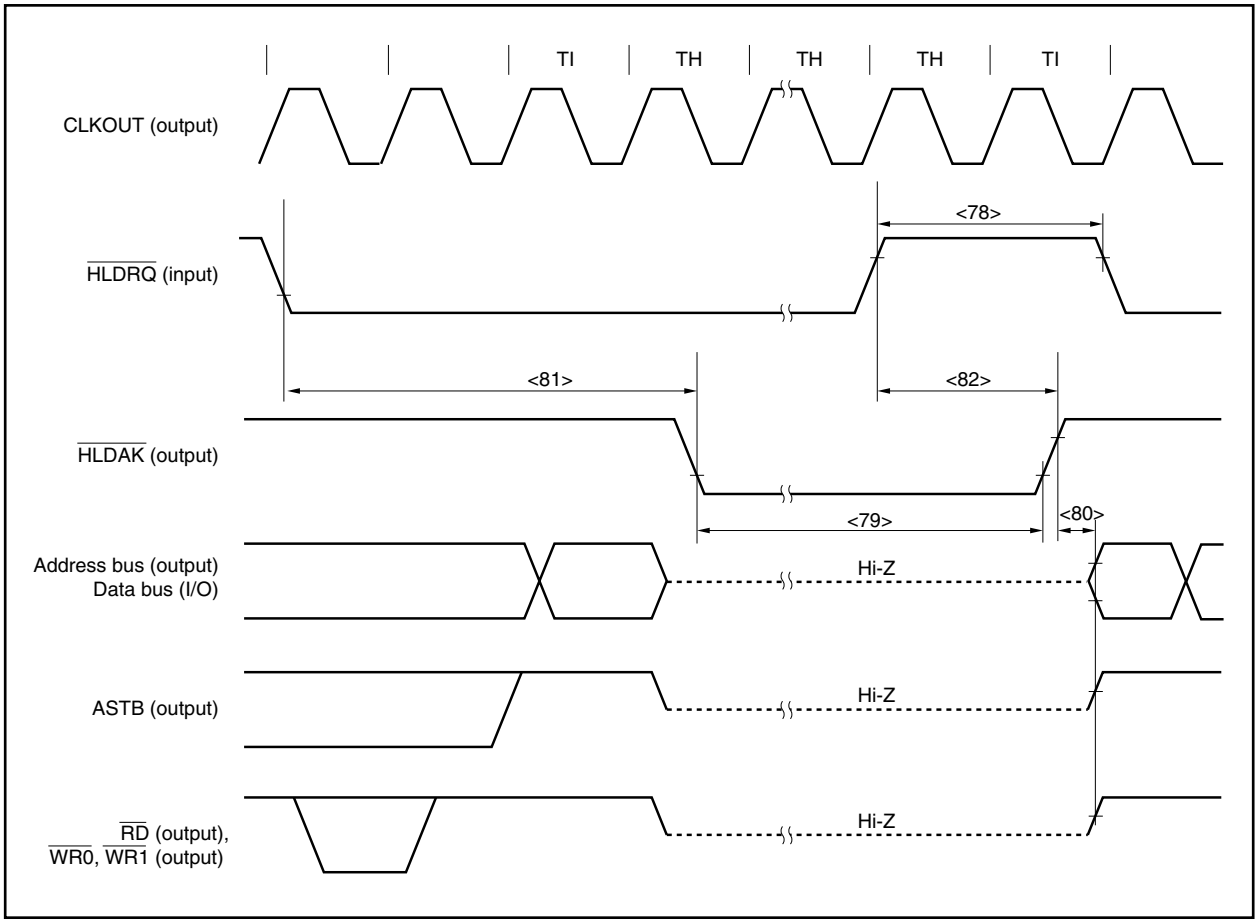
Remarks 1. T = 1/f_{CPU} (f_{CPU}: CPU operating clock frequency)

2. n: Number of wait clocks inserted in the bus cycle

The sampling timing changes when a programmable wait is inserted.

3. The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

Bus Hold (CLKOUT Asynchronous)



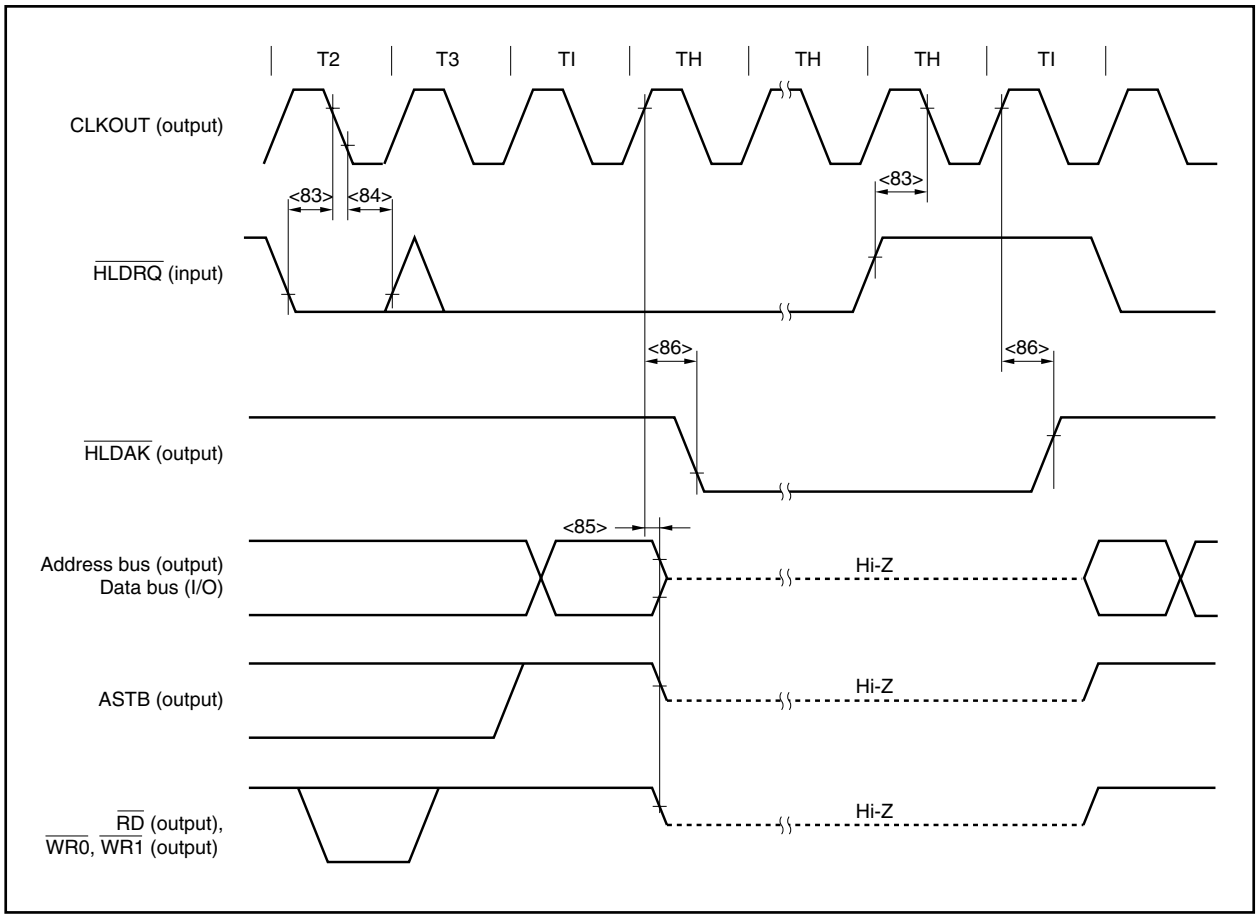
(b) CLKOUT synchronous

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$\overline{\text{HLDRQ}}$ setup time (to CLKOUT↓)	t_{SHQK}	<83>		20		ns
$\overline{\text{HLDRQ}}$ hold time (from CLKOUT↓)	t_{HKHQ}	<84>		5		ns
Delay time from CLKOUT↑ to bus float	t_{DKF}	<85>			19	ns
Delay time from CLKOUT↑ to $\overline{\text{HLDAK}}$	t_{DKHA}	<86>			19	ns

Remark The values in the above specifications are values for when clocks with a 1:1 duty ratio are input from X1.

Bus Hold (CLKOUT Synchronous)



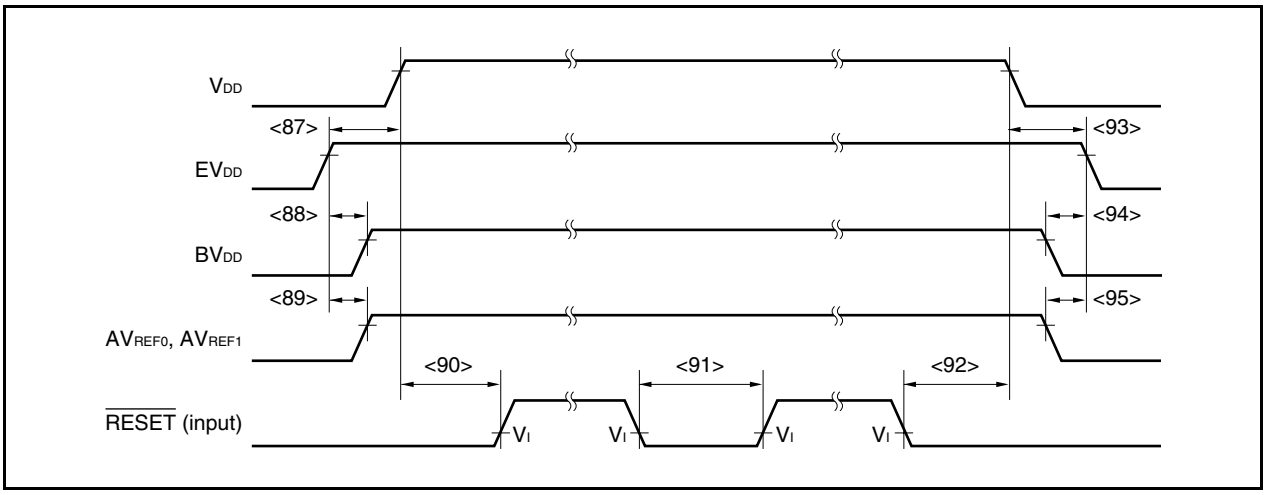
26.9 Basic Operation

(1) Power on/power off/reset timing

(T_A = –40 to +85°C, V_{SS} = AV_{SS} = BV_{SS} = EV_{SS} = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$EV_{DD}\uparrow \rightarrow V_{DD}\uparrow$	t_{REL}	<87>		0		ns
$EV_{DD}\uparrow \rightarrow BV_{DD}\uparrow$	t_{REB}	<88>		0	t_{REL}	ns
$EV_{DD}\uparrow \rightarrow AV_{REF0}$, $AV_{REF1}\uparrow$	t_{REA}	<89>		0	t_{REL}	ns
$EV_{DD}\uparrow \rightarrow \overline{RESET}\uparrow$	t_{RER}	<90>		500 + t_{REG} Note		ns
$\overline{RESET}$ low-level width	t_{WRSL}	<91>	Analog noise elimination	500		ns
$\overline{RESET}\downarrow \rightarrow V_{DD}\downarrow$	t_{FRE}	<92>		500		ns
$V_{DD}\downarrow \rightarrow EV_{DD}\downarrow$	t_{FEL}	<93>		0		ns
$BV_{DD}\downarrow \rightarrow EV_{DD}\downarrow$	t_{FEB}	<94>		0	t_{FEL}	ns
AV_{REF0}, $AV_{REF1}\downarrow \rightarrow EV_{DD}\downarrow$	t_{FEA}	<95>		0	t_{FEL}	ns

Note Depends on the on-chip regulator characteristics.



(2) Interrupt pin timing

(T_A = –40 to +85°C, $BV_{DD} \le V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
NMI high-level width	t_{WNIH}	Analog noise elimination	500		ns
NMI low-level width	t_{WNIL}	Analog noise elimination	500		ns
INTPn high-level width	t_{WITH}	n = 0 to 7 (Analog noise elimination)	500		ns
		n = 3 (Digital noise elimination)	$3T_{SMP}$ + 20		ns
INTPn low-level width	t_{WITL}	n = 0 to 7 (Analog noise elimination)	500		ns
		n = 3 (Digital noise elimination)	$3T_{SMP}$ + 20		ns

Remark T_{SMP}: Noise elimination sampling clock cycle

(3) Key return timing

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
KRn high-level width	t_{WKRH}	Analog noise elimination	500		ns
KRn low-level width	t_{WKRL}	Analog noise elimination	500		ns

Remark n = 0 to 7

(4) Timer timing

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
TI high-level width	t_{TIH}	TIP00, TIP01, TIP10, TIP11, TIP20, TIP21, TIP30, TIP31, TIP40, TIP41, TIP50[Note], TIP51[Note]	2T + 20		ns
<R> TI low-level width	t_{TIL}		2T + 20		ns

<R> **Note** μPD703253Y only

Remark T = 1/fxx

(5) UART timing

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Transmit rate				312.5	kbps
ASCK0 cycle time				10	MHz

(6) CSIB timing

(a) Master mode

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$\overline{SCKBn}$ cycle time	t_{KCY1}	<96>		125		ns
$\overline{SCKBn}$ high-/low-level width	t_{KH1}, t_{KL1}	<97>		$t_{KCY1}/2 - 5$		ns
SIBn setup time (to $\overline{SCKBn}\uparrow$)	t_{SIK1}	<98>		30		ns
SIBn hold time (from $\overline{SCKBn}\uparrow$)	t_{KSI1}	<99>		30		ns
Delay time from $\overline{SCKBn}\downarrow$ to SOBn output	t_{KSO1}	<100>			30	ns

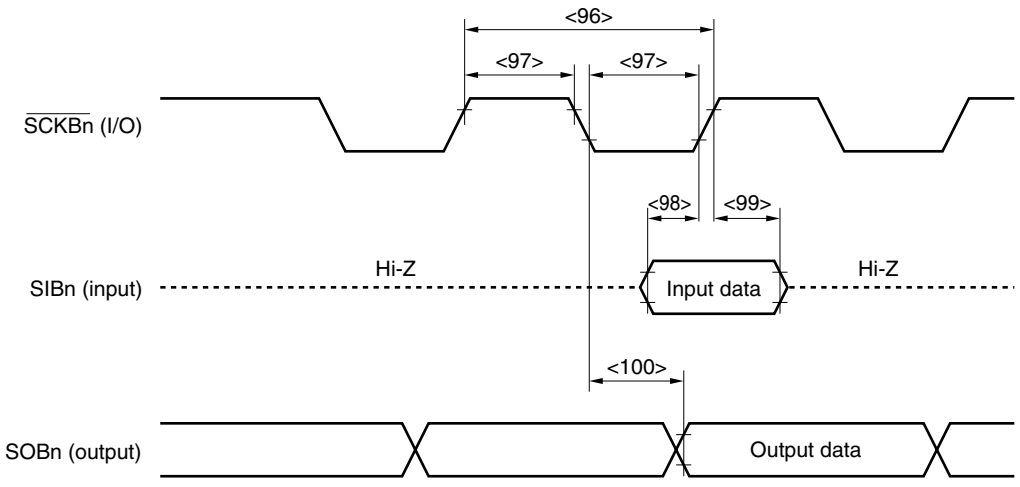
Remark n = 0, 1, 3

(b) Slave mode

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$\overline{SCKBn}$ cycle time	t_{KCY2}	<96>		125		ns
$\overline{SCKBn}$ high-/low-level width	t_{KH2}, t_{KL2}	<97>		57.5		ns
SIBn setup time (to $\overline{SCKBn}\uparrow$)	t_{SIK2}	<98>		30		ns
SIBn hold time (from $\overline{SCKBn}\uparrow$)	t_{KSI2}	<99>		30		ns
Delay time from $\overline{SCKBn}\downarrow$ to SOBn output	t_{KSO2}	<100>			30	ns

Remark n = 0, 1, 3



Remark n = 0, 1, 3

(7) I^2C bus mode

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter		Symbol		Normal Mode		High-Speed Mode		Unit
				MIN.	MAX.	MIN.	MAX.	
SCL0n clock frequency		f_{CLK}		0	100	0	400	kHz
Bus free time (Between start and stop conditions)		t_{BUF}	<101>	4.7	–	1.3	–	μs
Hold time[Note 1]		$t_{HD:STA}$	<102>	4.0	–	0.6	–	μs
SCL0n clock low-level width		t_{LOW}	<103>	4.7	–	1.3	–	μs
SCL0n clock high-level width		t_{HIGH}	<104>	4.0	–	0.6	–	μs
Setup time for start/restart conditions		$t_{SU:STA}$	<105>	4.7	–	0.6	–	μs
Data hold time	CBUS compatible master	$t_{HD:DAT}$	<106>	5.0	–	–	–	μs
	I^2C mode			0[Note 2]	–	0[Note 2]	0.9[Note 3]	μs
Data setup time		$t_{SU:DAT}$	<107>	250	–	100[Note 4]	–	ns
SDA0n and SCL0n signal rise time		t_R	<108>	–	1000	20 + 0.1Cb[Note 5]	300	ns
SDA0n and SCL0n signal fall time		t_F	<109>	–	300	20 + 0.1Cb[Note 5]	300	ns
Stop condition setup time		$t_{SU:STO}$	<110>	4.0	–	0.6	–	μs
Pulse width of spike suppressed by input filter		t_{SP}	<111>	–	–	0	50	ns
Capacitance load of each bus line		Cb		–	400	–	400	pF

Notes 1. At the start condition, the first clock pulse is generated after the hold time.

2. The system requires a minimum of 300 ns hold time internally for the SDA0n signal (at $V_{IHmin.}$ of SCL0n signal) in order to occupy the undefined area at the falling edge of SCL0n.

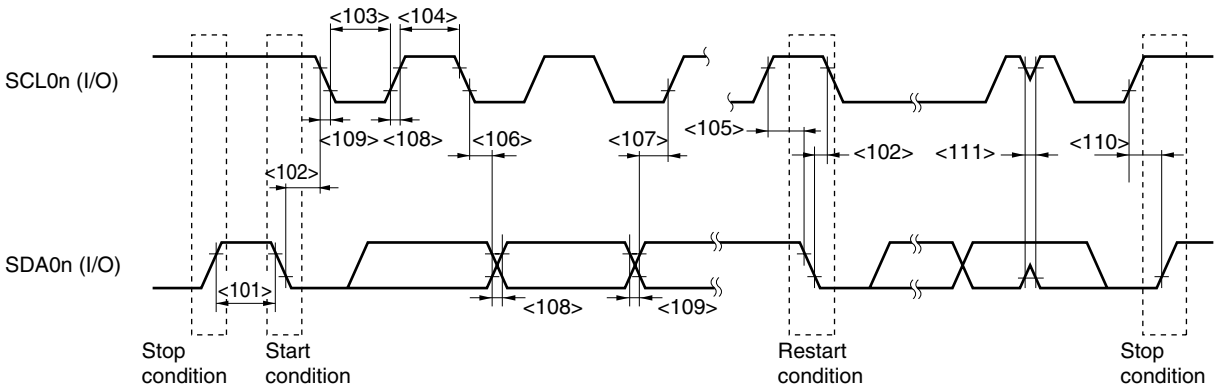
3. If the system does not extend the SCL0n signal low hold time (t_{LOW}), only the maximum data hold time ($t_{HD:DAT}$) needs to be satisfied.

4. The high-speed mode I^2C bus can be used in the normal-mode I^2C bus system. In this case, set the high-speed mode I^2C bus so that it meets the following conditions.

- If the system does not extend the SCL0n signal's low state hold time:
  $t_{SU:DAT} \geq 250$ ns
- If the system extends the SCL0n signal's low state hold time:
  Transmit the following data bit to the SDA0n line prior to the SCL0n line release ($t_{Rmax.} + t_{SU:DAT}$ = 1,000 + 250 = 1,250 ns: Normal mode I^2C bus specification).

5. Cb: Total capacitance of one bus line (unit: pF)

Remark n = 0, 1



Remark n = 0, 1

(8) IEBus controller (μPD703252Y only)

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

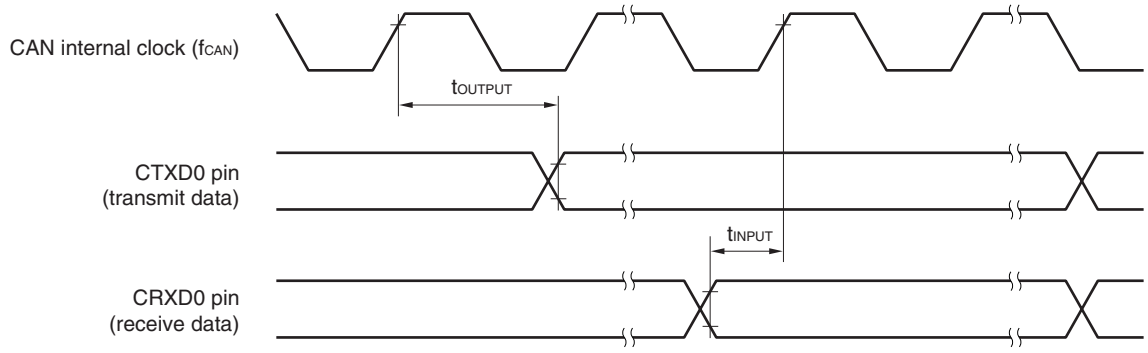
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
IEBus system clock frequency	f_S	Communication mode: Modes 1, 2	5.91	6.00[Note]	6.09	MHz
			6.20	6.29[Note]	6.38	MHz

Note IEBus system clock frequencies 6.0 MHz and 6.29 MHz cannot be used together.

<R> **(9) CAN timing (μPD703253Y only)**

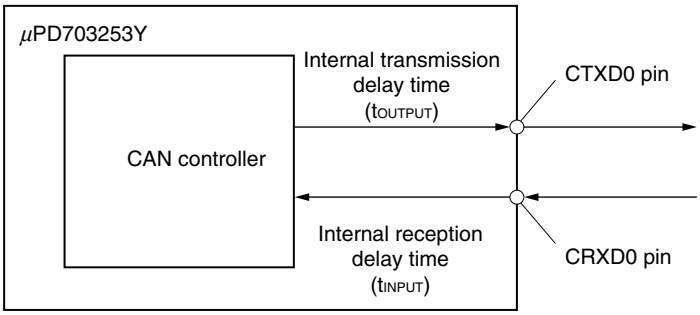
(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Transmit rate				1	Mbps
Internal delay time	t_{NODE}			100	ns



Remark CAN internal clock (f_{CAN}): CAN baud rate clock

Internal delay time (t_{NODE}) = Internal transmission delay time (t_{OUTPUT}) + Internal reception delay time (t_{INPUT})



(10) A/D converter

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, 3.0 V ≤ $AV_{REF0} \leq$ 3.6 V, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution					10	bit
Overall error[Note]		3.0 ≤ AV_{REF0} ≤ 3.6 V			±0.6	%FSR
Conversion time	t_{CONV}		2.6		24	μs
Zero scale error					±0.5	%FSR
Full scale error					±0.5	%FSR
Non-linearity error					±4.0	LSB
Differential linearity error					±4.0	LSB
Analog input voltage	V_{IAN}		AV_{SS}		AV_{REF0}	V
Reference voltage	AV_{REF0}		3.0		3.6	V
AV_{REF0} current	AI_{REF0}	Normal conversion mode		3	6.5	mA
		High-speed conversion mode		4	10	mA
		When A/D converter unused			5	μA

Note Excluding quantization error (±0.05 %FSR).

Caution Do not set (read/write) alternate-function ports during A/D conversion; otherwise the conversion resolution may be degraded.

Remark LSB: Least Significant Bit
FSR: Full Scale Range

<R> (11) D/A converter (μPD703253Y only)

(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, 3.0 V ≤ $AV_{REF1} \leq$ 3.6 V, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution					8	bit
Overall error[Note 1]		R = 2 MΩ			±1.2	%FSR
Settling time		C = 20 pF			3	μs
Output resistor	R_O	Output data 55H		3.5		kΩ
Reference voltage	AV_{REF1}		3.0		3.6	V
AV_{REF1} current[Note 2]	AI_{REF1}	D/A conversion operating		1	2.5	mA
		D/A conversion stopped			5	μA

Notes 1. Excluding quantization error (±0.5 LSB).
2. Value of 1 channel of D/A converter

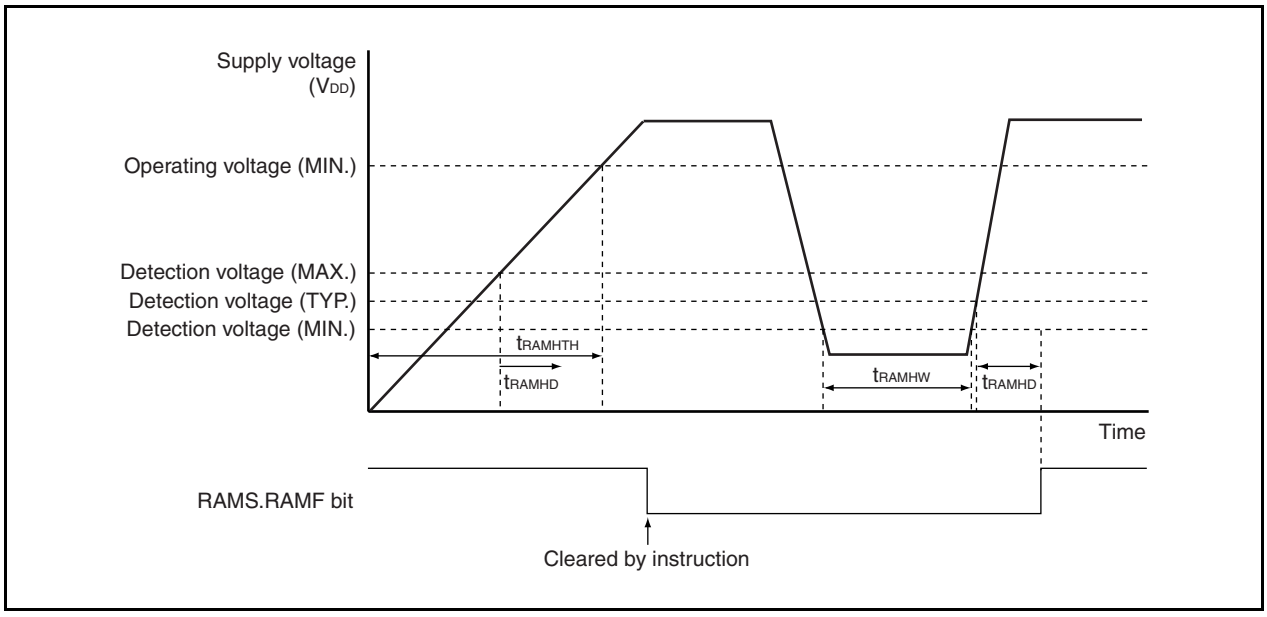
Remark R is the output pin load resistance and C is the output pin load capacitance.

<R> **(12) RAM retention detection (μPD703253Y only)**

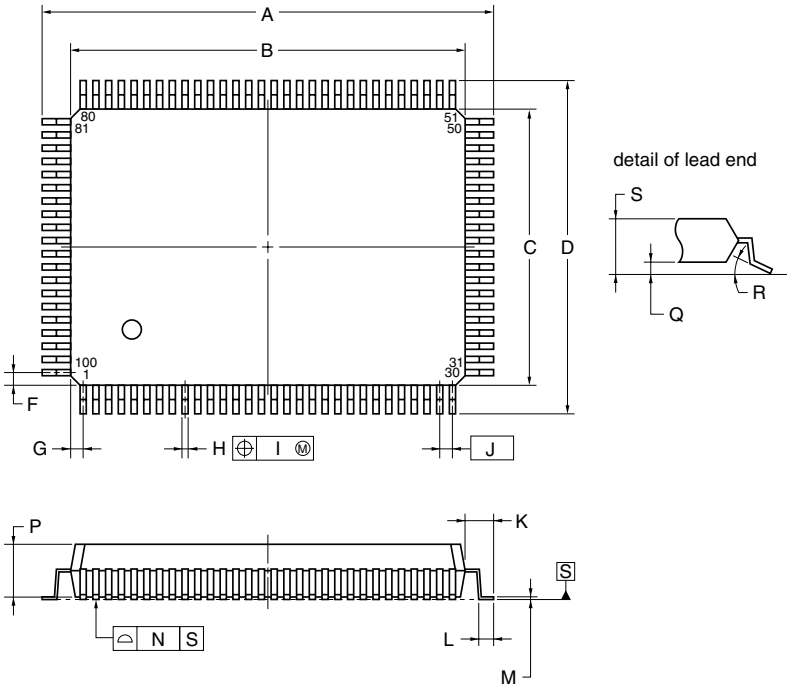
(T_A = –40 to +85°C, $BV_{DD} \leq V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1}$, $V_{SS} = EV_{SS} = BV_{SS} = AV_{SS}$ = 0 V, C_L = 50 pF)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Detection voltage	V_{RAMH}		1.9	2.0	2.1	V
Supply voltage rise time	t_{RAMHTH}	V_{DD} = 0 to 2.85 V	0.002			ms
Response time[Note]	t_{RAMHD}	After V_{DD} reaches 2.1 V		0.2	2.0	ms
Minimum pulse width	t_{RAMHW}		0.2			ms

Note Time required to detect the detection voltage and set the RAMS.RAMF bit.



100-PIN PLASTIC QFP (14x20)



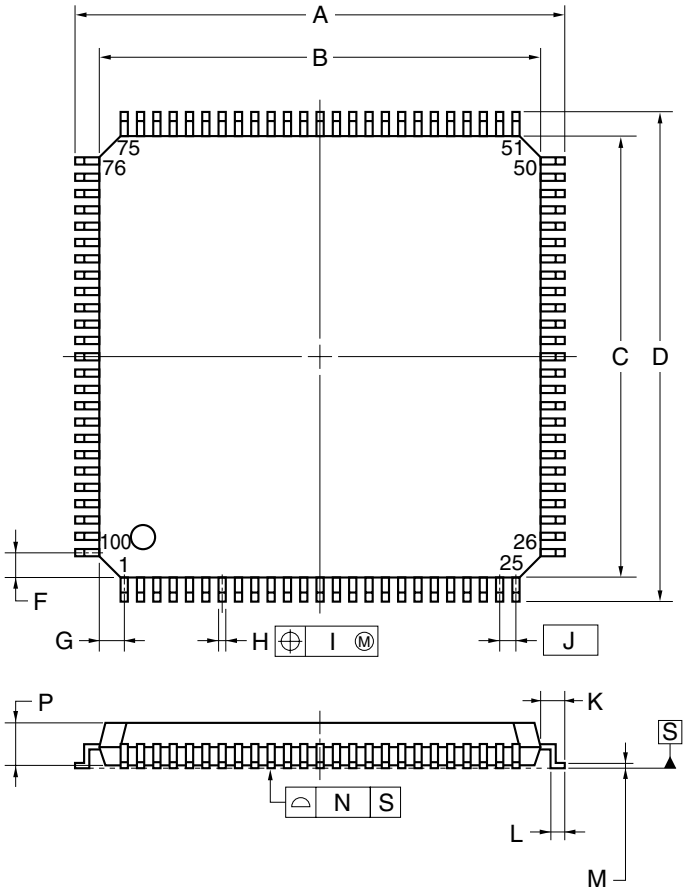
NOTE

Each lead centerline is located within 0.13 mm of its true position (T.P.) at maximum material condition.

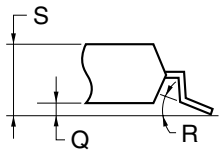
ITEM	MILLIMETERS
A	23.2±0.2
B	20.0±0.2
C	14.0±0.2
D	17.2±0.2
F	0.825
G	0.575
H	$0.32^{+0.08}_{-0.07}$
I	0.13
J	0.65 (T.P.)
K	1.6±0.2
L	0.8±0.2
M	$0.17^{+0.06}_{-0.05}$
N	0.10
P	2.7±0.1
Q	0.125±0.075
R	$3^{\circ}{}^{+7^{\circ}}_{-3^{\circ}}$
S	3.0 MAX.

S100GF-65-JBT-2

100-PIN PLASTIC LQFP (FINE PITCH) (14x14)



detail of lead end



NOTE

Each lead centerline is located within 0.08 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	16.00±0.20
B	14.00±0.20
C	14.00±0.20
D	16.00±0.20
F	1.00
G	1.00
H	$0.22^{+0.05}_{-0.04}$
I	0.08
J	0.50 (T.P.)
K	1.00±0.20
L	0.50±0.20
M	$0.17^{+0.03}_{-0.07}$
N	0.08
P	1.40±0.05
Q	0.10±0.05
R	$3^{\circ}{}^{+7^{\circ}}_{-3^{\circ}}$
S	1.60 MAX.

S100GC-50-8EU, 8EA-2

<R>

CHAPTER 28 RECOMMENDED SOLDERING CONDITIONS

These products should be soldered and mounted under the following recommended conditions.
For technical information, see the following website.

Semiconductor Device Mount Manual (http://www.necel.com/pkg/en/mount/index.html)

Table 28-1. Surface Mounting Type Soldering Conditions

μPD703249YGF-xxx-JBT-A: 100-pin plastic QFP (14 × 20)
μPD703249YGC-xxx-8EA-A: 100-pin plastic LQFP (fine pitch) (14 × 14)
μPD703252YGF-xxx-JBT-A: 100-pin plastic QFP (14 × 20)
μPD703252YGC-xxx-8EA-A: 100-pin plastic LQFP (fine pitch) (14 × 14)
μPD703253YGC-×××-8EA-A: 100-pin plastic LQFP (fine pitch) (14 × 14)

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 20 to 72 hours)	IR60-207-3
Wave soldering	For details, contact an NEC Electronics sales representative.	–
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.

Caution Do not use different soldering methods together (except for partial heating).

Remarks 1. The V850ES/SG1 is lead-free products.

2. For soldering methods and conditions other than those recommended, please contact an NEC Electronics sales representative.

APPENDIX A DEVELOPMENT TOOLS

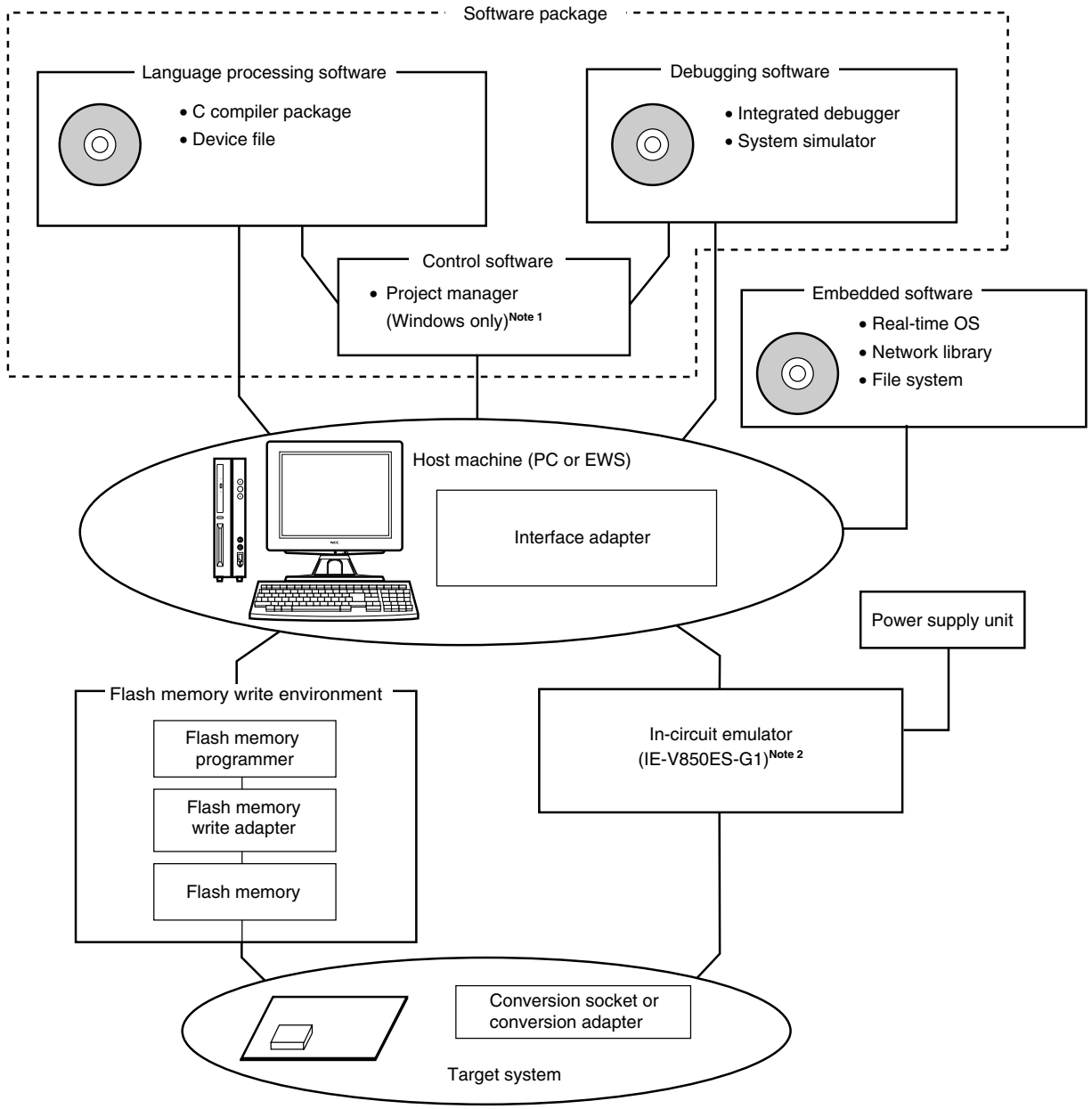
The following development tools are available for the development of systems that employ the V850ES/SG1. Figure A-1 shows the development tool configuration.

- **Support for PC98-NX series**
  Unless otherwise specified, products supported by IBM PC/AT™ compatibles are compatible with PC98-NX series computers. When using PC98-NX series computers, refer to the explanation for IBM PC/AT compatibles.

- **Windows™**
  Unless otherwise specified, “Windows” means the following OSs.
  - Windows 98, 2000
  - Windows Me
  - Windows XP
  - Windows NT™ Ver. 4.0

Figure A-1. Development Tool Configuration (1/4)

(1) When using in-circuit emulator IE-V850ES-G1



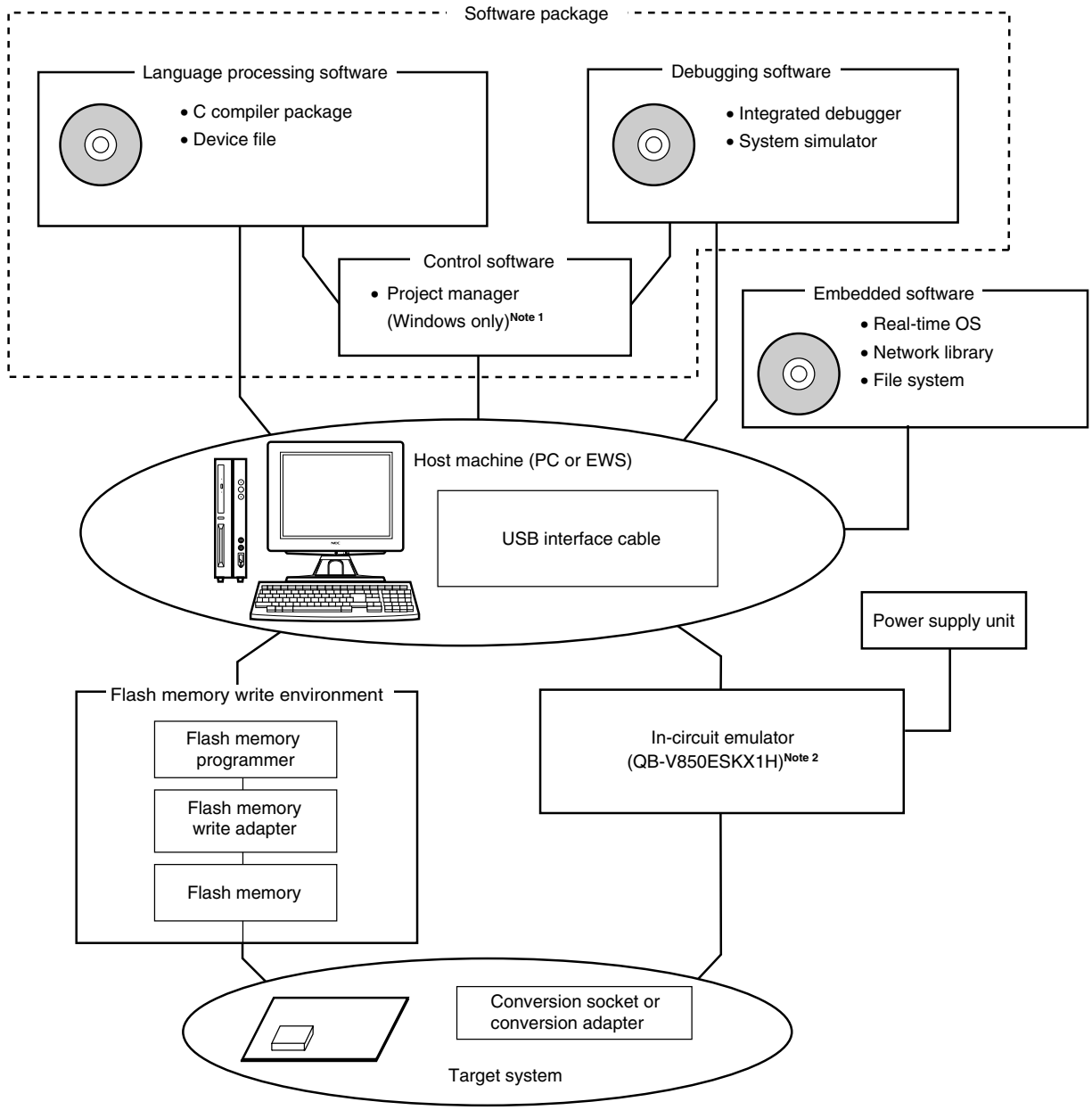
Notes 1. The project manager PM+ is included in the C compiler package.
The PM+ is only used for Windows.

2. No flash memory versions are provided in the V850ES/SG1. However, the μPD70F3261Y, <R> 70F3271Y, and 70F3281Y of the V850ES/SG2 can be used as the flash memory versions for developing V850ES/SG1 software.

3. Products other than in-circuit emulator IE-V850ES-G1 are all sold separately.

Figure A-1. Development Tool Configuration (2/4)

(2) When using IECUBE® QB-V850ESSX2



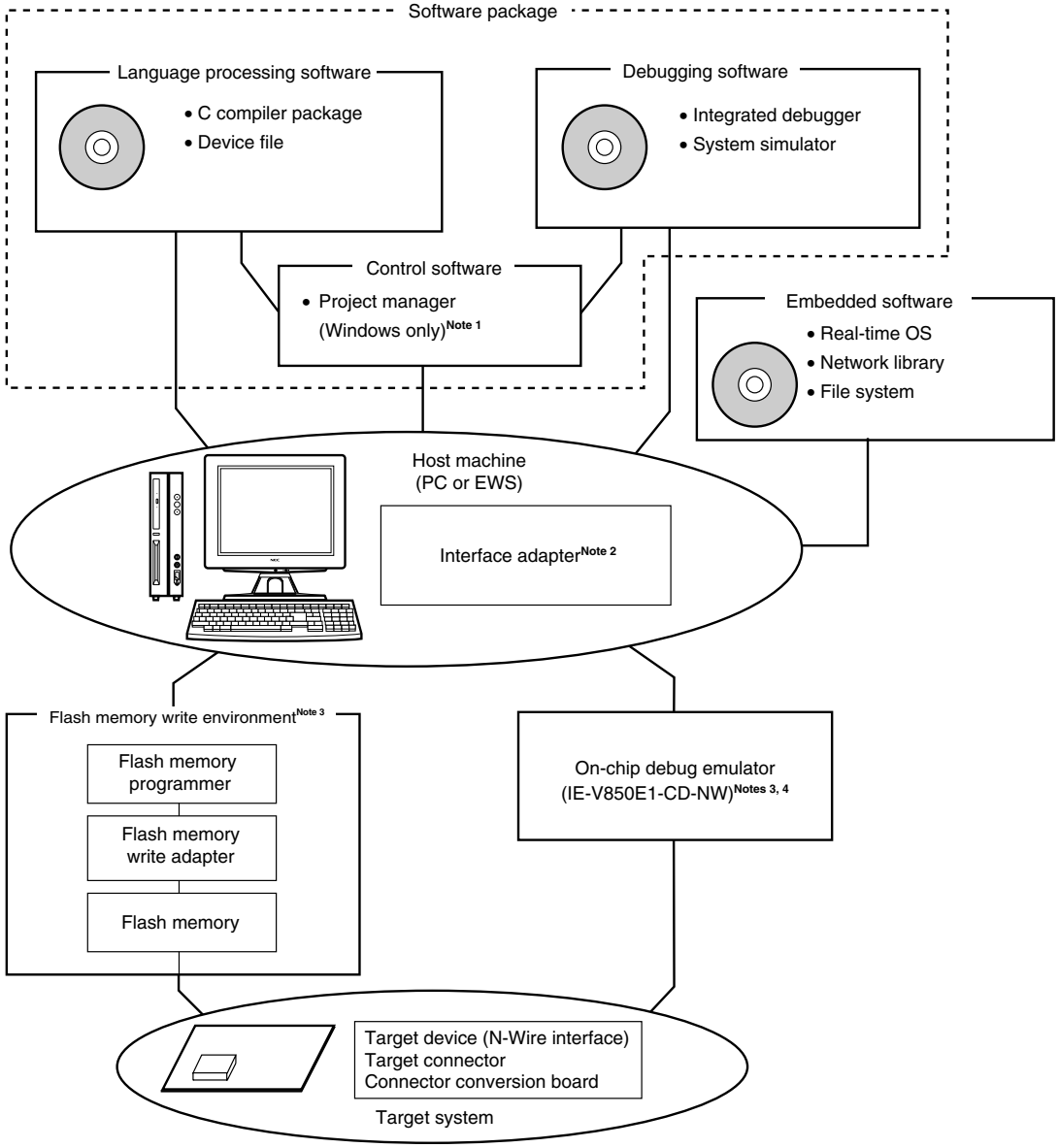
Notes 1. The project manager PM+ is included in the C compiler package.
The PM+ is only used for Windows.

2. No flash memory versions are provided in the V850ES/SG1. However, the μPD70F3261Y, 70F3271Y, and 70F3281Y of the V850ES/SG2 can be used as the flash memory versions for developing V850ES/SG1 software. <R>

3. In-circuit emulator QB-V850ESSX2 is supplied with integrated debugger ID850QB, a simple programmer PG-FPL, power supply unit, and USB interface cable. Any other products are sold separately.

Figure A-1. Development Tool Configuration (3/4)

<R> **(3) When using on-chip debug emulator IE-V850E1-CD-NW**



Notes 1. The project manager PM+ is included in the C compiler package.
The PM+ is only used for Windows.

2. The IE-V850E1-CD-NW supports only the PCMCIA card interface.

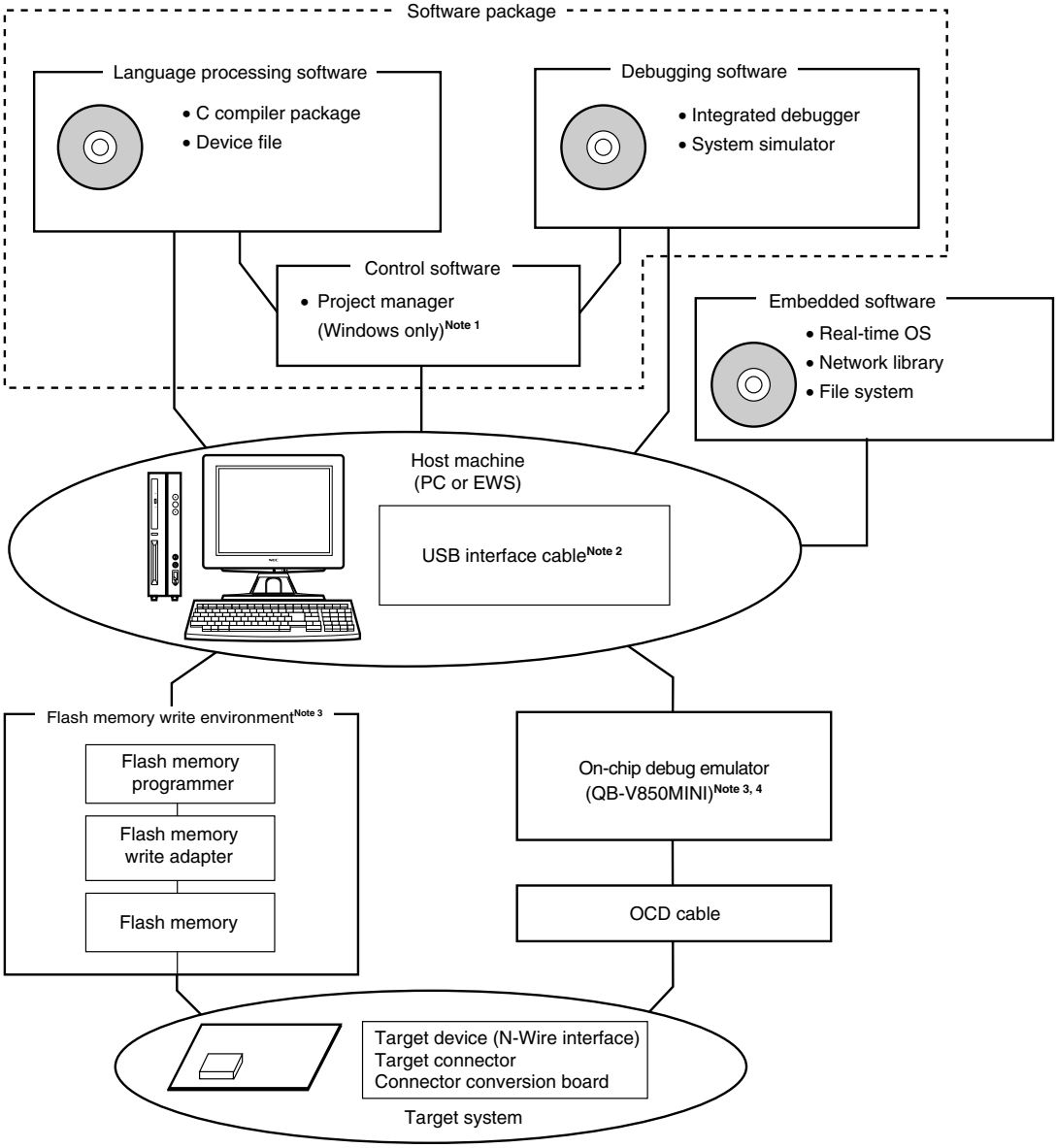
3. No flash memory versions and on-chip debug function versions are provided in the V850ES/SG1. However, the μPD70F3261Y, 70F3271Y, and 70F3281Y of the V850ES/SG2 can be used as the flash memory versions for developing V850ES/SG1 software. In this case, the on-chip debug function can also be used.

4. The IE-V850E1-CD-NW is supplied with an ID850QB, IE connection cable, IE connector, and connector conversion board. All other products are sold separately.

Figure A-1. Development Tool Configuration (4/4)

<R>

(4) When using MINICUBE® QB-V850MINI



Notes 1. The project manager PM+ is included in the C compiler package.
The PM+ is only used for Windows.

2. The QB-V850MINI supports the USB interface.

3. No flash memory versions and on-chip debug function versions are provided in the V850ES/SG1. However, the μPD70F3261Y, 70F3271Y, and 70F3281Y of the V850ES/SG2 can be used as the flash memory versions for developing V850ES/SG1 software. In this case, the on-chip debug function can also be used.

4. The QB-V850MINI is supplied with a KEL connector and KEL adapter as target connectors and connector conversion board in addition to an ID850QB, USB interface cable, and OCD cable. All other target connectors are sold separately.

A.1 Software Package

SP850 V850 microcontroller software package	Development tools (software) common to the V850 microcontroller are combined in this package.
	Part number: μS××××SP850

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××SP850

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	

A.2 Language Processing Software

CA850 C compiler package	This compiler converts programs written in C language into object codes executable with a microcontroller. This compiler is started from project manager PM+.
	Part number: μS××××CA703000
<R> DF703252/DF703253[Note] <R> Device file	This file contains information peculiar to the device. This device file should be used in combination with a tool (CA850 or ID850). The corresponding OS and host machine differ depending on the tool to be used.

Note μPD703249Y, 703252Y: DF703252

<R> μPD703253Y: DF703253

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××CA703000

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	
3K17	SPARCstation™	SunOS™ (Rel. 4.1.4), Solaris™ (Rel. 2.5.1)	

A.3 Control Software

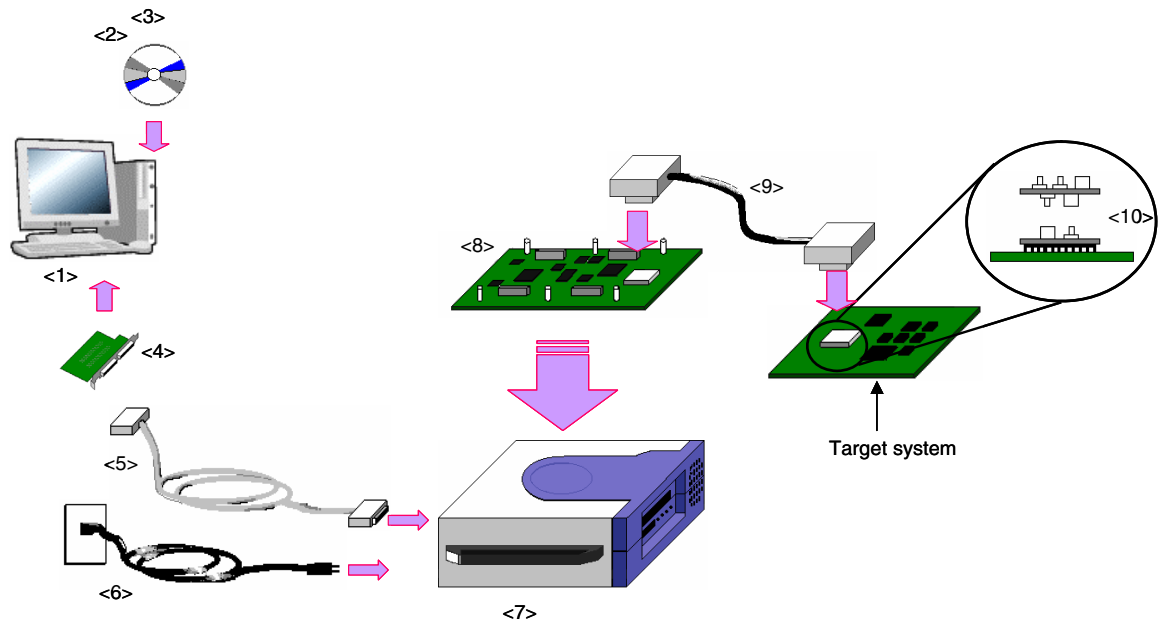
PM+ Project manager	This is control software designed to enable efficient user program development in the Windows environment. All operations used in development of a user program, such as starting the editor, building, and starting the debugger, can be performed from PM+. **<Caution>** PM+ is included in the C compiler package CA850. It can only be used in Windows.

A.4 Debugging Tools (Hardware)

A.4.1 When using in-circuit emulator IE-V850ES-G1

The system configuration when connecting the IE-703288-G1-EM1 to the IE-V850ES-G1 and use it connecting to the host machine (PC-9800 series, PC/AT compatible) is shown below.

Figure A-2. System Configuration (IE-V850ES-G1 Used)



<1> Host machine (PC-9800 series, PC/AT compatibles)
<2> Debugger
<3> Device file
<4> PC interface board (for PCI bus or PCMCIA)
<5> PC interface cable (supplied with the IE-V850ES-G1)
<6> Power supply cable (supplied with the IE-V850ES-G1)
<7> In-circuit emulator (IE-V850ES-G1)
<8> In-circuit emulator emulation board (IE-703288-G1-EM1)
<9> Probe (supplied with the IE-703288-G1-EM1)
<10> EV-703288GF100 (for GF package) or EV-703288GC100 (for GC package)

<7> IE-V850ES-G1 In-circuit emulator	The in-circuit emulator serves to debug hardware and software when developing application systems using a V850 microcontroller product. It corresponds to the integrated debugger ID850. This emulator should be used in combination with a power supply unit, emulation probe, and the interface adapter required to connect this emulator to the host machine.
<4> IE-70000-CD-IF-A PC card interface	This is PC card and interface cable required when using a notebook-type computer as the host machine (PCMCIA socket compatible).
<4> IE-70000-PCI-IF-A Interface adapter	This adapter is required when using a computer with a PCI bus as the host machine.
<8> IE-703288-G1-EM1 Emulation board	This board emulates the operations of the peripheral hardware peculiar to a device. It should be used in combination with an in-circuit emulator.
<9> GXP-CABLE Emulation probe	This probe is used to connect the in-circuit emulator and target system. This is supplied with emulation board IE-703288-G1-EM1.
<10> EV-703288GF100 Conversion adapter	This conversion adapter is used to connect the emulation probe and target system board on which a 100-pin plastic LQFP (GF-JBT type) can be mounted.
<10> EV-703288GC100 Conversion adapter	This conversion adapter is used to connect the emulation probe and target system board on which a 100-pin plastic LQFP (GC-8EA type) can be mounted.

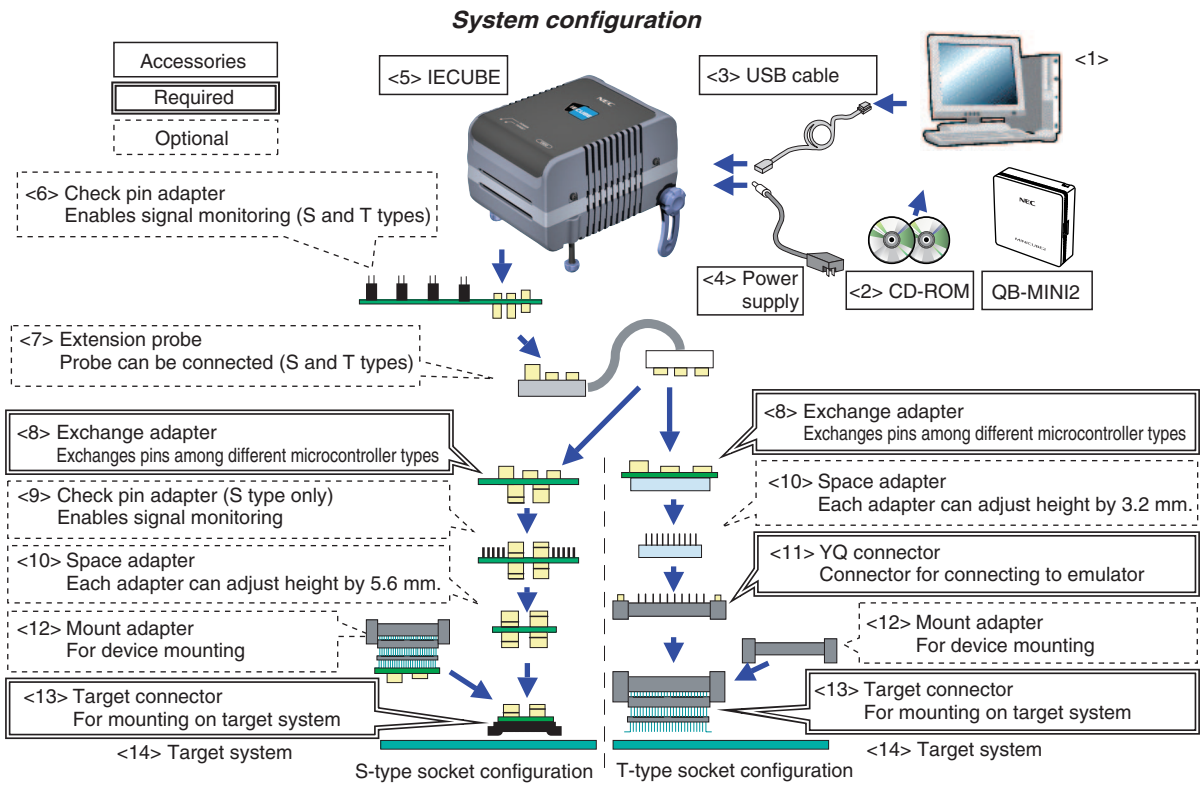
Remarks 1. The numbers in the square brackets correspond to the numbers in Figure A-2.

2. EV-703288GF100 and EV-703288GC100 are products of Application Corporation.
TEL: +81-42-732-1377 Application Corporation

A.4.2 When using IECUBE QB-V850ESSX2

The system configuration when connecting the QB-V850ESSX2 to the host machine (PC-9821 series, PC/AT compatible) is shown below. If no option products are prepared, connection is possible.

Figure A-3. System Configuration (QB-V850ESSX2 Used) (1/2)



<1> Host machine (PC-9821 series, IBM-PC/AT compatibles)

<2> Debugger, USB driver, manuals, etc. (ID850QB Disk, Accessory Disk[Note 1])

<3> USB interface cable

<4> AC adapter

<5> In-circuit emulator (QB-V850ESSX2)

<R> <6> Check pin adapter (common to S and T types) (QB-144-CA-01) (option)

<7> Extension probe coaxial type (common to S and T types) (QB-144-EP-01S) (option)

<8> Exchange adapter[Note 2] (S type: QB-100GF-EA-01S (for GF package), QB-100GC-EA-01S (for GF package), T type: QB-100GF-EA-01T (for GF package), QB-100GC-EA-01T (for GC package)

<9> Check pin adapter[Note 3] (S type only) (QB-100-CA-01S) (option)

<10> Space adapter[Note 3] (S type: QB-100-SA-01S, T type: QB-100GF-YS-01T (for GF package), QB-100GC-YS-01T (for GC package)) (option)

<11> YQ connector[Note 2] (T type only) (QB-100GF-YQ-01T (for GF package), QB-100GC-YQ-01T (for GC package))

<12> Mount adapter (S type: QB-100GF-MA-01S (for GF package), QB-100GC-MA-01S (for GC package), T type: QB-100GF-HQ-01T (for GF package), QB-100GC-HQ-01T (GC package) (option)

Figure A-3. System Configuration (QB-V850ESSX2 Used) (2/2)

<13> Target connector[Note 2] (S type: QB-100GF-TC-01S (for GF package), QB-100GC-TC-01S (for GC package), T type: QB-100GF-NQ-01T (for GF package), QB-100GC-NQ-01T (for GF package))

<14> Target system

Notes 1. Obtain the device file from the NEC Electronics website.
http://www.necel.com/micro/ods/jpn/index.html

2. Depending on the ordering number, supplied with the device.

- When QB-V850ESSX2-ZZZ is ordered
  The exchange adapter and the target connector are not supplied.
- When QB-V850ESSX2-S100GF is ordered
  The QB-100GF-EA-01S and QB-100GF-TC-01S are supplied.
- When QB-V850ESSX2-S100GC is ordered
  The QB-100GC-EA-01S and QB-100GC-TC-01S are supplied.
- When QB-V850ESSX2-T100GF is ordered
  The QB-100GF-EA-01T, QB-100GF-YQ-01T, and QB-100GF-NQ-01T are supplied.
- When QB-V850ESSX2-T100GC is ordered
  The QB-100GC-EA-01T, QB-100GC-YQ-01T, and QB-100GC-NQ-01T are supplied.

3. When using both <9> and <10>, the order between <9> and <10> is not cared.

<5> QB-V850ESSX2[Note] In-circuit emulator	The in-circuit emulator serves to debug hardware and software when developing application systems using a V850ES/SG1 product. It supports the integrated debugger ID850QB. This emulator should be used in combination with a power supply unit and emulation probe. Use USB to connect this emulator to the host machine.
<3> USB interface cable	Cable to connect the host machine and the QB-V850ESSX2.
<4> AC adapter	100 to 240 V can be supported by replacing the AC plug.
<8> QB-100GF-EA-01S, QB-100GC-EA-01S, QB-100GF-EA-01T, QB-100GC-EA-01T Exchange adapter	Adapter to perform pin conversion. • QB-100GF-EA-01S, QB-100GF-EA-01T: For 100-pin plastic QFP (GF-JBT type) • QB-100GC-EA-01S, QB-100GC-EA-01T: For 100-pin plastic LQFP (GC-8EA type)
<9> QB-100-CA-01S Check pin adapter	Adapter used in waveform monitoring using the oscilloscope, etc.
<10> QB-100-SA-01S, QB-100GF-YS-01T, QB-100GC-YS-01T Space adapter	Adapter to adjust the height. • QB-100-SA-01S, QB-100GF-YS-01T: For 100-pin plastic QFP (GF-JBT type) • QB-100-SA-01S, QB-100GC-YS-01T: For 100-pin plastic LQFP (GC-8EA type)
<12> QB-100GF-MA-01S, QB-100GC-MA-01S, QB-100GF-HQ-01T, QB-100GC-HQ-01T Mount adapter	Adapter to mount the V850ES/SG1 with socket. • QB-100GF-MA-01S, QB-100GF-HQ-01T: For 100-pin plastic QFP (GF-JBT type) • QB-100GC-MA-01S, QB-100GC-HQ-01T: For 100-pin plastic LQFP (GC-8EA type)
<13> QB-100GF-TC-01S, QB-100GC-TC-01S, QB-100GF-NQ-01T, QB-100GC-NQ-01T Target connector	Connector to solder on the target system. • QB-100GF-TC-01S, QB-100GF-NQ-01T: For 100-pin plastic QFP (GF-JBT type) • QB-100GC-TC-01S, QB-100GC-NQ-01T: For 100-pin plastic LQFP (GC-8EA type)

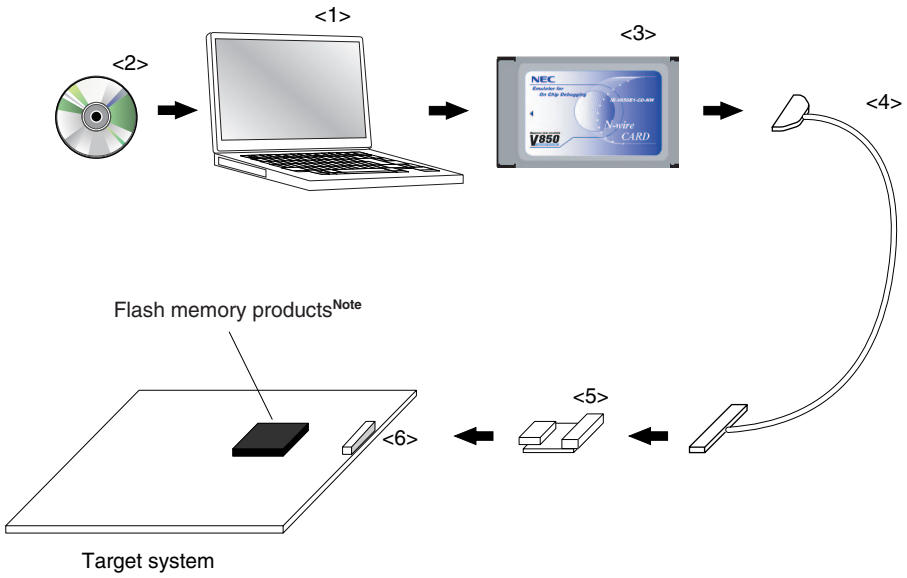
<R> **Note** The QB-V850ESSX2 is supplied with a power supply unit, USB interface cable, and QB-MINI2. It is also supplied with integrated debugger ID850QB as control software.

Remark The numbers in the square brackets correspond to the numbers in Figure A-3.

<R> A.4.3 When using on-chip debug emulator IE-V850E1-CD-NW

The system configuration when connecting the IE-V850E1-CD-NW to the host machine (PC-9821 series, PC/AT compatible) is shown below.

Figure A-4. System Configuration (IE-V850E1-CD-NW Used)



Note No flash memory versions are provided in the V850ES/SG1. However, the μPD70F3261Y, 70F3271Y and 70F3281Y of the V850ES/SG2 can be used as the flash memory versions for developing V850ES/SG1 software.

<1> Host machine	Personal computer including PCMCIA compliant with the PCMCIA2.1/JEIDA standard Ver. 4.2. When using a product which does not have a PCMCIA slot, use a PCI-PCMCIA conversion board or the like. For details on the conversion board, consult an NEC Electronics sales representative.
<2> CD-ROM[Note 1]	The integrated debugger ID850QB, N-Wire Checker, device driver, documents and so on in the CD-ROM format are included. This CD-ROM is supplied with the IE-V850E1-CD-NW.
<3> IE-V850E1-CD-NW On-chip debug emulator	This on-chip debug emulator is used to debug hardware and software when application systems using the V850ES/SG1 is developed. It supports the integrated debugger ID850QB.
<4> IE-V850E1-CD-NW connection cable	This connection cable is used to connect the IE-V850E1-CD-NW and the target system. It is supplied with the IE-V850E1-CD-NW. The cable length is approximately 50 cm.
<5> Connector conversion board KEL adapter	It is supplied with the IE-V850E1-CD-NW.
<6> IE-V850E1-CD-NW connector KEL connector[Note 2]	8830E-026-170S (It is supplied with the IE-V850E1-CD-NW.) 8830E-026-170L (sold separately)

Notes 1. Obtain the device file from the NEC Electronics website.
http://www.necel.com/micro/ods/eng/index.html

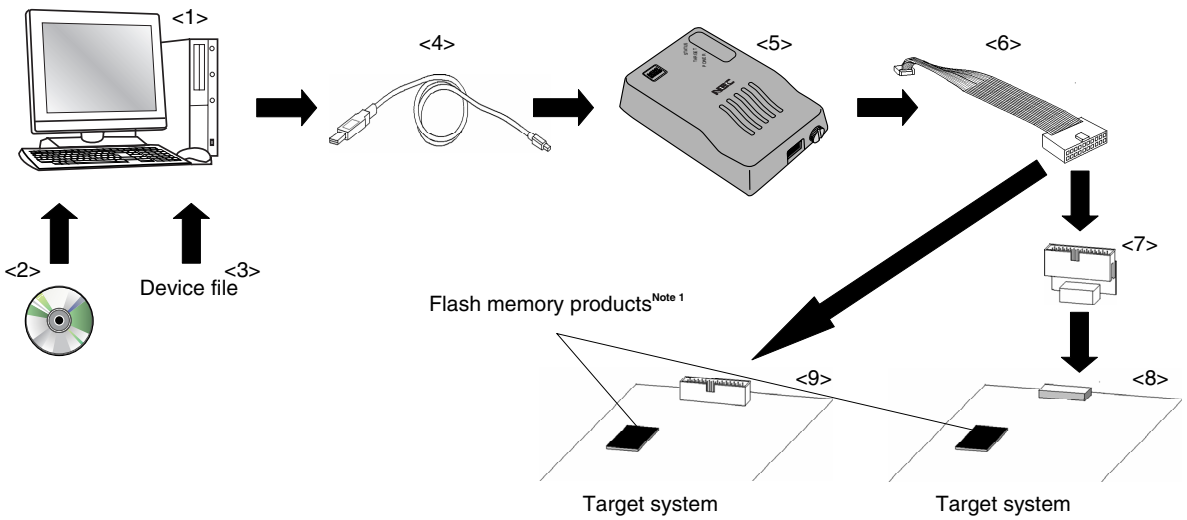
2. Product of KEL Corporation

Remark The numbers in the table correspond to the numbers in Figure A-4.

<R> **A.4.4 When using MINICUBE QB-V850MINI**

The system configuration when connecting the QB-V850MINI to the host machine (PC-9821 series, PC/AT compatible) is shown below.

Figure A-5. System Configuration (QB-V850MINI Used)



<1> Host machine (with on-chip USB port)
<2> ID850QB Disk (Software tools for debugging are packaged.)
<3> Device file Note 2
<4> USB interface cable (supplied with <5>)
<5> On-chip debug emulator (QB-V850MINI)
<6> OCD cable (supplied with <5>)
<7> KEL adapter (supplied with <5>) Notes 3, 4
<8> KEL connector (supplied with <5>) Notes 3, 4
<9> 2.54 mm pitch 20-pin general-purpose connector (sold separately) Note 4

Notes 1. No flash memory versions are provided in the V850ES/SG1. However, the μPD70F3261Y, 70F3271Y and 70F3281Y of the V850ES/SG2 can be used as the flash memory versions for developing V850ES/SG1 software.

2. Obtain the device file from the NEC Electronics website.
http://www.necel.com/micro/ods/eng/index.html

3. Product of KEL Corporation

4. A connector other than KEL connectors can also be used as the target connector. For details, refer to the **QB-V850MINI User's Manual**.

A.5 Debugging Tools (Software)

<R>	SM+ for V850ES/Sx2 System simulator	This is a system simulator for the V850 microcontrollers. The SM+ are Windows-based softwares. 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 the SM+ 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. The SM850 and SM+ should be used in combination with the device file (sold separately).
<R>		Part number: μS××××SM703289-B
	ID850 Integrated debugger (supporting in-circuit emulator IE-V850ES-G1)	This debugger supports the in-circuit emulators for the V850 microcontrollers. The ID850 and ID850QB are Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. It should be used in combination with the device file (sold separately).
	ID850QB Integrated debugger (supporting in-circuit emulator QB-V850ESSX2)	Part number: μS××××ID703000, μS××××ID703000-GC (ID850)

<R> **Remark** ×××× in the part number differs depending on the OS used.

μS××××SM703289-B
μS××××ID703000
μS××××ID703000-GC

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	

A.6 Embedded Software

RX850, RX850 Pro Real-time OS	The RX850 and RX850 Pro are real-time OSs conforming to μITRON 3.0 specifications. A tool (configurator) for generating multiple information tables is supplied. RX850 Pro has more functions than RX850.
	Part number: μS××××RX703000-ΔΔΔΔ (RX850) μS××××RX703100-ΔΔΔΔ (RX850 Pro)
V850mini-NET (provisional name) (Network library)	This is a network library conforming to RFC. It is a lightweight TCP/IP of compact design, requiring only a small memory. In addition to the TCP/IP standard set, an HTTP server, SMTP client, and POP client are also supported.
RX-FS850 (File system)	This is a FAT file system function. It is a file system that supports the CD-ROM file system function. This file system is used with the real-time OS RX850 Pro.

Caution To purchase the RX850 or RX850 Pro, first fill in the purchase application form and sign the user agreement.

Remark ×××× and ΔΔΔΔ in the part number differ depending on the host machine and OS used.

μS××××RX703000-ΔΔΔΔ
μS××××RX703100-ΔΔΔΔ

ΔΔΔΔ	Product Outline	Maximum Number for Use in Mass Production
001	Evaluation object	Do not use for mass-produced product.
100K	Mass-production object	0.1 million units
001M		1 million units
010M		10 million units
S01	Source program	Object source program for mass production

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	
3K17	SPARCstation	Solaris (Rel. 2.5.1)	

A.7 Flash Memory Writing Tools

No flash memory versions are provided in the V850ES/SG1. However, the μPD70F3261Y, 70F3271Y, and
<R> 70F3281Y of the V850ES/SG2 can be used as the flash memory versions for developing V850ES/SG1 software.

For details on the V850ES/SG2 (μPD70F3261Y, μPD70F3271Y, μPD70F3281Y), refer to the **V850ES/SG2, V850ES/SG2-H Hardware User's Manual (U16541E)**.

Flashpro IV (part number: PG-FP4) Flash memory programmer	Flash memory programmer dedicated to microcontrollers with on-chip flash memory.
<R> Flashpro V (part number: PG-FP5) Flash memory programmer	
FA-100GF-3BA-A Flash memory writing adapter	Flash memory writing adapter used connected to Flashpro IV or Flashpro V. • FA-100GF-3BA-A: For 100-pin plastic QFP (GF-JBT type)
FA-100GC-8EU-A Flash memory writing adapter	Flash memory writing adapter used connected to Flashpro IV or Flashpro V. • FA-100GC-8EU-A: For 100-pin plastic LQFP (GC-8EA type)

Remark FA-100GF-3BA-A and FA-100GC-8EU-A are products of Naito Densei Machida Mfg. Co., Ltd.
<R> TEL: +81-42-750-4172 Naito Densei Machida Mfg. Co., Ltd.

<R>

APPENDIX B MAJOR DIFFERENCES BETWEEN V850ES/SG1 (μPD703249Y, 703252Y) AND V850ES/SG1 (μPD703253Y) AND V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)

Differences between the V850ES/SG1 (μPD703249Y, 703252Y) and V850ES/SG1 (μPD703253Y) and V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y) are shown below. For details, refer to each corresponding section. For each part numbers of the V850ES/SG1 (μPD703249Y, 703252Y) and V850ES/SG1 (μPD703253Y) and V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y), refer to remark of **1.1 General**.

Table B-1. Major Differences Between V850ES/SG1 (μPD703249Y, 703252Y) and V850ES/SG1 (μPD703253Y) and V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y) (1/2)

Major Differences		V850ES/SG1	V850ES/SG1	V850ES/SG2	Refer to:
		μPD703249Y, 703252Y	μPD703253Y	μPD70F3261Y, 70F3271Y, 70F3281Y	
Introduction	Package	100-pin plastic QFP (14 × 20) 100-pin plastic LQFP (14 × 14)	100-pin plastic LQFP (14 × 14)	100-pin plastic QFP (14 × 20)[Note] 100-pin plastic LQFP (14 × 14)	1.4
Pin function: P10, P11 I/O circuit type		Type 5	Type 12-D	Type 12-D	2.3
CPU function	Internal ROM	256 KB (mask ROM)	128 KB (mask ROM)	384 KB (flash memory)	3.4.4 (1)
	Internal RAM	12 KB	8 KB	32 KB	3.4.4 (2)
	Programmable peripheral I/O area	None	Provided	Provided (μPD70F3281Y only)	3.4.4 (4)
	Special register	PSC, CKC, PCC, CLM, RESF, OCDM registers	PSC, CKC, PCC, CLM, RESF, RAMS, OCDM registers	PSC, CKC, PCC, CLM, RESF, LVIM, RAMS, OCDM registers	3.4.8
Timer P	Number of channels	5 ch	6 ch	6 ch	Chapter 7
	Selector function	None	None	Provided	
D/A converter		None	Provided	Provided	Chapter 12
UARTA	Number of channel	2 ch	2 ch	3 ch	Chapter 13
	LIN function	None	None	Provided	
CSIB	Number of channel	3 ch	3 ch	5 ch	Chapter 14
	Dedicated BRG	1 ch	1 ch	3 ch	
I^2C: number of channel		2 ch	2 ch	3 ch	Chapter 15
IEBus controller		Provided (μPD703252Y only)	None	Provided (μPD70F3271Y only)	Chapter 16
CAN controller	1 ch	None	Provided	Provided (μPD70F3281Y only)	Chapter 17
	Time stamp function	None	None	Provided	
DMA function		None	Provided	Provided	Chapter 18
Interrupt: maskable interrupt number		31/35 factors	42 factors	47/51 factors	19.1

Note μPD70F3261Y, 70F3271Y only

Table B-1. Major Differences Between V850ES/SG1 (μPD703249Y, 703252Y) and V850ES/SG1 (μPD703253Y) and V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y) (2/2)

Major Differences		V850ES/SG1	V850ES/SG1	V850ES/SG2	Refer to:
		μPD703249Y, 703252Y	μPD703253Y	μPD70F3261Y, 70F3271Y, 70F3281Y	
Reset function	RESF.LVIRF bit	None	None	Provided	22.2 (1)
	Reset by low-voltage detector	None	None	Provided	–
	Valid/invalid of internal RAM data	None	Provided	Provided	22.4
Electrical specifications	Operating conditions	V_{DD} = 2.85 to 3.6 V: A/D converter stopped V_{DD} = 3.0 to 3.6 V: A/D converter operation	V_{DD} = 2.85 to 3.6 V: A/D converter stopped, D/A converter stopped V_{DD} = 3.0 to 3.6 V: A/D converter operation, D/A converter operation	V_{DD} = 2.85 to 3.6 V: A/D converter stopped, D/A converter stopped V_{DD} = 3.0 to 3.6 V: A/D converter operation, D/A converter operation	26.3
	DC specifications (supply current)	Spec differs.			26.6.2
	IEBus controller	Provided (μPD703252Y only)	None	Provided (μPD70F3271Y only)	26.9 (8)
	CAN timing	None	Provided	Provided (μPD70F3281Y only)	26.9 (9)
	DIA converter	None	Provided	Provided	26.9 (11)
	RAM retention detection	None	Provided	Provided	26.9 (12)
Package drawing		S100GF-65-JBT-2 S100GC-50-8EU, 8EA-2	S100GC-50-8EU, 8EA-2	S100GF-65-JBT-2[Note] S100GC-50-8EU, 8EA-2	Chapter 27

Note μPD70F3261Y, 70F3271Y only

APPENDIX C REGISTER INDEX

(1/10)

Symbol	Name	Unit	Page
ADA0CR0	A/D conversion result register 0	ADC	354
ADA0CR0H	A/D conversion result register 0H	ADC	354
ADA0CR1	A/D conversion result register 1	ADC	354
ADA0CR1H	A/D conversion result register 1H	ADC	354
ADA0CR10	A/D conversion result register 10	ADC	354
ADA0CR10H	A/D conversion result register 10H	ADC	354
ADA0CR11	A/D conversion result register 11	ADC	354
ADA0CR11H	A/D conversion result register 11H	ADC	354
ADA0CR1H	A/D conversion result register 1H	ADC	354
ADA0CR2	A/D conversion result register 2	ADC	354
ADA0CR2H	A/D conversion result register 2H	ADC	354
ADA0CR3	A/D conversion result register 3	ADC	354
ADA0CR3H	A/D conversion result register 3H	ADC	354
ADA0CR4	A/D conversion result register 4	ADC	354
ADA0CR4H	A/D conversion result register 4H	ADC	354
ADA0CR5	A/D conversion result register 5	ADC	354
ADA0CR5H	A/D conversion result register 5H	ADC	354
ADA0CR6	A/D conversion result register 6	ADC	354
ADA0CR6H	A/D conversion result register 6H	ADC	354
ADA0CR7	A/D conversion result register 7	ADC	354
ADA0CR7H	A/D conversion result register 7H	ADC	354
ADA0CR8	A/D conversion result register 8	ADC	354
ADA0CR8H	A/D conversion result register 8H	ADC	354
ADA0CR9	A/D conversion result register 9	ADC	354
ADA0CR9H	A/D conversion result register 9H	ADC	354
ADA0M0	A/D converter mode register 0	ADC	347
ADA0M1	A/D converter mode register 1	ADC	349
ADA0M2	A/D converter mode register 2	ADC	352
ADA0PFM	Power fail compare mode register	ADC	356
ADA0PFT	Power fail compare threshold value register	ADC	357
ADA0S	A/D converter channel specification register	ADC	353
ADIC	Interrupt control register	INTC	795
AWC	Address wait control register	BCU	196
BCC	Bus cycle control register	BCU	197
BCR	IEBus control register	IEBus	561
BPC	Peripheral I/O area select control register	BCU	79
BSC	Bus size configuration register	BCU	185
C0BRP	CAN0 module bit rate prescaler register	CAN	677
C0BTR	CAN0 module bit rate register	CAN	678
C0CTRL	CAN0 module control register	CAN	667

(2/10)

Symbol	Name	Unit	Page
C0ERC	CAN0 module error counter register	CAN	673
C0GMABT	CAN0 global automatic block transmission control register	CAN	662
C0GMABTD	CAN0 global automatic block transmission delay setting register	CAN	664
C0GMCS	CAN0 global clock selection register	CAN	661
C0GMCTRL	CAN0 global control register	CAN	659
C0IE	CAN0 module interrupt enable register	CAN	674
C0INFO	CAN0 module information register	CAN	672
C0INTS	CAN0 module interrupt status register	CAN	676
C0LEC	CAN0 module last error information register	CAN	671
C0LIPT	CAN0 module last in-pointer register	CAN	680
C0LOPT	CAN0 module last out-pointer register	CAN	682
C0MASK1H	CAN0 module mask 1 register H	CAN	665
C0MASK1L	CAN0 module mask 1 register L	CAN	665
C0MASK2H	CAN0 module mask 2 register H	CAN	665
C0MASK2L	CAN0 module mask 2 register L	CAN	665
C0MASK3H	CAN0 module mask 3 register H	CAN	665
C0MASK3L	CAN0 module mask 3 register L	CAN	665
C0MASK4H	CAN0 module mask 4 register H	CAN	665
C0MASK4L	CAN0 module mask 4 register L	CAN	665
C0MCONFm	CAN0 message configuration register m	CAN	687
C0MCTRLm	CAN0 message control register m	CAN	689
C0MDATA01m	CAN0 massage data byte 01 register m	CAN	684
C0MDATA0m	CAN0 massage data byte 0 register m	CAN	684
C0MDATA1m	CAN0 massage data byte 1 register m	CAN	684
C0MDATA23m	CAN0 massage data byte 23 register m	CAN	684
C0MDATA2m	CAN0 massage data byte 2 register m	CAN	684
C0MDATA3m	CAN0 massage data byte 3 register m	CAN	684
C0MDATA45m	CAN0 massage data byte 45 register m	CAN	684
C0MDATA4m	CAN0 massage data byte 4 register m	CAN	684
C0MDATA5m	CAN0 massage data byte 5 register m	CAN	684
C0MDATA67m	CAN0 massage data byte 67 register m	CAN	684
C0MDATA6m	CAN0 massage data byte 6 register m	CAN	684
C0MDATA7m	CAN0 massage data byte 7 register m	CAN	684
C0MDLCm	CAN0 message data length register m	CAN	686
C0MIDHm	CAN0 message ID register mH	CAN	688
C0MIDLm	CAN0 message ID register mL	CAN	688
C0RGPT	CAN0 module receive history list register	CAN	681
C0TGPT	CAN0 module transmit history list register	CAN	683
CB0CTL0	CSIB0 control register 0	CSIB	418
CB0CTL1	CSIB0 control register 1	CSIB	422
CB0CTL2	CSIB0 control register 2	CSIB	423
CB0RIC	Interrupt control register	INTC	795
CB0RX	CSIB0 receive data register	CSIB	417

Remark m = 00 to 31

(3/10)

Symbol	Name	Unit	Page
CB0RXL	CSIB0 receive data register L	CSIB	417
CB0STR	CSIB0 status register	CSIB	425
CB0TIC	Interrupt control register	INTC	795
CB0TX	CSIB0 transmit data register	CSI	417
CB0TXL	CSIB0 transmit data register L	CSI	417
CB1CTL0	CSIB1 control register 0	CSI	418
CB1CTL1	CSIB1 control register 1	CSI	422
CB1CTL2	CSIB1 control register 2	CSI	423
CB1RIC	Interrupt control register	INTC	795
CB1RX	CSIB1 receive data register	CSI	417
CB1RXL	CSIB1 receive data register L	CSI	417
CB1STR	CSIB1 status register	CSI	425
CB1TIC	Interrupt control register	INTC	795
CB1TX	CSIB1 transmit data register	CSI	417
CB1TXL	CSIB1 transmit data register L	CSI	417
CB3CTL0	CSIB3 control register 0	CSI	418
CB3CTL1	CSIB3 control register 1	CSI	422
CB3CTL2	CSIB3 control register 2	CSI	423
CB3RIC	Interrupt control register	INTC	795
CB3RX	CSIB3 receive data register	CSI	417
CB3RXL	CSIB3 receive data register L	CSI	417
CB3STR	CSIB3 status register	CSI	425
CB3TIC	Interrupt control register	INTC	795
CB3TX	CSIB3 transmit data register	CSI	417
CB3TXL	CSIB3 transmit data register L	CSI	417
CCLS	CPU operation clock status register	CG	214
CCR	IEBus communication count register	IEBus	589
CDR	IEBus control data register	IEBus	580
CKC	Clock control register	CG	217
CLM	Clock monitor mode register	CLM	852
CORAD0	Correction address register 0	ROMC	859
CORAD0H	Correction address register 0H	ROMC	859
CORAD0L	Correction address register 0L	ROMC	859
CORAD1	Correction address register 1	ROMC	859
CORAD1H	Correction address register 1H	ROMC	859
CORAD1L	Correction address register 1L	ROMC	859
CORAD2	Correction address register 2	ROMC	859
CORAD2H	Correction address register 2H	ROMC	859
CORAD2L	Correction address register 2L	ROMC	859
CORAD3	Correction address register 3	ROMC	859
CORAD3H	Correction address register 3H	ROMC	859
CORAD3L	Correction address register 3L	ROMC	859
CORCN	Correction control register	ROMC	860
CTBP	CALLT base pointer	CPU	56
CTPC	CALLT execution status saving register	CPU	55

(4/10)

Symbol	Name	Unit	Page
CTPSW	CALLT execution status saving register	CPU	55
DA0CS0	D/A converter conversion value setting register 0	DAC	382
DA0CS1	D/A converter conversion value setting register 1	DAC	382
DA0M	D/A converter mode register	DAC	381
DADC0	DMA addressing control register 0	DMAC	760
DADC1	DMA addressing control register 1	DMAC	760
DADC2	DMA addressing control register 2	DMAC	760
DADC3	DMA addressing control register 3	DMAC	760
DBC0	DMA transfer count register 0	DMAC	759
DBC1	DMA transfer count register 1	DMAC	759
DBC2	DMA transfer count register 2	DMAC	759
DBC3	DMA transfer count register 3	DMAC	759
DBPC	Exception/debug trap status saving register	CPU	56
DBPSW	Exception/debug trap status saving register	CPU	56
DCHC0	DMA channel control register 0	DMAC	761
DCHC1	DMA channel control register 1	DMAC	761
DCHC2	DMA channel control register 2	DMAC	761
DCHC3	DMA channel control register 3	DMAC	761
DDA0H	DMA destination address register 0H	DMAC	758
DDA0L	DMA destination address register 0L	DMAC	758
DDA1H	DMA destination address register 1H	DMAC	758
DDA1L	DMA destination address register 1L	DMAC	758
DDA2H	DMA destination address register 2H	DMAC	758
DDA2L	DMA destination address register 2L	DMAC	758
DDA3H	DMA destination address register 3H	DMAC	758
DDA3L	DMA destination address register 3L	DMAC	758
DLR	IEBus telegraph length register	DLR	585
DMAIC0	Interrupt control register	INTC	795
DMAIC1	Interrupt control register	INTC	795
DMAIC2	Interrupt control register	INTC	795
DMAIC3	Interrupt control register	INTC	795
DR	IEBus data register	IEBus	586
DSA0H	DMA source address register 0H	DMAC	757
DSA0L	DMA source address register 0L	DMAC	757
DSA1H	DMA source address register 1H	DMAC	757
DSA1L	DMA source address register 1L	DMAC	757
DSA2H	DMA source address register 2H	DMAC	757
DSA2L	DMA source address register 2L	DMAC	757
DSA3H	DMA source address register 3H	DMAC	757
DSA3L	DMA source address register 3L	DMAC	757
DTFR0	DMA trigger factor register 0	DMAC	762
DTFR1	DMA trigger factor register 1	DMAC	762
DTFR2	DMA trigger factor register 2	DMAC	762
DTFR3	DMA trigger factor register 3	DMAC	762
DWC0	Data wait control register 0	BCU	193

(5/10)

Symbol	Name	Unit	Page
ECR	Interrupt source register	CPU	53
EIPC	Interrupt status saving register	CPU	52
EIPSW	Interrupt status saving register	CPU	52
ERRIC0	Interrupt control register	INTC	795
ERRIC	Interrupt control register	INTC	795
ESR	IEBus error status register	IEBus	573
EXIMC	External bus interface mode control register	BCU	184
FEPC	NMI status saving register	CPU	53
FEPSW	NMI status saving register	CPU	53
FSR	IEBus field status register	IEBus	587
IEIC1	Interrupt control register	INTC	795
IEIC2	Interrupt control register	INTC	795
IIC0	IIC shift register 0	I^2C	483
IIC1	IIC shift register 1	I^2C	483
IICC0	IIC control register 0	I^2C	470
IICC1	IIC control register 1	I^2C	470
IICCL0	IIC clock select register 0	I^2C	480
IICCL1	IIC clock select register 1	I^2C	480
IICF0	IIC flag register 0	I^2C	478
IICF1	IIC flag register 1	I^2C	478
IICIC0	Interrupt control register	INTC	795
IICIC1	Interrupt control register	INTC	795
IICS0	IIC status register 0	I^2C	475
IICS1	IIC status register 1	I^2C	475
IICX0	IIC function expansion register 0	I^2C	481
IICX1	IIC function expansion register 1	I^2C	481
IMR0	Interrupt mask register 0	INTC	797
IMR0H	Interrupt mask register 0H	INTC	797
IMR0L	Interrupt mask register 0L	INTC	797
IMR1	Interrupt mask register 1	INTC	797
IMR1H	Interrupt mask register 1H	INTC	797
IMR1L	Interrupt mask register 1L	INTC	797
IMR2	Interrupt mask register 2	INTC	797
IMR2H	Interrupt mask register 2H	INTC	797
IMR2L	Interrupt mask register 2L	INTC	797
IMR3	Interrupt mask register 3	INTC	797
IMR3H	Interrupt mask register 3H	INTC	797
IMR3L	Interrupt mask register 3L	INTC	797
INTF0	External interrupt falling edge specification register 0	INTC	809
INTF3	External interrupt falling edge specification register 3	INTC	810
INTF9H	External interrupt falling edge specification register 9H	INTC	811
INTR0	External interrupt rising edge specification register 0	INTC	809
INTR3	External interrupt rising edge specification register 3	INTC	810
INTR9H	External interrupt rising edge specification register 9H	INTC	811
ISPR	In-service priority register	INTC	799

(6/10)

Symbol	Name	Unit	Page
ISR	IEBus interrupt status register	IEBus	570
KRIC	Interrupt control register	INTC	795
KRM	Key return mode register	KR	817
LOCKR	Lock register	CG	218
NFC	Noise elimination control register	INTC	812
OCDM	On-chip debug mode register	Debug	179
OCKS0	IIC division clock select register 0	I^2C	483
OCKS1	IIC division clock select register 1	I^2C	483
OCKS2	IEBus clock select register	IEBus	590
OSTS	Oscillation stabilization time select register	Standby	822
P0	Port 0 register	Port	95
P1	Port 1 register	Port	98
P3	Port 3 register	Port	100
P3H	Port 3 register H	Port	100
P3L	Port 3 register L	Port	100
P4	Port 4 register	Port	106
P5	Port 5 register	Port	109
P7H	Port 7 register H	Port	114
P7L	Port 7 register L	Port	114
P9	Port 9 register	Port	116
P9H	Port 9 register H	Port	116
P9L	Port 9 register L	Port	116
PAR	IEBus partner address register	IEBus	579
PC	Program counter	CPU	50
PCC	Processor clock control register	CG	210
PCM	Port CM register	Port	123
PCT	Port CT register	Port	125
PDH	Port DH register	Port	127
PDL	Port DL register	Port	130
PDLH	Port DL register H	Port	130
PDLL	Port DL register L	Port	130
PEMU1	Peripheral emulation register 1	CPU	850
PF0	Port 0 function control register	Port	97
PF3	Port 3 function control register	Port	105
PF3H	Port 3 function control register H	Port	105
PF3L	Port 3 function control register L	Port	105
PF4	Port 4 function control register	Port	108
PF5	Port 5 function control register	Port	112
PF9	Port 9 function control register	Port	122
PF9H	Port 9 function control register H	Port	122
PF9L	Port 9 function control register L	Port	122
PFC0	Port 0 function control register	Port	97
PFC3	Port 3 function control register	Port	103
PFC3H	Port 3 function control register H	Port	103
PFC3L	Port 3 function control register L	Port	103

(7/10)

Symbol	Name	Unit	Page
PFC4	Port 4 function control register	Port	107
PFC5	Port 5 function control register	Port	111
PFC9	Port 9 function control register	Port	119
PFC9H	Port 9 function control register H	Port	119
PFC9L	Port 9 function control register L	Port	119
PFCE3L	Port 3 function control expansion register L	Port	103
PFCE5	Port 5 function control expansion register	Port	111
PFCE9	Port 9 function control expansion register	Port	119
PFCE9H	Port 9 function control expansion register H	Port	119
PFCE9L	Port 9 function control expansion register L	Port	119
PIC0	Interrupt control register	INTC	795
PIC1	Interrupt control register	INTC	795
PIC2	Interrupt control register	INTC	795
PIC3	Interrupt control register	INTC	795
PIC4	Interrupt control register	INTC	795
PIC5	Interrupt control register	INTC	795
PIC6	Interrupt control register	INTC	795
PIC7	Interrupt control register	INTC	795
PLLCTL	PLL control register	CG	216
PLLS	PLL lockup time specification register	CG	219
PM0	Port 0 mode register	Port	96
PM1	Port 1 mode register	Port	99
PM3	Port 3 mode register	Port	101
PM3H	Port 3 mode register H	Port	101
PM3L	Port 3 mode register L	Port	101
PM4	Port 4 mode register	Port	106
PM5	Port 5 mode register	Port	110
PM7H	Port 7 mode register H	Port	114
PM7L	Port 7 mode register L	Port	114
PM9	Port 9 mode register	Port	116
PM9H	Port 9 mode register H	Port	116
PM9L	Port 9 mode register L	Port	116
PMC0	Port 0 mode control register	Port	96
PMC3	Port 3 mode control register	Port	101
PMC3H	Port 3 mode control register H	Port	101
PMC3L	Port 3 mode control register L	Port	101
PMC4	Port 4 mode control register	Port	107
PMC5	Port 5 mode control register	Port	110
PMC9	Port 9 mode control register	Port	117
PMC9H	Port 9 mode control register H	Port	117
PMC9L	Port 9 mode control register L	Port	117
PMCCM	Port CM mode control register	Port	124
PMCCT	Port CT mode control register	Port	126
PMCDH	Port DH mode control register	Port	128
PMCDL	Port DL mode control register	Port	131

(8/10)

Symbol	Name	Unit	Page
PMCDLH	Port DL mode control register H	Port	131
PMCDLL	Port DL mode control register L	Port	131
PMCM	Port CM mode register	Port	123
PMCT	Port CT mode register	Port	125
PMDH	Port DH mode register	Port	127
PMDL	Port DL mode register	Port	130
PMDLH	Port DL mode register H	Port	130
PMDLL	Port DL mode register L	Port	130
PRCMD	Command register	CPU	82
PRSCM0	Prescaler compare register 0	WT	331
PRSCM1	BRG1 prescaler compare register	BRG	462
PRSM0	Prescaler mode register 0	WT	330
PRSM1	BRG1 prescaler mode register	BRG	461
PSC	Power save control register	CG	820
PSMR	Power save mode register	CG	821
PSR	IEBus power save register	IEBus	565
PSW	Program status word	CPU	54
r0 to r31	General-purpose registers	CPU	50
RAMS	Internal RAM data status register	CG	848
RCM	Internal oscillation mode register	CG	214
RECIC0	Interrupt control register	INTC	795
RESF	Reset source flag register	Reset	839
RSA	IEBus receive slave address register	IEBus	579
SAR	IEBus slave address register	IEBus	578
SCR	IEBus success count register	IEBus	588
SSR	IEBus slave status register	IEBus	566
STAIC	Interrupt control register	INTC	795
SVA0	Slave address register 0	I^2C	484
SVA1	Slave address register 1	I^2C	484
SYS	System status register 0	CPU	83
TM0CMP0	TMM0 compare register 0	Timer	320
TM0CTL0	TMM0 control register 0	Timer	321
TM0EQIC0	Interrupt control register	INTC	795
TP0CCIC0	Interrupt control register	INTC	795
TP0CCIC1	Interrupt control register	INTC	795
TP0CCR0	TMP0 capture/compare register 0	Timer	232
TP0CCR1	TMP0 capture/compare register 1	Timer	234
TP0CNT	TMP0 counter read buffer register	Timer	236
TP0CTL0	TMP0 control register 0	Timer	225
TP0CTL1	TMP0 control register 1	Timer	225
TP0IOC0	TMP0 I/O control register 0	Timer	227
TP0IOC1	TMP0 I/O control register 1	Timer	229
TP0IOC2	TMP0 I/O control register 2	Timer	230
TP0OPT0	TMP0 option register 0	Timer	231
TP0OVIC	Interrupt control register	INTC	795

(9/10)

Symbol	Name	Unit	Page
TP1CCIC0	Interrupt control register	INTC	795
TP1CCIC1	Interrupt control register	INTC	795
TP1CCR0	TMP1 capture/compare register 0	Timer	232
TP1CCR1	TMP1 capture/compare register 1	Timer	234
TP1CNT	TMP1 counter read buffer register	Timer	236
TP1CTL0	TMP1 control register 0	Timer	225
TP1CTL1	TMP1 control register 1	Timer	225
TP1IOC0	TMP1 I/O control register 0	Timer	227
TP1IOC1	TMP1 I/O control register 1	Timer	229
TP1IOC2	TMP1 I/O control register 2	Timer	230
TP1OPT0	TMP1 option register 0	Timer	231
TP1OVIC	Interrupt control register	INTC	795
TP2CCIC0	Interrupt control register	INTC	795
TP2CCIC1	Interrupt control register	INTC	795
TP2CCR0	TMP2 capture/compare register 0	Timer	232
TP2CCR1	TMP2 capture/compare register 1	Timer	234
TP2CNT	TMP2 counter read buffer register	Timer	236
TP2CTL0	TMP2 control register 0	Timer	225
TP2CTL1	TMP2 control register 1	Timer	225
TP2IOC0	TMP2 I/O control register 0	Timer	227
TP2IOC1	TMP2 I/O control register 1	Timer	229
TP2IOC2	TMP2 I/O control register 2	Timer	230
TP2OPT0	TMP2 option register 0	Timer	231
TP2OVIC	Interrupt control register	INTC	795
TP3CCIC0	Interrupt control register	INTC	795
TP3CCIC1	Interrupt control register	INTC	795
TP3CCR0	TMP3 capture/compare register 0	Timer	232
TP3CCR1	TMP3 capture/compare register 1	Timer	234
TP3CNT	TMP3 counter read buffer register	Timer	236
TP3CTL0	TMP3 control register 0	Timer	225
TP3CTL1	TMP3 control register 1	Timer	225
TP3IOC0	TMP3 I/O control register 0	Timer	227
TP3IOC1	TMP3 I/O control register 1	Timer	229
TP3IOC2	TMP3 I/O control register 2	Timer	230
TP3OPT0	TMP3 option register 0	Timer	231
TP3OVIC	Interrupt control register	INTC	795
TP4CCIC0	Interrupt control register	INTC	795
TP4CCIC1	Interrupt control register	INTC	795
TP4CCR0	TMP4 capture/compare register 0	Timer	232
TP4CCR1	TMP4 capture/compare register 1	Timer	234
TP4CNT	TMP4 counter read buffer register	Timer	236
TP4CTL0	TMP4 control register 0	Timer	225
TP4CTL1	TMP4 control register 1	Timer	225
TP4IOC0	TMP4 I/O control register 0	Timer	227
TP4IOC1	TMP4 I/O control register 1	Timer	229

(10/10)

Symbol	Name	Unit	Page
TP4IOC2	TMP4 I/O control register 2	Timer	230
TP4OPT0	TMP4 option register 0	Timer	231
TP4OVIC	Interrupt control register	INTC	795
TP5CCIC0	Interrupt control register	INTC	795
TP5CCIC1	Interrupt control register	INTC	795
TP5CCR0	TMP5 capture/compare register 0	Timer	232
TP5CCR1	TMP5 capture/compare register 1	Timer	234
TP5CNT	TMP5 counter read buffer register	Timer	236
TP5CTL0	TMP5 control register 0	Timer	225
TP5CTL1	TMP5 control register 1	Timer	225
TP5IOC0	TMP5 I/O control register 0	Timer	227
TP5IOC1	TMP5 I/O control register 1	Timer	229
TP5IOC2	TMP5 I/O control register 2	Timer	230
TP5OPT0	TMP5 option register 0	Timer	231
TP5OVIC	Interrupt control register	INTC	795
TRXIC0	Interrupt control register	INTC	795
UA0CTL0	UARTA0 control register 0	UARTA	388
UA0CTL1	UARTA0 control register 1	UARTA	390
UA0CTL2	UARTA0 control register 2	UARTA	390
UA0OPT0	UARTA0 option control register 0	UARTA	390
UA0RIC	Interrupt control register	INTC	795
UA0RX	UARTA0 receive data register	UARTA	393
UA0STR	UARTA0 status register	UARTA	391
UA0TIC	Interrupt control register	INTC	795
UA0TX	UARTA0 transmit data register	UARTA	393
UA1CTL0	UARTA1 control register 0	UARTA	388
UA1CTL1	UARTA1 control register 1	UARTA	390, 406
UA1CTL2	UARTA1 control register 2	UARTA	390, 407
UA1OPT0	UARTA1 option control register 0	UARTA	390
UA1RIC	Interrupt control register	INTC	795
UA1RX	UARTA1 receive data register	UARTA	391
UA1STR	UARTA1 status register	UARTA	391
UA1TIC	Interrupt control register	INTC	795
UA1TX	UARTA1 transmit data register	UARTA	393
UAR	IEBus unit address register	IEBus	578
USR	IEBus unit status register	IEBus	567
VSWC	System wait control register	CPU	84
WDTE	Watchdog timer enable register	WDT	341
WDTM2	Watchdog timer mode register 2	WDT	339, 800
WTIC	Interrupt control register	INTC	795
WTIIC	Interrupt control register	INTC	795
WTM	Watch timer operation mode register	WT	332
WUPIC0	Interrupt control register	INTC	795

APPENDIX D INSTRUCTION SET LIST

D.1 Conventions

(1) Register symbols used to describe operands

Register Symbol	Explanation
reg1	General-purpose registers: Used as source registers.
reg2	General-purpose registers: Used mainly as destination registers. Also used as source register in some instructions.
reg3	General-purpose registers: Used mainly to store the remainders of division results and the higher 32 bits of multiplication results.
bit#3	3-bit data for specifying the bit number
immX	X bit immediate data
dispX	X bit displacement data
regID	System register number
vector	5-bit data that specifies the trap vector (00H to 1FH)
cccc	4-bit data that shows the conditions code
sp	Stack pointer (r3)
ep	Element pointer (r30)
listX	X item register list

(2) Register symbols used to describe opcodes

Register Symbol	Explanation
R	1-bit data of a code that specifies reg1 or regID
r	1-bit data of the code that specifies reg2
w	1-bit data of the code that specifies reg3
d	1-bit displacement data
I	1-bit immediate data (indicates the higher bits of immediate data)
i	1-bit immediate data
cccc	4-bit data that shows the condition codes
CCCC	4-bit data that shows the condition codes of Bcond instruction
bbb	3-bit data for specifying the bit number
L	1-bit data that specifies a program register in the register list

(3) Register symbols used in operations

Register Symbol	Explanation
←	Input for
GR [ ]	General-purpose register
SR [ ]	System register
zero-extend (n)	Expand n with zeros until word length.
sign-extend (n)	Expand n with signs until word length.
load-memory (a, b)	Read size b data from address a.
store-memory (a, b, c)	Write data b into address a in size c.
load-memory-bit (a, b)	Read bit b of address a.
store-memory-bit (a, b, c)	Write c to bit b of address a.
saturated (n)	Execute saturated processing of n (n is a 2's complement). If, as a result of calculations, n ≥ 7FFFFFFFH, let it be 7FFFFFFFH. n ≤ 80000000H, let it be 80000000H.
result	Reflects the results in a flag.
Byte	Byte (8 bits)
Halfword	Half word (16 bits)
Word	Word (32 bits)
+	Addition
–	Subtraction
ll	Bit concatenation
×	Multiplication
÷	Division
%	Remainder from division results
AND	Logical product
OR	Logical sum
XOR	Exclusive OR
NOT	Logical negation
logically shift left by	Logical shift left
logically shift right by	Logical shift right
arithmetically shift right by	Arithmetic shift right

(4) Register symbols used in execution clock

Register Symbol	Explanation
i	If executing another instruction immediately after executing the first instruction (issue).
r	If repeating execution of the same instruction immediately after executing the first instruction (repeat).
l	If using the results of instruction execution in the instruction immediately after the execution (latency).

(5) Register symbols used in flag operations

Identifier	Explanation
(Blank)	No change
0	Clear to 0
X	Set or cleared in accordance with the results.
R	Previously saved values are restored.

(6) Condition codes

Condition Code (cccc)	Condition Formula	Explanation
0 0 0 0	OV = 1	Overflow
1 0 0 0	OV = 0	No overflow
0 0 0 1	CY = 1	Carry Lower (Less than)
1 0 0 1	CY = 0	No carry Not lower (Greater than or equal)
0 0 1 0	Z = 1	Zero
1 0 1 0	Z = 0	Not zero
0 0 1 1	(CY or Z) = 1	Not higher (Less than or equal)
1 0 1 1	(CY or Z) = 0	Higher (Greater than)
0 1 0 0	S = 1	Negative
1 1 0 0	S = 0	Positive
0 1 0 1	–	Always (Unconditional)
1 1 0 1	SAT = 1	Saturated
0 1 1 0	(S xor OV) = 1	Less than signed
1 1 1 0	(S xor OV) = 0	Greater than or equal signed
0 1 1 1	((S xor OV) or Z) = 1	Less than or equal signed
1 1 1 1	((S xor OV) or Z) = 0	Greater than signed

D.2 Instruction Set (in Alphabetical Order)

(1/6)

Mnemonic	Operand	Opcode	Operation		Execution Clock			Flags				
					i	r	l	CY	OV	S	Z	SAT
ADD	reg1,reg2	rrrrr001110RRRRR	GR[reg2]←GR[reg2]+GR[reg1]		1	1	1	×	×	×	×	
	imm5,reg2	rrrrr010010iiiii	GR[reg2]←GR[reg2]+sign-extend(imm5)		1	1	1	×	×	×	×	
ADDI	imm16,reg1,reg2	rrrrr110000RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+sign-extend(imm16)		1	1	1	×	×	×	×	
AND	reg1,reg2	rrrrr001010RRRRR	GR[reg2]←GR[reg2]AND GR[reg1]		1	1	1		0	×	×	
ANDI	imm16,reg1,reg2	rrrrr110110RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]AND zero-extend(imm16)		1	1	1		0	×	×	
Bcond	disp9	ddddd1011dddcccc **Note 1**	if conditions are satisfied then PC←PC+sign-extend(disp9)	When conditions are satisfied	2 Note 2	2 Note 2	2 Note 2					
				When conditions are not satisfied	1	1	1					
BSH	reg2,reg3	rrrrr11111100000 wwwww01101000010	GR[reg3]←GR[reg2] (23 : 16) ll GR[reg2] (31 : 24) ll GR[reg2] (7 : 0) ll GR[reg2] (15 : 8)		1	1	1	×	0	×	×	
BSW	reg2,reg3	rrrrr11111100000 wwwww01101000000	GR[reg3]←GR[reg2] (7 : 0) ll GR[reg2] (15 : 8) ll GR [reg2] (23 : 16) ll GR[reg2] (31 : 24)		1	1	1	×	0	×	×	
CALLT	imm6	0000001000iiiiii	CTPC←PC+2(return PC) CTPSW←PSW adr←CTBP+zero-extend(imm6 logically shift left by 1) PC←CTBP+zero-extend(Load-memory(adr,Halfword))		4	4	4					
CLR1	bit#3,disp16[reg1]	10bbb111110RRRRR dddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,0)		3 Note 3	3 Note 3	3 Note 3				×	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100100	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,0)		3 Note 3	3 Note 3	3 Note 3				×	
CMOV	cccc,imm5,reg2,reg3	rrrrr111111iiiii wwwww011000cccc0	if conditions are satisfied then GR[reg3]←sign-extended(imm5) else GR[reg3]←GR[reg2]		1	1	1					
	cccc,reg1,reg2,reg3	rrrrr111111RRRR wwwww011001cccc0	if conditions are satisfied then GR[reg3]←GR[reg1] else GR[reg3]←GR[reg2]		1	1	1					
CMP	reg1,reg2	rrrrr001111RRRRR	result←GR[reg2]–GR[reg1]		1	1	1	×	×	×	×	
	imm5,reg2	rrrrr010011iiiii	result←GR[reg2]–sign-extend(imm5)		1	1	1	×	×	×	×	
CTRET		0000011111100000 0000000101000100	PC←CTPC PSW←CTPSW		3	3	3	R	R	R	R	R
DBRET		0000011111100000 0000000101000110	PC←DBPC PSW←DBPSW		3	3	3	R	R	R	R	R

(2/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
DBTRAP		1111100001000000	DBPC←PC+2 (restored PC) DBPSW←PSW PSW.NP←1 PSW.EP←1 PSW.ID←1 PC←00000060H	3	3	3					
DI		0000011111100000 0000000101100000	PSW.ID←1	1	1	1					
DISPOSE	imm5,list12	0000011001iiiiiL LLLLLLLLLLL00000	sp←sp+zero-extend(imm5 logically shift left by 2) GR[reg in list12]←Load-memory(sp,Word) sp←sp+4 repeat 2 steps above until all regs in list12 is loaded	n+1 Note 4	n+1 Note 4	n+1 Note 4					
	imm5,list12,[reg1]	0000011001iiiiiL LLLLLLLLLLRRRRR **Note 5**	sp←sp+zero-extend(imm5 logically shift left by 2) GR[reg in list12]←Load-memory(sp,Word) sp←sp+4 repeat 2 steps above until all regs in list12 is loaded PC←GR[reg1]	n+3 Note 4	n+3 Note 4	n+3 Note 4					
DIV	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01011000000	GR[reg2]←GR[reg2]÷GR[reg1] GR[reg3]←GR[reg2]%GR[reg1]	35	35	35		×	×	×	
DIVH	reg1,reg2	rrrrr000010RRRRR	GR[reg2]←GR[reg2]÷GR[reg1]Note 6	35	35	35		×	×	×	
	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01010000000	GR[reg2]←GR[reg2]÷GR[reg1]Note 6 GR[reg3]←GR[reg2]%GR[reg1]	35	35	35		×	×	×	
DIVHU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01010000010	GR[reg2]←GR[reg2]÷GR[reg1]Note 6 GR[reg3]←GR[reg2]%GR[reg1]	34	34	34		×	×	×	
DIVU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01011000010	GR[reg2]←GR[reg2]÷GR[reg1] GR[reg3]←GR[reg2]%GR[reg1]	34	34	34		×	×	×	
EI		1000011111100000 0000000101100000	PSW.ID←0	1	1	1					
HALT		0000011111100000 0000000100100000	Stop	1	1	1					
HSW	reg2,reg3	rrrrr11111100000 wwwww01101000100	GR[reg3]←GR[reg2](15 : 0) ll GR[reg2] (31 : 16)	1	1	1	×	0	×	×	
JARL	disp22,reg2	rrrrr11110ddddddd ddddddddddddddd0 **Note 7**	GR[reg2]←PC+4 PC←PC+sign-extend(disp22)	2	2	2					
JMP	[reg1]	00000000011RRRRR	PC←GR[reg1]	3	3	3					
JR	disp22	0000011110ddddddd ddddddddddddddd0 **Note 7**	PC←PC+sign-extend(disp22)	2	2	2					
LD.B	disp16[reg1],reg2	rrrrr111000RRRRR ddddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	Note 11					
LD.BU	disp16[reg1],reg2	rrrrr11110bRRRRR ddddddddddddddd1 **Notes 8, 10**	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←zero-extend(Load-memory(adr,Byte))	1	1	Note 11					

(3/6)

Mnemonic	Operand	Opcode	Operation		Execution Clock			Flags				
					i	r	l	CY	OV	S	Z	SAT
LD.H	disp16[reg1],reg2	rrrrr111001RRRRR ddddddddddddddd0 **Note 8**	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Halfword))		1	1	Note 11					
LDSR	reg2,regID	rrrrr111111RRRRR 0000000000100000 **Note 12**	SR[regID]←GR[reg2]	Other than regID = PSW	1	1	1					
				regID = PSW	1	1	1	×	×	×	×	×
LD.HU	disp16[reg1],reg2	rrrrr111111RRRRR ddddddddddddddd1 **Note 8**	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←zero-extend(Load-memory(adr,Halfword)		1	1	Note 11					
LD.W	disp16[reg1],reg2	rrrrr111001RRRRR ddddddddddddddd1 **Note 8**	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←Load-memory(adr,Word)		1	1	Note 11					
MOV	reg1,reg2	rrrrr000000RRRRR	GR[reg2]←GR[reg1]		1	1	1					
	imm5,reg2	rrrrr010000iiiii	GR[reg2]←sign-extend(imm5)		1	1	1					
	imm32,reg1	00000110001RRRRR iiiiiiiiiiiiiiii llllllllllllllll	GR[reg1]←imm32		2	2	2					
MOVEA	imm16,reg1,reg2	rrrrr110001RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+sign-extend(imm16)		1	1	1					
MOVHI	imm16,reg1,reg2	rrrrr110010RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+(imm16 ll 0^{16})		1	1	1					
MUL	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100000 **Note 14**	GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1]		1	4	5					
	imm9,reg2,reg3	rrrrr111111iiiii wwwww01001IIII00 **Note 13**	GR[reg3] ll GR[reg2]←GR[reg2]xsign-extend(imm9)		1	4	5					
MULH	reg1,reg2	rrrrr000111RRRRR	GR[reg2]←GR[reg2][Note 6]xGR[reg1][Note 6]		1	1	2					
	imm5,reg2	rrrrr010111iiiii	GR[reg2]←GR[reg2][Note 6]xsign-extend(imm5)		1	1	2					
MULHI	imm16,reg1,reg2	rrrrr110111RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1][Note 6]ximm16		1	1	2					
MULU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100010 **Note 14**	GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1]		1	4	5					
	imm9,reg2,reg3	rrrrr111111iiiii wwwww01001IIII10 **Note 13**	GR[reg3] ll GR[reg2]←GR[reg2]xzero-extend(imm9)		1	4	5					
NOP		0000000000000000	Pass at least one clock cycle doing nothing.		1	1	1					
NOT	reg1,reg2	rrrrr000001RRRRR	GR[reg2]←NOT(GR[reg1])		1	1	1		0	×	×	
NOT1	bit#3,disp16[reg1]	01bbb111110RRRRR dddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,Z flag)		3 Note 3	3 Note 3	3 Note 3				×	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100010	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,Z flag)		3 Note 3	3 Note 3	3 Note 3				×	

(4/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
OR	reg1,reg2	rrrrr001000RRRRR	GR[reg2]←GR[reg2]OR GR[reg1]	1	1	1		0	×	×	
ORI	imm16,reg1,reg2	rrrrr110100RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]OR zero-extend(imm16)	1	1	1		0	×	×	
PREPARE	list12,imm5	0000011110iiiiiL LLLLLLLLLLL00001	Store-memory(sp–4,GR[reg in list12],Word) sp←sp–4 repeat 1 step above until all regs in list12 is stored sp←sp-zero-extend(imm5)	n+1 Note 4	n+1 Note 4	n+1 Note 4					
	list12,imm5, sp/imm Note 15	0000011110iiiiiL LLLLLLLLLLLff011 imm16/imm32 **Note 16**	Store-memory(sp–4,GR[reg in list12],Word) sp←sp+4 repeat 1 step above until all regs in list12 is stored sp←sp-zero-extend (imm5) ep←sp/imm	n+2 Note 4 Note 17	n+2 Note 4 Note 17	n+2 Note 4 Note 17					
RETI		0000011111100000 0000000101000000	if PSW.EP=1 then PC ←EIPC PSW ←EIPSW else if PSW.NP=1 then PC ←FEPC PSW ←FEPSW else PC ←EIPC PSW ←EIPSW	3	3	3	R	R	R	R	R
SAR	reg1,reg2	rrrrr111111RRRRR 0000000010100000	GR[reg2]←GR[reg2]arithmetically shift right by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010101iiiii	GR[reg2]←GR[reg2]arithmetically shift right by zero-extend (imm5)	1	1	1	×	0	×	×	
SASF	cccc,reg2	rrrrr1111110cccc 0000001000000000	if conditions are satisfied then GR[reg2]←(GR[reg2]Logically shift left by 1) OR 00000001H else GR[reg2]←(GR[reg2]Logically shift left by 1) OR 00000000H	1	1	1					
SATADD	reg1,reg2	rrrrr000110RRRRR	GR[reg2]←saturated(GR[reg2]+GR[reg1])	1	1	1	×	×	×	×	×
	imm5,reg2	rrrrr010001iiiii	GR[reg2]←saturated(GR[reg2]+sign-extend(imm5)	1	1	1	×	×	×	×	×
SATSUB	reg1,reg2	rrrrr000101RRRRR	GR[reg2]←saturated(GR[reg2]–GR[reg1])	1	1	1	×	×	×	×	×
SATSUBI	imm16,reg1,reg2	rrrrr110011RRRRR iiiiiiiiiiiiiiii	GR[reg2]←saturated(GR[reg1]–sign-extend(imm16)	1	1	1	×	×	×	×	×
SATSUBR	reg1,reg2	rrrrr000100RRRRR	GR[reg2]←saturated(GR[reg1]–GR[reg2])	1	1	1	×	×	×	×	×
SETF	cccc,reg2	rrrrr1111110cccc 0000000000000000	If conditions are satisfied then GR[reg2]←00000001H else GR[reg2]←00000000H	1	1	1					

(5/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
SET1	bit#3,disp16[reg1]	00bbb111110RRRRR dddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not (Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,1)	3 Note 3	3 Note 3	3 Note 3				×	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100000	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,1)	3 Note 3	3 Note 3	3 Note 3				×	
SHL	reg1,reg2	rrrrr111111RRRRR 0000000011000000	GR[reg2]←GR[reg2] logically shift left by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010110iiiii	GR[reg2]←GR[reg2] logically shift left by zero-extend(imm5)	1	1	1	×	0	×	×	
SHR	reg1,reg2	rrrrr111111RRRRR 0000000010000000	GR[reg2]←GR[reg2] logically shift right by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010100iiiii	GR[reg2]←GR[reg2] logically shift right by zero-extend(imm5)	1	1	1	×	0	×	×	
SLD.B	disp7[ep],reg2	rrrrr0110ddddddd	adr←ep+zero-extend(disp7) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	Note 9					
SLD.BU	disp4[ep],reg2	rrrrr0000110dddd **Note 18**	adr←ep+zero-extend(disp4) GR[reg2]←zero-extend(Load-memory(adr,Byte))	1	1	Note 9					
SLD.H	disp8[ep],reg2	rrrrr1000ddddddd **Note 19**	adr←ep+zero-extend(disp8) GR[reg2]←sign-extend(Load-memory(adr,Halfword))	1	1	Note 9					
SLD.HU	disp5[ep],reg2	rrrrr0000111dddd **Notes 18, 20**	adr←ep+zero-extend(disp5) GR[reg2]←zero-extend(Load-memory(adr,Halfword))	1	1	Note 9					
SLD.W	disp8[ep],reg2	rrrrr1010ddddddd0 **Note 21**	adr←ep+zero-extend(disp8) GR[reg2]←Load-memory(adr,Word)	1	1	Note 9					
SST.B	reg2,disp7[ep]	rrrrr0111ddddddd	adr←ep+zero-extend(disp7) Store-memory(adr,GR[reg2],Byte)	1	1	1					
SST.H	reg2,disp8[ep]	rrrrr1001ddddddd **Note 19**	adr←ep+zero-extend(disp8) Store-memory(adr,GR[reg2],Halfword)	1	1	1					
SST.W	reg2,disp8[ep]	rrrrr1010ddddddd1 **Note 21**	adr←ep+zero-extend(disp8) Store-memory(adr,GR[reg2],Word)	1	1	1					
ST.B	reg2,disp16[reg1]	rrrrr111010RRRRR dddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Store-memory(adr,GR[reg2],Byte)	1	1	1					
ST.H	reg2,disp16[reg1]	rrrrr111011RRRRR ddddddddddddddd0 **Note 8**	adr←GR[reg1]+sign-extend(disp16) Store-memory (adr,GR[reg2], Halfword)	1	1	1					
ST.W	reg2,disp16[reg1]	rrrrr111011RRRRR ddddddddddddddd1 **Note 8**	adr←GR[reg1]+sign-extend(disp16) Store-memory (adr,GR[reg2], Word)	1	1	1					
STSR	regID,reg2	rrrrr111111RRRRR 0000000001000000	GR[reg2]←SR[regID]	1	1	1					

(6/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
SUB	reg1,reg2	rrrrr001101RRRRR	GR[reg2]←GR[reg2]–GR[reg1]	1	1	1	×	×	×	×	
SUBR	reg1,reg2	rrrrr001100RRRRR	GR[reg2]←GR[reg1]–GR[reg2]	1	1	1	×	×	×	×	
SWITCH	reg1	00000000010RRRRR	adr←(PC+2) + (GR [reg1] logically shift left by 1) PC←(PC+2) + (sign-extend (Load-memory (adr,Halfword)) logically shift left by 1	5	5	5					
SXB	reg1	00000000101RRRRR	GR[reg1]←sign-extend (GR[reg1] (7 : 0))	1	1	1					
SXH	reg1	00000000111RRRRR	GR[reg1]←sign-extend (GR[reg1] (15 : 0))	1	1	1					
TRAP	vector	0 0 0 0 0 1 1 1 1 1 1 iiiii 0000000100000000	EIPC ←PC+4 (Restored PC) EIPSW ←PSW ECR.EICC ←Interrupt code PSW.EP ←1 PSW.ID ←1 PC ←00000040H (when vector is 00H to 0FH) 00000050H (when vector is 10H to 1FH)	3	3	3					
TST	reg1,reg2	rrrrr001011RRRRR	result←GR[reg2] AND GR[reg1]	1	1	1		0	×	×	
TST1	bit#3,disp16[reg1]	11bbb111110RRRRR ddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not (Load-memory-bit (adr,bit#3))	3 Note 3	3 Note 3	3 Note 3				×	
	reg2, [reg1]	rrrrr111111RRRRR 0000000011100110	adr←GR[reg1] Z flag←Not (Load-memory-bit (adr,reg2))	3 Note 3	3 Note 3	3 Note 3				×	
XOR	reg1,reg2	rrrrr001001RRRRR	GR[reg2]←GR[reg2] XOR GR[reg1]	1	1	1		0	×	×	
XORI	imm16,reg1,reg2	rrrrr110101RRRRR i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1] XOR zero-extend (imm16)	1	1	1		0	×	×	
ZXB	reg1	00000000100RRRRR	GR[reg1]←zero-extend (GR[reg1] (7 : 0))	1	1	1					
ZXH	reg1	00000000110RRRRR	GR[reg1]←zero-extend (GR[reg1] (15 : 0))	1	1	1					

Notes 1. ddddddddd: Higher 8 bits of disp9.

2. 3 if there is an instruction that rewrites the contents of the PSW immediately before.

3. If there is no wait state (3 + the number of read access wait states).

4. n is the total number of list12 load registers. (According to the number of wait states. Also, if there are no wait states, n is the total number of list12 registers. If n = 0, same operation as when n = 1)

5. RRRRR: other than 00000.

6. The lower halfword data only are valid.

7. ddddddddddddddddddddddd: The higher 21 bits of disp22.

8. ddddddddddddddd: The higher 15 bits of disp16.

9. According to the number of wait states (1 if there are no wait states).

10. b: bit 0 of disp16.

11. According to the number of wait states (2 if there are no wait states).

Notes 12. In this instruction, for convenience of mnemonic description, the source register is made reg2, but the reg1 field is used in the opcode. Therefore, the meaning of register specification in the mnemonic description and in the opcode differs from other instructions.

rrrrr = regID specification

RRRRR = reg2 specification

13. iiiii: Lower 5 bits of imm9.

IIII: Higher 4 bits of imm9.

14. Do not specify the same register for general-purpose registers reg1 and reg3.

15. sp/imm: specified by bits 19 and 20 of the sub-opcode.

16. ff = 00: Load sp in ep.

01: Load sign expanded 16-bit immediate data (bits 47 to 32) in ep.

10: Load 16-bit logically left shifted 16-bit immediate data (bits 47 to 32) in ep.

11: Load 32-bit immediate data (bits 63 to 32) in ep.

17. If imm = imm32, n + 3 clocks.

18. rrrrr: Other than 00000.

19. ddddddd: Higher 7 bits of disp8.

20. dddd: Higher 4 bits of disp5.

21. dddddd: Higher 6 bits of disp8.

<R>

APPENDIX E LIST OF CAUTIONS

This appendix lists cautions described in this document.
“Classification **(**hard/soft**)**” in table is as follows.

Hard: Cautions for microcontroller internal/external hardware
Soft: Cautions for software such as register settings or programs

(1/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 1	Hard	Introduction	ANI0 to ANI11	To use port 7 (P70/ANI0 to P711/ANI11) as A/D converter function pins and port I/O pins in mix, be sure to observe usage cautions (refer to 11.6 (4) Alternate I/O).	p.24 ☐
			IC	Directly connect this pin to V_{SS}.	p.24 ☐
			REGC	Connect the REGC pin to V_{SS} via a 4.7 μF capacitor.	p.24 ☐
			A0 to A15	Port 9 cannot be used as port pins or other alternate-function pins when the A0 to A15 pins are used in the separate bus mode.	p.24 ☐
Chapter 2	Hard	Pin function	Cautions on power application	When the power is turned on, the following pins may momentarily output an undefined level. • P10/ANO0[Note] pin • P11/ANO1[Note] pin • P53/KR3 pin	p.47 ☐
Chapter 3	Soft	CPU function	EIPC register EIPSW register FEPC register FEPSW register	Because only one set of these registers is available, the contents of these registers must be saved by program if multiple interrupts are enabled.	p.51 ☐
			EIPC, FEPC, CTPC registers	Even if EIPC or FEPC, or bit 0 of CTPC is set to 1 by the LDSR instruction, bit 0 is ignored when execution is returned to the main routine by the RETI instruction after interrupt servicing (this is because bit 0 of the PC is fixed to 0). Set an even value to EIPC, FEPC, and CTPC (bit 0 = 0).	p.51 ☐
			Program space	Because the 4 KB area of addresses 03FFF000H to 03FFFFFFH is an on-chip peripheral I/O area, instructions cannot be fetched from this area. Therefore, do not execute an operation in which the result of a branch address calculation affects this area.	p.59 ☐
			On-chip peripheral I/O area	If a register that can be accessed in byte units is accessed in halfword units, the higher 8 bits are undefined when the register is read, and data is written to the lower 8 bits.	p.64 ☐
				Addresses not defined as registers are reserved for future expansion. The operation is undefined and not guaranteed when these addresses are accessed.	p.64 ☐
				The internal ROM/RAM area and on-chip peripheral I/O area are assigned to successive addresses. When accessing the internal ROM/RAM area by incrementing or decrementing addresses using pointer operations and such, therefore, be careful not to access the onchip peripheral I/O area by mistakenly extending over the internal ROM/RAM area boundary.	p.64 ☐
			Programmable peripheral I/O area	The programmable peripheral I/O area exists only in the μPD703253Y. This area cannot be used in the μPD703249Y and 703252Y.	p.65 ☐
			A20, A21 pins	The V850ES/SG1 has 22 address pins (A0 to A21), so the external memory area appears as a repeated 4 MB image. When 22 address pins are used, it is necessary that $EV_{DD} = BV_{DD} = V_{DD}$.	p.65 ☐
			BPC register	When setting the PA15 bit to 1, be sure to set the BPC register to 8FFBH. When clearing the PA15 bit to 0, be sure to set the BPC register to 0000H.	p.79 ☐
			Setting data to special registers	Five NOP instructions or more must be inserted immediately after setting the IDLE1 mode, IDLE2 mode, or STOP mode (by setting the PSC.STP bit to 1).	p.81 ☐

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 3	Soft	CPU function	Setting data to special registers	When a store instruction is executed to store data in the command register, interrupts are not acknowledged. This is because it is assumed that steps <3> and <4> above are performed by successive store instructions. If another instruction is placed between <3> and <4>, and if an interrupt is acknowledged by that instruction, the above sequence may not be established, causing malfunction.	p.81 ☐
				Although dummy data is written to the PRCMD register, use the same general-purpose register used to set the special register (<4> in Example) to write data to the PRCMD register (<3> in Example). The same applies when a general-purpose register is used for addressing.	p.81 ☐
			Registers to be set first	Be sure to set the following registers first when using the V850ES/SG1. • System wait control register (VSWC) • On-chip debug mode register (OCDM) • Watchdog timer mode register 2 (WDTM2)	p.84 ☐
			VSWC register	Three clocks are required to access an on-chip peripheral I/O register (without a wait cycle). The V850ES/SG1 requires wait cycles according to the operating frequency. Set the following value to the VSWC register in accordance with the frequency used.	p.84 ☐
			WDTM2 register	Watchdog timer 2 automatically starts in the reset mode after reset is released. Write the WDTM2 register to activate this operation.	p.84 ☐
			Accessing specific on-chip peripheral I/O registers	When specific on-chip peripheral I/O registers are accessed, more wait states may be required in addition to the wait states set by the VSWC register.	p.85 ☐
				Accessing the above registers is prohibited in the following statuses. If a wait cycle is generated, it can only be cleared by a reset. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.86 ☐
			System reserved area	When the data in the flash memory has been deleted, all the bits are set to 1.	p.87 ☐
Chapter 4	Hard, soft	Port function	Port 0	This pin functions alternately as the on-chip debug function pin ($\overline{\text{DRST}}$) in the flash memory version (V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)). If the flash memory is used but on-chip debugging is not used, fix the P05/INTP2 pin to low level between when the reset signal of the $\overline{\text{RESET}}$ pin is released and when the OCDM.OCDM0 bit is cleared (0). For details, see 4.6.3 Cautions on on-chip debug pins.	p.95 ☐
	Soft		PF0 register	When an output pin is pulled up at EV_{DD} or higher, be sure to set the corresponding PF0n bit to 1.	p.97 ☐
			Port 1	The P10 and P11 pins may momentarily output an undefined level upon power application.	p.98 ☐
			P1 register	In the μPD703253Y, do not perform read from or write to the P1 register during D/A conversion (refer to 12.4.3 Cautions).	p.98 ☐
			PM1 register	When using P1n as the alternate function (ANOn pin output), set the PM1n bit to 1.	p.99 ☐
				In the μPD703253Y, when using one of the P10 and P11 pins as an I/O port and the other as a D/A output pin, do so in an application where the port I/O level does not change during D/A output.	p.99 ☐
			Port 3 alternate function specifications	The INTP7 pin and RXDA0 pin are alternate-function pins. When using the pin as the RXDA0 pin, disable edge detection for the INTP7 alternate-function pin. (Clear the INTF3.INTF31 bit and the INTR3.INTR31 bit to 0.) When using the pin as the INTP7 pin, stop UARTA0 reception. (Clear the UA0CTL0.UA0RXE bit to 0.)	p.104 ☐
			PF3 register	When an output pin is pulled up at EV_{DD} or higher, be sure to set the corresponding PF3n bit to 1.	p.105 ☐

(3/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 4	Soft	Port function	PF4 register	When an output pin is pulled up at EV_{DD} or higher, be sure to set the corresponding PF4n bit to 1.	p.108 ☐
	Hard, soft		Port 5	This pin functions alternately as the on-chip debug function pin (DDI, DDO, DCK, or DMS) in the flash memory version (V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)). If the flash memory is used but on-chip debugging is not used, fix the P05/INTP2 pin to low level between when the reset signal of the $\overline{\text{RESET}}$ pin is released and when the OCDM.OCDM0 bit is cleared (0). Although the mask ROM versions do not support the on-chip debug mode, handle the P05/INTP2 pin the same as the flash memory versions. For details, see 4.6.3 Cautions on on-chip debug pins.	p.109 ☐
	Hard			The P53 pin may momentarily output an undefined level upon power application.	p.109 ☐
	Soft		PF5 register	When an output pin is pulled up at EV_{DD} or higher, be sure to set the corresponding PF5n bit to 1.	p.112 ☐
			P7H register, P7L register	Do not perform read or write to the P7H and P7L registers during A/D conversion (refer to 11.6 (4) Alternate I/O).	p.114 ☐
			PM7H register, PM7L register	When using the P7n pin as its alternate function (ANIn pin), set the PM7n bit to 1.	p.114 ☐
			PMC9 register	Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once. If even one of the A0 to A15 pins is not used in the separate bus mode, port 9 pins can be used as port pins or other alternate-function pins.	p.118 ☐
			PFC9 register	Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once. If even one of the A0 to A15 pins is not used in the separate bus mode, port 9 pins can be used as port pins or other alternate-function pins.	p.119 ☐
			Port 9 alternate function specifications	The RXDA1 and KR7 pins must not be used at the same time. When using the RXDA1 pin, do not use the KR7 pin. When using the KR7 pin, do not use the RXDA1 pin (it is recommended to set the PFC91 bit to 1 and clear the PFCE91 bit to 0).	p.121 ☐
			PF9 register	When an output pin is pulled up at EV_{DD} or higher, be sure to set the corresponding PF9n bit to 1.	p.122 ☐
			PMCDL register	When the SMSEL bit of the EXIMC register = 1 (separate mode) and the BS30 to BS00 bits of the BSC register = 0 (8-bit bus width), do not specify the AD8 to AD15 pins.	p.131 ☐
	Hard		Using port pin as alternate-function pin	In the μPD703253Y, when using one of the P10 and P11 pins as an I/O port and the other as a D/A output pin **(ANO0, ANO1)**, do so in an application where the port I/O level does not change during D/A output.	p.169 ☐
	Soft		Cautions on switching from port mode to alternate-function mode	To switch from the port mode to alternate-function mode in the following order. <1> Set the PFn register[Note]: N-ch open-drain setting <2> Set the PFCn and PFCEn registers: Alternate-function selection <3> Set the corresponding bit of the PMCn register to 1: Switch to alternate-function mode If the PMCn register is set first, note with caution that, at that moment or depending on the change of the pin states in accordance with the setting of the PFn, PFCn, and PFCEn registers, unexpected operations may occur.	p.175 ☐

(4/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 4	Soft	Port function	Cautions on switching from port mode to alternate-function mode	Regardless of the port mode/alternate-function mode, the Pn register is read and written as follows. • Pn register read: Read the port output latch value (when PMn.PMnm bit = 0), or read the pin states (PMn.PMnm bit = 1). • Pn register write: Write to the port output latch	p.175 ☐
			Cautions on alternate-function mode (input)	Therefore, switch between the port mode and alternate-function mode in the following sequence. • To switch from port mode to alternate-function mode (input) Set the pins to the alternate-function mode using the PMCn register and then enable the alternate-function operation. • To switch from alternate-function mode (input) to port mode Stop the alternate-function operation and then switch the pins to the port mode.	p.176 ☐
			PFn.PFnm bit in port mode	In port mode, the PFn.PFnm bit is valid only in the output mode (PMn.PMnm bit = 0). In the input mode (PMnm bit = 1), the value of the PFnm bit is not reflected in the buffer.	p.177 ☐
			Cautions on bit manipulation instruction for port n register (Pn)	When a 1-bit manipulation instruction is executed on a port that provides both input and output functions, the value of the output latch of an input port that is not subject to manipulation may be written in addition to the targeted bit. Therefore, it is recommended to rewrite the output latch when switching a port from input mode to output mode.	p.178 ☐
	Hard, soft		Cautions on on-chip debug pins	The V850ES/SG1 does not have an on-chip debug function. However, when using V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y) as the flash memory version for developing software, the following actions are required. • Clear the OCDM0 bit of the OCDM register (special register) (0) At this time, fix the P05/INTP2 pin to low level from when reset by the $\overline{\text{RESET}}$ pin is released until the above action is taken. If a high level is input to the P05/INTP2 pin before the above action is taken, it may cause a malfunction (CPU deadlock). Handle the P05/INTP2 pin with the utmost care.	p.179 ☐
	Soft			The V850ES/SG1 does not have an on-chip debug function but the P05/INTP2 pin has an onchip pull-down resistor. With the V850ES/SG1, therefore, the on-chip pull-down resistor must be disconnected by clearing the OCDM0 bit to 0.	p.179 ☐
	Hard		Cautions on P10, P11, P53 pin when power is turned on	When the power is turned on, the following pins may output an undefined level temporarily even during reset. • P10/AND0 pin • P11/ANO1 pin • P53/KR3 pin	p.180 ☐
			Hysteresis characteristic	In port mode, the following port pins do not have hysteresis characteristics. P02 to P06 P31 to P35, P37 to P39 P40 to P42 P50 to P55 P90 to P97, P99, P910, and P912 to P915	p.180 ☐
	Soft		Cautions on separate bus mode	Port 9 pins cannot be used as port pins or other alternate-function pins if even one of the A0 to A15 pins is used in the separate bus mode. After setting the PFC9 and PFCE9 registers to 0000H, therefore, set all 16 bits of the PMC9 register to FFFFH at once. If even one of the A0 to A15 pins is not used in the separate bus mode, port 9 pins can be used as port pins or other alternate-function pins.	p.180 ☐

(5/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 5	Soft	Bus control function	Pin status when internal ROM is accessed	When a write access is performed to the internal ROM area, address, data, and control signals are activated in the same way as access to the external memory area.	p.182 ☐
			EXIMC register	Set the EXIMC register from the internal ROM or internal RAM area before making an external access. After setting the EXIMC register, be sure to insert a NOP instruction.	p.184 ☐
			BSC register	Write to the BSC register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the BSC register are complete.	p.185 ☐
				Be sure to set bits 14, 12, 10, and 8 to “1”, and clear bits 15, 13, 11, 9, 7, 5, 3, and 1 to “0”.	p.185 ☐
			DWC0 register	Write to the DWC0 register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the DWC0 register are complete.	p.193 ☐
				Be sure to clear bits 15, 11, 7, and 3 to “0”.	p.193 ☐
			AWC register	Write to the AWC register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the AWC register are complete.	p.196 ☐
				Be sure to set bits 15 to 8 to “1”.	p.196 ☐
			BCC register	Write to the BCC register after reset, and then do not change the set values. Also, do not access an external memory area until the initial settings of the BCC register are complete.	p.197 ☐
				Be sure to set bits 15, 13, 11, and 9 to “1”, and clear bits 14, 12, 10, 8, 6, 4, 2, and 0 to “0”.	p.197 ☐
Chapter 6	Soft	Clock generation function	PCC register	Do not change the CPU clock (by using the CK3 to CK0 bits) while CLKOUT is being output.	p.211 ☐
				Use a bit manipulation instruction to manipulate the CK3 bit. When using an 8-bit manipulation instruction, do not change the set values of the CK2 to CK0 bits.	p.211 ☐
				When stopping the main clock, stop the PLL. Also stop the operations of the on-chip peripheral functions operating with the main clock.	p.212 ☐
				If the following conditions are not satisfied, change the CK2 to CK0 bits so that the conditions are satisfied, then change to the subclock operation mode. Internal system clock (f_{CLK}) > Subclock (f_{XT}: 32.768 kHz) × 4	p.212 ☐
				Enable operation of the on-chip peripheral functions operating with the main clock only after the oscillation of the main clock stabilizes. If their operations are enabled before the lapse of the oscillation stabilization time, a malfunction may occur.	p.213 ☐
			RCM register	The internal oscillator cannot be stopped while the CPU is operating on the internal oscillation clock (CCLS.CCLSF bit = 1). Do not set the RSTOP bit to 1.	p.214 ☐
				The internal oscillator oscillates if the CCLS.CCLSF bit is set to 1 (when WDT overflow occurs during oscillation stabilization) even when the RSTOP bit is set to 1. At this time, the RSTOP bit remains being set to 1.	p.214 ☐
			PLLCTL register	To stop the PLL operation, first set the clock through mode (SELPLL bit = 0), wait for at least 8 clocks, and then stop the PLL (PLLON bit = 0). When the PLLON bit is cleared to 0, the SELPLL bit is automatically cleared to 0 (clock-through mode), but be sure to stop the PLL in the above procedure.	p.216 ☐
				The SELPLL bit can be set to 1 only when the PLL clock frequency is stabilized. If not (unlocked), “0” is written to the SELPLL bit if data is written to it.	p.216 ☐

(6/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 6	Soft	Clock generation function	CKC register	The PLL mode cannot be used at 5.0 MHz < f_X ≤ 10.0 MHz.	p.217 ☐
				Before changing the multiplication factor between 4 and 8 by using the CKC register, set the clock-through mode and stop the PLL.	p.217 ☐
				Be sure to set bits 3 and 1 to “1” and clear bits 7 to 4 and 2 to “0”.	p.217 ☐
			LOCKR register	The LOCK register does not reflect the lock status of the PLL in real time.	p.218 ☐
			PLLS register	Set so that the lockup time is 800 μs or longer.	p.219 ☐
				Do not change the PLLS register setting during the lockup period.	p.219 ☐
Chapter 7	Soft	16-bit timer/event counter P (TMP)	TPnCTL0 register	Set the TPnCKS2 to TPnCKS0 bits when the TPnCE bit = 0. When the value of the TPnCE bit is changed from 0 to 1, the TPnCKS2 to TPnCKS0 bits can be set simultaneously.	p.225 ☐
				Be sure to clear bits 3 to 6 to “0”.	p.225 ☐
			TPnCTL1 register	Set the TPnEEE and TPnMD2 to TPnMD0 bits when the TPnCTL0.TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) The operation is not guaranteed when rewriting is performed with the TPnCE bit = 1. If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.	p.226 ☐
				Be sure to clear bits 3, 4, and 7 to “0”.	p.226 ☐
			TPnIOC0 registers	The pin output changes if the setting of the TPnIOC0 register is rewritten when the port is set to output TOPn0 and TOPn1. Therefore, note changes in the pin status by setting the port to the input mode and making the output status of the pins a high-impedance state.	p.228 ☐
				Rewrite the TPnOL1, TPnOE1, TPnOL0, and TPnOE0 bits when the TPnCTL0.TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.	p.228 ☐
				Even if the TPnOLm bit is manipulated when the TPnCE and TPnOEm bits are 0, the TOPnm pin output level varies (m = 0, 1).	p.228 ☐
			TPnIOC1 register	Rewrite the TPnIS3 to TPnIS0 bits when the TPnCTL0.TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.	p.229 ☐
			TPnIOC2 register	Rewrite the TPnEES1, TPnEES0, TPnETS1, and TPnETS0 bits when the TPnCTL0.TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.	p.230 ☐
			TPnOPT0 register	Rewrite the TPnCCS1 and TPnCCS0 bits when the TPnCE bit = 0. (The same value can be written when the TPnCE bit = 1.) If rewriting was mistakenly performed, clear the TPnCE bit to 0 and then set the bits again.	p.231 ☐
				Be sure to clear bits 1 to 3, 6, and 7 to “0”.	p.231 ☐
			TPnCCR0 register	Accessing the TPnCCR0 register is prohibited in the following statuses. For details, see 3.4.9 (2) Accessing specific on-chip peripheral I/O registers. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.232 ☐
			TPnCCR1 register	Accessing the TPnCCR1 register is prohibited in the following statuses. For details, see 3.4.9 (2) Accessing specific on-chip peripheral I/O registers. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.234 ☐

(7/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 7	Soft	16-bit timer/event counter P (TMP)	TPnCNT register	Accessing the TPnCNT register is prohibited in the following statuses. For details, see 3.4.9 (2) Accessing specific on-chip peripheral I/O registers. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.236 ☐
			Operation	To use the external event count mode, specify that the valid edge of the TIPn0 pin capture trigger input is not detected (by clearing the TPnIOC1.TPnIS1 and TPnIOC1.TPnIS0 bits to “00”).	p.238 ☐
				To use the external trigger pulse output mode, one-shot pulse output mode, or pulse width measurement mode, select the internal clock (by setting the TPnCTL1.TPnEEE bit to 0) as the count clock.	p.238 ☐
			Overflow operation	After the overflow interrupt request signal (INTTPnOV) has been generated, be sure to check that the overflow flag (TPnOVF bit) is set to 1.	p.239 ☐
			Batch write	Writing to the TPnCCR1 register includes enabling of batch write. Thus, rewrite the TPnCCR1 register after rewriting the TPnCCR0 register.	p.243 ☐
			Notes on rewriting TPnCCR0 register	When the value of the TPnCCR0 register is rewritten to a smaller value during counting, the 16-bit counter may overflow. If there is a possibility of overflow, stop counting and then change the set value.	pp.252, 263 ☐
			External event count mode	In the external event count mode, the TPnCCR0 and TPnCCR1 registers must not be cleared to 0000H.	p.257 ☐
			Register setting for operation in external event count mode	Set 00H to the TPnIOC0 register.	p.260 ☐
				When the external clock is used as the count clock, the external clock can be input only from the TIPn0 pin. At this time, clear the TPnIOC1.TPnIS1 and TPnIS0 bits to 00 (capture trigger input (TIPn0 pin): No edge detected).	p.260 ☐
			Operation timing in external event count mode	In the external event count mode, do not set the TPnCCR0 and TPnCCR1 registers to 0000H.	p.262 ☐
				In the external event count mode, use of the timer output (TOPn0, TOPn1) is disabled.	p.262 ☐
			External trigger pulse output mode	In external trigger pulse output mode, select the internal clock (set TPnCTL1.TPnEEE bit = 0) as the count clock.	p.266 ☐
			Note on changing pulse width during operation	To change the PWM waveform while the counter is operating, write the TPnCCR1 register last. Rewrite the TPnCCRm register after writing the TPnCCR1 register after the INTTPnCC0 signal is detected.	p.272 ☐
			One-shot pulse output mode	In one-shot pulse output mode, select the internal clock (set TPnCTL1.TPnEEE bit = 0) as the count clock.	p.278 ☐
			Setting of registers in one-shot pulse output mode	The one-shot pulse is not output if the set value of the TPnCCR1 register is greater than the set value of the TPnCCR0 register in the one-shot pulse output mode.	p.281 ☐
			Notes on rewriting TPnCCRm register	When the value of the TPnCCRm register is rewritten to a smaller value during counting, the 16-bit counter may overflow. If there is a possibility of overflow, stop counting and then change the set value.	p.283 ☐

(8/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 7	Hard	16-bit timer/event counter P (TMP)	Processing of overflow if capture trigger interval is long	If the pulse width is greater than one cycle of the 16-bit counter, care must be exercised because an overflow may occur more than once from the first capture trigger to the next.	p.309 ☐
	Soft		Note on capture operation	If the capture operation is used and if a slow clock is selected as the count clock, FFFFH, not 0000H, may be captured to the TPnCCRm register if the capture trigger is input immediately after the TPnCTL0.TPnCE bit is set to 1.	p.311 ☐
			Pulse width measurement mode (TPnMD2 to TPnMD0 bits = 110)	When in pulse width measurement mode, select the internal clock (set TPnCTL1.TPnEEE bit = 0) as the count clock.	p.312 ☐
				If a slow clock is selected as the count clock, FFFFH, not 0000H, may be captured to the TPnCCRm register if the capture trigger is input immediately after the TPnCTL0.TPnCE bit has been set to 1.	p.317 ☐
Chapter 8	Soft	16-bit interval timer M (TMM)	TM0CMP0 register	Do not set the TM0CMP0 register to FFFFH.	p.320 ☐
			TM0CTL0 register	Set the TM0CKS2 to TM0CKS0 bits when TM0CE bit = 0. When changing the value of TM0CE from 0 to 1, it is not possible to set the value of the TM0CKS2 to TM0CKS0 bits simultaneously.	p.321
				Be sure to clear bits 3 to 6 to “0”.	p.321 ☐
			Interval timer mode operation timing	Do not set the TM0CMP0 register to FFFFH.	pp.322, 325, 326 ☐
			Count operation	It takes the 16-bit counter up to the following time to start counting after the TM0CTL0.TM0CE bit is set to 1, depending on the count clock selected.	p.326 ☐
			TM0CMP0, TM0CTL0 registers	Rewriting the TM0CMP0 and TM0CTL0 registers is prohibited while TMM0 is operating. If these registers are rewritten while the TM0CE bit is 1, the operation cannot be guaranteed. If they are rewritten by mistake, clear the TM0CTL0.TM0CE bit to 0, and re-set the registers.	p.326 ☐
Chapter 9	Soft	Watch timer function	PRSM0 register	Do not change the values of the BGCS00 and BGCS01 bits during watch timer operation.	p.330 ☐
				Set the PRSM0 register before setting the BGCE0 bit to 1.	p.330 ☐
				Set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain an f_{BRG} frequency of 32.768 kHz.	p.330 ☐
			PRSCM0 register	Do not rewrite the PRSCM0 register during watch timer operation.	p.331 ☐
				Set the PRSCM0 register before setting the PRSM0.BGCE0 bit to 1.	p.331 ☐
				Set the PRSM0 and PRSCM0 registers according to the main clock frequency that is used so as to obtain an f_{BRG} frequency of 32.768 kHz.	p.331 ☐
			WTM register	Rewrite the WTM2 to WTM7 bits while both the WTM0 and WTM1 bits are 0.	p.333 ☐
	Hard		INTWT signal	Some time is required before the first watch timer interrupt request signal (INTWT) is generated after operation is enabled (WTM.WTM1 and WTM.WTM0 bits = 1).	p.336 ☐
Chapter 10	Soft	Functions of watchdog timer 2	Default start watchdog timer	Watchdog timer 2 automatically starts in the reset mode following reset release. When watchdog timer 2 is not used, either stop its operation before reset is executed via this function, or clear watchdog timer 2 once and stop it within the next interval time. Also, write to the WDTM2 register for verification purposes only once, even if the default settings (reset mode, interval time: $f_R/2^{19}$) do not need to be changed.	p.337 ☐
				For the non-maskable interrupt servicing due to a non-maskable interrupt request signal (INTWDT2), see 19.2.2 (2) INTWDT2 signal.	p.337 ☐

(9/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 10	Soft	Functions of watchdog timer 2	WDTM2 register	Accessing the WDTM2 register is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.339 ☐
				Although watchdog timer 2 can be stopped just by stopping operation of the internal oscillator, clear the WDTM2 register to 00H to securely stop the timer (to avoid selection of the main clock or subclock due to an erroneous write operation).	p.339 ☐
				If the WDTM2 register is rewritten twice after reset, an overflow signal is forcibly generated and the counter is reset.	p.339 ☐
				To intentionally generate an overflow signal, write to the WDTM2 register only twice or write a value other than ACH to the WDTE register once. However, when watchdog timer 2 is set to stop operation, an overflow signal is not generated even if data is written to the WDTM2 register only twice, or a value other than “ACH” is written to the WDTE register only once.	p.339 ☐
				To stop the operation of watchdog timer 2, set the RCM.RSTOP bit to 1 (to stop the internal oscillator) and write 00H in the WDTM2 register. If the RCM.RSTOP bit cannot be set to 1, set the WDCS23 bit to 1 ($2^n/f_{XX}$ is selected and the clock can be stopped in the IDLE1, IDLW2, sub-IDLE, and subclock operation modes).	p.339 ☐
			WDTE register	When a value other than “ACH” is written to the WDTE register, an overflow signal is forcibly output.	p.341 ☐
				When a 1-bit memory manipulation instruction is executed for the WDTE register, an overflow signal is forcibly output.	p.341 ☐
				To intentionally generate an overflow signal, write to the WDTM2 register only twice or write a value other than ACH to the WDTE register once. However, when the watchdog timer 2 is set to stop operation, an overflow signal is not generated even if data is written to the WDTM2 register only twice, or a value other than “ACH” is written to the WDTE register only once.	p.341 ☐
				The read value of the WDTE register is “9AH” (which differs from written value “ACH”).	p.341 ☐
Chapter 11	Hard	A/D converter	ANI0 to ANI11 pins	Make sure that the voltages input to the ANI0 to ANI11 pins do not exceed the rated values. In particular if a voltage of AV_{REF0} or higher is input to a channel, the conversion value of that channel becomes undefined, and the conversion values of the other channels may also be affected.	p.346 ☐
	Soft		ADA0M0 register	Accessing the ADA0M0 register is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.347 ☐
				Changing the ADA0M1 register bits is prohibited while A/D conversion is enabled (ADA0CE bit = 1).	p.348 ☐
				Stop the A/D conversion operation (ADA0CE = 0) before writing to the ADA0M0, ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1). • Normal conversion mode • One-shot select mode/one-shot scan mode in high-speed conversion mode	p.348 ☐
				When the external trigger or timer trigger mode is selected (ADA0TMD bit = 1), set the high-speed conversion mode (ADA0M1.ADA0HS1 bit = 1). Do not input a trigger during the stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0CE bit = 1).	p.348 ☐

(10/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 11	Soft	A/D converter	ADA0M1 register	Changing the ADA0M1 register is prohibited while A/D conversion is enabled (ADA0M0.ADA0CE bit = 1).	p.349 ☐
				When the external trigger or timer trigger mode is selected (ADA0M0.ADA0TMD bit = 1), set high-speed conversion mode (ADA0HS1 bit = 1), and do not input a trigger during the stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0CE = 1).	p.349 ☐
				Be sure to clear bits 6 to 3 to “0”.	p.349 ☐
				Set as 2.6 μs ≤ conversion time ≤ 10.4 μs.	pp.350, 351 ☐
			ADA0M2 register	Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0M2 register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1). • Normal conversion mode • One-shot select mode/one-shot scan mode in high-speed conversion mode	p.352 ☐
				Be sure to clear bits 7 to 2 to “0”.	p.352 ☐
			ADA0S register	Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0S register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1). • Normal conversion mode • One-shot select mode/one-shot scan mode in high-speed conversion mode	p.353 ☐
				Be sure to clear bits 7 to 4 to “0”.	p.353 ☐
			ADA0CRn, ADA0CRnH registers	Accessing the ADA0CRn and ADA0CRnH registers is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.354 ☐
				A write operation to the ADA0M0 and ADA0S registers may cause the contents of the ADA0CRn register to become undefined. After the conversion, read the conversion result before writing to the ADA0M0 and ADA0S registers. Correct conversion results may not be read if a sequence other than the above is used.	p.354 ☐
			ADA0PFM register	Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0PFM register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1). • Normal conversion mode • One-shot select mode/one-shot scan mode in high-speed conversion mode	p.356 ☐
			ADA0PFT register	Stop the A/D conversion operation (ADA0M0.ADA0CE = 0) before writing to the ADA0PFT register in either of the following modes, then enable the A/D conversion operation (ADA0CE = 1). • Normal conversion mode • One-shot select mode/one-shot scan mode in high-speed conversion mode	p.357 ☐
			External trigger mode	When the external trigger mode is selected, set the high-speed conversion mode. Do not input a trigger during stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0M0.ADA0CE bit = 1).	p.360 ☐
			Timer trigger mode	When the external trigger mode is selected, set the high-speed conversion mode. Do not input a trigger during stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0M0.ADA0CE bit = 1).	p.361 ☐

(11/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 11	Soft	A/D converter	Input range of ANI0 to ANI11 pins	Input the voltage within the specified range to the ANI0 to ANI11 pins. If a voltage equal to or higher than AV_{REF0} or equal to or lower than AVSS (even within the range of the absolute maximum ratings) is input to any of these pins, the conversion value of that channel is undefined, and the conversion value of the other channels may also be affected.	p.371 ☐
			Counter-measures against noise	To maintain the 10-bit resolution, the ANI0 to ANI11 pins must be effectively protected from noise. The influence of noise increases as the output impedance of the analog input source becomes higher. To lower the noise, connecting an external capacitor as shown in Figure 11-12 is recommended.	p.371 ☐
			Alternate I/O	The analog input (ANI0 to ANI11) pins are multiplexed with port pins. The AVREF0 power pin is multiplexed with the reference power supply to the A/D converter and the I/O buffer power supply of port 7. If any of the following processings is performed during A/D conversion, therefore, the expected A/D conversion value may not be obtained.	p.372 ☐
			Interrupt request flag (ADIF)	The interrupt request flag (ADIF) is not cleared even if the contents of the ADA0S register are changed. If the analog input pin is changed during A/D conversion, therefore, the result of converting the previously selected analog input signal may be stored and the conversion end interrupt request flag may be set immediately before the ADA0S register is rewritten. If the ADIF flag is read immediately after the ADA0S register is rewritten, the ADIF flag may be set even though the A/D conversion of the newly selected analog input pin has not been completed. When A/D conversion is stopped, clear the ADIF flag before resuming conversion.	p.373 ☐
	Hard		AV_{REF0} pin	The AV_{REF0} pin is used as the power supply pin of the A/D converter and also supplies power to the alternate-function ports. In an application where a backup power supply is used, be sure to supply the same voltage as V_{DD} to the AV_{REF0} pin as shown in Figure 11-15.	p.374 ☐
				The AV_{REF0} pin is also used as the reference voltage pin of the A/D converter. If the source supplying power to the AV_{REF0} pin has a high impedance or if the power supply has a low current supply capability, the reference voltage may fluctuate due to the current that flows during conversion (especially, immediately after the conversion operation enable bit ADA0CE has been set to 1). As a result, the conversion accuracy may drop. To avoid this, it is recommended to connect a capacitor across the AV_{REF0} and AV_{SS} pins to suppress the reference voltage fluctuation as shown in Figure 11-15.	p.374 ☐
				If the source supplying power to the AV_{REF0} pin has a high DC resistance (for example, because of insertion of a diode), the voltage when conversion is enabled may be lower than the voltage when conversion is stopped, because of a voltage drop caused by the A/D conversion current.	p.374 ☐
	Soft		Reading ADA0CRn register	When the ADA0M0 to ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register is written, the contents of the ADA0CRn register may be undefined. Read the conversion result after completion of conversion and before writing to the ADA0M0 to ADA0M2, ADA0S, ADA0PFM, or ADA0PFT register. Also, when an external/timer trigger is acknowledged, the contents of the ADA0CRn register may be undefined. Read the conversion result after completion of conversion and before the next external/timer trigger is acknowledged. The correct conversion result may not be read at a timing different from the above.	p.374 ☐

(12/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 11	Soft	A/D converter	Standby mode	Because the A/D converter stops operating in the STOP mode, conversion results are invalid, so power consumption can be reduced. Operations are resumed after the STOP mode is released, but the A/D conversion results after the STOP mode is released are invalid. When using the A/D converter after the STOP mode is released, before setting the STOP mode or releasing the STOP mode, clear the ADA0M0.ADA0CE bit to 0 then set the ADA0CE bit to 1 after releasing the STOP mode. In the IDLE1, IDLE2, or subclock operation mode, operation continues. To lower the power consumption, therefore, clear the ADA0M0.ADA0CE bit to 0. In the IDLE1 and IDLE2 modes, since the analog input voltage value cannot be retained, the A/D conversion results after the IDLE1 and IDLE2 modes are released are invalid. The results of conversions before the IDLE1 and IDLE2 modes were set are valid.	p.374 ☐
			Restriction for each mode	When the external trigger mode or timer trigger mode is selected, set the high-speed conversion mode. Do not input a trigger during stabilization time that is inserted only once after the A/D conversion operation is enabled (ADA0M0.ADA0CE bit = 1).	p.375 ☐
			Variation of A/D conversion results	The results of the A/D conversion may vary depending on the fluctuation of the supply voltage, or may be affected by noise. To reduce the variation, take counteractive measures with the program such as averaging the A/D conversion results.	p.375 ☐
	Hard		A/D conversion result hysteresis characteristics	The successive comparison type A/D converter holds the analog input voltage in the internal sample & hold capacitor and then performs A/D conversion. After the A/D conversion has finished, the analog input voltage remains in the internal sample & hold capacitor. As a result, the following phenomena may occur. • When the same channel is used for A/D conversions, if the voltage is higher or lower than the previous A/D conversion, then hysteresis characteristics may appear where the conversion result is affected by the previous value. Thus, even if the conversion is performed at the same potential, the result may vary. • When switching the analog input channel, hysteresis characteristics may appear where the conversion result is affected by the previous channel value. This is because one A/D converter is used for the A/D conversions. Thus, even if the conversion is performed at the same potential, the result may vary.	p.375 ☐
Chapter 12	Soft	D/A converter	DA0CS0, DA0CS1 registers	In the real-time output mode (DA0M.DA0MDn bit = 1), set the DA0CSn register before the INTTP2CC0/INTTP3CC0 signals are generated. D/A conversion starts when the INTTP2CC0/INTTP3CC0 signals are generated.	p.382 ☐
			Cautions	Do not change the set value of the DA0CSn register while the trigger signal is being issued in the real-time output mode.	p.384 ☐
				Before changing the operation mode, be sure to clear the DA0M.DA0CEn bit to 0.	p.384 ☐
				When using one of the P10/AN00 and P11/AN01 pins as an I/O port and the other as a D/A output pin, do so in an application where the port I/O level does not change during D/A output.	p.384 ☐
	Hard			Make sure that $AV_{REF0} = V_{DD} = AV_{REF1}$ = 3.0 to 3.6 V. If this range is exceeded, the operation is not guaranteed.	p.384 ☐
				Apply power to AV_{REF1} at the same timing as AV_{REF0}.	p.384 ☐
				No current can be output from the ANOn pin (n = 0, 1) because the output impedance of the D/A converter is high. When connecting a resistor of 2 MΩ or less, insert a JFET input operational amplifier between the resistor and the ANOn pin.	p.384 ☐
	Soft			Because the D/A converter stops operation in the STOP mode, the ANO0 and ANO1 pins go into a highimpedance state, and the power consumption can be reduced. In the IDLE1, IDLE2, or subclock operation mode, however, the operation continues. To lower the power consumption, therefore, clear the DA0M.DA0CEn bit to 0.	p.384 ☐

(13/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 13	Soft	Asynchronous serial interface A (UARTA)	UAnSTR register	Be sure to read the error flags of the UAnPE, UAnFE, and UAnOVE bits to check the flag status, and then clear the flags by writing "0" to them.	p.391 ☐
			Continuous transmission procedure	When initializing transmissions during the execution of continuous transmissions, make sure that the UAnSTR.UAnTSF bit is 0, then perform the initialization. Transmit data that is initialized when the UAnTSF bit is 1 cannot be guaranteed.	p.398 ☐
			UART reception	Be sure to read the UAnRX register even when a reception error occurs. If the UAnRX register is not read, an overrun error occurs during reception of the next data, and reception errors continue occurring indefinitely.	p.400 ☐
				When reception is completed, read the UAnRX register after the reception complete interrupt request signal (INTUAnR) has been generated, and clear the UAnPWR or UAnRXE bit to 0. If the UAnPWR or UAnRXE bit is cleared to 0 before the INTUAnR signal is generated, the read value of the UAnRX register cannot be guaranteed.	p.400 ☐
				If receive completion processing (INTUAnR signal generation) of UARTAn and the UAnPWR bit = 0 or UAnRXE bit = 0 conflict, the INTUAnR signal may be generated in spite of these being no data stored in the UAnRX register. To complete reception without waiting INTUAnR signal generation, be sure to clear (0) the interrupt request flag (UAnRIF) of the UAnRIC register, after setting (1) the interrupt mask flag (UAnRMK) of the interrupt control register (UAnRIC) and then set (1) the UAnPWR bit = 0 or UAnRXE bit = 0.	p.400 ☐
			Reception errors	When an INTUAnR signal is generated, the UAnSTR register must be read to check for errors.	p.401 ☐
				If a receive error interrupt occurs during continuous reception, read the contents of the UAnSTR register must be read before the next reception is completed, then perform error processing.	p.402 ☐
			Baud rate generator configuration	UARTAn cannot be used if the CPU clock (f_{CPU}) is slower than f_{UCLK}.	p.405 ☐
			UAnCTL1 register	Clear the UAnCTL0.UAnPWR bit to 0 before rewriting the UAnCTL1 register.	p.406 ☐
			UAnCTL2 register	Clear the UAnCTL0.UAnPWR bit to 0 or clear the UAnTXE and UAnRXE bits to 00 before rewriting the UAnCTL2 register.	p.407 ☐
			Baud rate error	The baud rate error during transmission must be within the error tolerance on the receiving side.	p.408 ☐
				The baud rate error during reception must satisfy the range indicated in (5) Allowable baud rate range during reception.	p.408 ☐
			Allowable baud rate range during reception	The baud rate error during reception must be set within the allowable error range using the following equation.	p.410 ☐
			When the clock supply to UARTAn is stopped	When the clock supply to UARTAn is stopped (for example, in IDLE1, IDLE2, or STOP mode), the operation stops with each register retaining the value it had immediately before the clock supply was stopped. The TXDAn pin output also holds and outputs the value it had immediately before the clock supply was stopped. However, the operation is not guaranteed after the clock supply is resumed. Therefore, after the clock supply is resumed, the circuits should be initialized by setting the UAnCTL0.UAnPWR, UAnCTL0.UAnRXEn, and UAnCTL0.UAnTXEn bits to 000.	p.413 ☐
			RXDA1 pin KR7 pin	The RXDA1 and KR7 pins must not be used at the same time. To use the RXDA1 pin, do not use the KR7 pin. To use the KR7 pin, do not use the RXDA1 pin (it is recommended to set the PFC91 bit to 1 and clear PFCE91 bit to 0).	p.413 ☐

(14/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 13	Soft	Asynchronous serial interface A (UARTA)	Transfer of transmit data and receive data using DMA transfer	In UARTAn, the interrupt caused by a communication error does not occur. When performing the transfer of transmit data and receive data using DMA transfer, error processing cannot be performed even if errors (parity, overrun, framing) occur during transfer. Either read the UAnSTR register after DMA transfer has been completed to make sure that there are no errors, or read the UAnSTR register during communication to check for errors.	p.413 ☐
			Start up the UARTAn	Start up the UARTAn in the following sequence. <1> Set the UAnCTL0.UAnPWR bit to 1. <2> Set the ports. <3> Set the UAnCTL0.UAnTXE bit to 1, UAnCTL0.UAnRXE bit to 1.	p.413 ☐
			Stop UARTAn	Stop the UARTAn in the following sequence. <1> Set the UAnCTL0.UAnTXE bit to 0, UAnCTL0.UAnRXE bit to 0. <2> Set the ports and set the UAnCTL0.UAnPWR bit to 0 (it is not a problem if port setting is not changed).	p.413 ☐
			Transmit mode	In transmit mode (UAnCTL0.UAnPWR bit = 1 and UAnCTL0.UAnTXE bit = 1), do not overwrite the same value to the UAnTX register by software because transmission starts by writing to this register. To transmit the same value continuously, overwrite the same value.	p.413 ☐
			Continuous transmission	In continuous transmission, the communication rate from the stop bit to the next start bit is extended 2 base clocks more than usual. However, the reception side initializes the timing by detecting the start bit, so the reception result is not affected.	p.413 ☐
Chapter 14	Soft	3-wire variable-length serial I/O (CSIB)	CSIB0 and I^2C01 mode switching	The transmit/receive operation of CSIB0 and I^2C01 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.	p.414 ☐
			CBnCTL0 register	To forcibly suspend transmission/reception, clear the CBnPWR bit instead of the CBnTXE and CBnRxE bit to 0.	p.418 ☐
				Be sure to set bits 3 and 2 to "0".	p.418 ☐
			CBnCTL1 register	The CBnCTL1 register can be rewritten only when the CBnCTL0.CBnPWR bit = 0.	p.422 ☐
			CBnCTL2 register	The CBnCTL2 register can be rewritten only when the CBnCTL0.CBnPWR bit = 0 or when both the CBnTXE and CBnRXE bits = 0.	p.423 ☐
			CBnSTR register	In single transfer mode, writing to the CBnTX register with the CBnTSF bit set to 1 is ignored. This has no influence on the operation during transfer.	p.425 ☐
			Continuous transfer mode (master mode, transmission mode)	In continuous transmission mode, the reception completion interrupt request signal (INTCBnR) is not generated.	p.440 ☐
			Continuous transfer mode (slave mode, transmission mode)	In continuous transmission mode, the reception completion interrupt request signal (INTCBnR) is not generated.	p.449 ☐
			Clock timing	In single transfer mode, writing to the CBnTX register with the CBnTSF bit set to 1 is ignored. This has no influence on the operation during transfer.	pp.458, 559 ☐
			PRSM1 register	Do not rewrite the PRSM1 register during operation.	p.461 ☐
				Set the PRSM1 register before setting the BGCE1 bit to 1.	p.461 ☐
			PRSCM1 register	Do not rewrite the PRSCM1 register during operation.	p.462 ☐
				Set the PRSCM1 register before setting the PRSM1.BGCE1 bit to 1.	p.462 ☐
			Baud rate generation	Set f_{BRG1} to 8 MHz or lower.	p.462 ☐

(15/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 14	Soft	3-wire variable-length serial I/O (CSIB)	When transferring transmit data and receive data using DMA transfer	When transferring transmit data and receive data using DMA transfer (μPD703253Y only), error processing cannot be performed even if an overrun error occurs during serial transfer. Check that the no overrun error has occurred by reading the CBnSTR.CBnOVE bit after DMA transfer has been completed.	p.463 ☐
			CBnCTL0 register CBnCTL1 register CBnCTL2 register	In regards to registers that are forbidden from being rewritten during operations (CBnCTL0.CBnPWR bit is 1), if rewriting has been carried out by mistake during operations, set the CBnCTL0.CBnPWR bit to 0 once, then initialize CSIBn. Registers to which rewriting during operation are prohibited are shown below. • CBnCTL0 register: CBnTXE, CBnRXE, CBnDIR, CBnTMS bits • CBnCTL1 register: CBnCKP, CBnDAP, CBnCKS2 to CBnCKS0 bits • CBnCTL2 register: CBnCL3 to CBnCL0 bits	p.463 ☐
			Communication types 2 and 4	In communication type 2 or 4 (CBnCTL1.CBnDAP bit = 1), the CBnSTR.CBnTSF bit is cleared half a $\overline{\text{SCKBn}}$ clock after occurrence of a reception complete interrupt (INTCBnR). In the single transfer mode, writing the next transmit data is ignored during communication (CBnTSF bit = 1), and the next communication is not started. Also if reception-only communication (CBnCTL0.CBnTXE bit = 0, CBnCTL0.CBnRXE bit = 1) is set, the next communication is not started even if the receive data is read during communication (CBnTSF bit = 1). Therefore, when using the single transfer mode with communication type 2 or 4 (CBnDAP bit = 1), pay particular attention to the following. • To start the next transmission, confirm that CBnTSF bit = 0 and then write the transmit data to the CBnTX register. • To perform the next reception continuously when reception-only communication (CBnTXE bit = 0, CBnRXE bit = 1) is set, confirm that CBnTSF bit = 0 and then read the CBnRX register. Or, use the continuous transfer mode instead of the single transfer mode. Use of the continuous transfer mode is recommended especially for using DMA (μPD703253Y only).	p.463 ☐
Chapter 15	Soft	I^2C bus	I^2C bus	To use the I^2C bus function, set the P38/SDA00, P39/SCL00, P40/SDA01, and P41/SCL01 pins as the serial transmit/receive data I/O pins (SDA00 and SDA01) and serial clock I/O pins (SCL00 and SCL01), and set them to N-ch open-drain output.	p.464 ☐
			CSIB0 and I^2C01 mode switching	The transmit/receive operation of CSIB0 and I^2C01 is not guaranteed if these functions are switched during transmission or reception. Be sure to disable the one that is not used.	p.464 ☐
			IICCn register	If the I^2Cn operation is enabled (IICEn bit = 1) when the SCL0n line is high level and the SDA0n line is low level, the start condition is detected immediately. To avoid this, after enabling the I^2Cn operation, immediately set the LRELn bit to 1 with a bit manipulation instruction.	p.471 ☐
			IICSn register	Accessing the IICSn register is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.475 ☐
			IICFn register	Write the STCENn bit only when operation is stopped (IICEn bit = 0).	p.479 ☐
				When the STCENn bit = 1, the bus released status (IICBSYn bit = 0) is recognized regardless of the actual bus status immediately after the I^2Cn bus operation is enabled. Therefore, to issue the first start condition (STTn bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.	p.479 ☐
				Write the IICRSVn bit only when operation is stopped (IICEn bit = 0).	p.479 ☐
			IICCLn register	Be sure to clear bits 7 and 6 to “0”.	p.480 ☐

(16/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 15	Soft	I^2C bus	Start condition	When the IICCn.IICEn bit of the V850ES/SG1 is set to 1 while communications with other devices are in progress, the start condition may be detected depending on the status of the communication line. Be sure to set the IICCn.IICEn bit to 1 when the SCL0n and SDA0n lines are high level.	p.486 ☐
			Status during arbitration and interrupt request signal generation timing	When there is a possibility that arbitration will occur, set the SPIEn bit to 1 for master device operation (n = 0, 1).	p.520 ☐
			When IICFn.STCENn bit = 0	Immediately after the I^2C0n operation is enabled, the bus communication status (IICFn.IICBSYn bit = 1) is recognized regardless of the actual bus status. To execute master communication in the status where a stop condition has not been detected, generate a stop condition and then release the bus before starting the master communication. Use the following sequence for generating a stop condition. <1> Set the IICCLn register. <2> Set the IICCn.IICEn bit. <3> Set the IICCn.SPTn bit.	p.526 ☐
			When IICFn.STCENn bit = 1	Immediately after I^2C0n operation is enabled, the bus released status (IICBSYn bit = 0) is recognized regardless of the actual bus status. To generate the first start condition (IICCn.STTn bit = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.	p.526 ☐
			While communications with other devices are in progress	When the IICCn.IICEn bit of the V850ES/SG1 is set to 1 while communications with other devices are in progress, the start condition may be detected depending on the status of the communication line. Be sure to set the IICCn.IICEn bit to 1 when the SCL0n and SDA0n lines are high level.	p.526 ☐
			Enabling operation	Determine the operation clock frequency by the IICCLn, IICXn, and OCKSn registers before enabling the operation (IICCn.IICEn bit = 1). To change the operation clock frequency, clear the IICCn.IICEn bit to 0 once.	p.526 ☐
			IICCn.STTn, SPTn bits	After the IICCn.STTn and IICCn.SPTn bits have been set to 1, they must not be re-set without being cleared to 0 first.	p.526 ☐
			Transmission reservation	If transmission has been reserved, set the IICCn.SPIEn bit to 1 so that an interrupt request is generated by the detection of a stop condition. After an interrupt request has been generated, the wait status will be released by writing communication data to I^2Cn, then transferring will begin. If an interrupt is not generated by the detection of a stop condition, transmission will halt in the wait status because an interrupt request was not generated. However, it is not necessary to set the SPIEn bit to 1 for the software to detect the IICSn.MSTSn bit.	p.526 ☐
			Master operation in single master system	Release the I^2C0n bus (SCL0n, SDA0n pins = high level) in conformity with the specifications of the product in communication. For example, when the EEPROM™ outputs a low level to the SDA0n pin, set the SCL0n pin to the output port and output clock pulses from that output port until when the SDA0n pin is constantly high level.	p.528 ☐
Chapter 16	Soft	IEBus controller	Effective transfer rate	Different modes must not be mixed on one IEBus.	p.543 ☐
			Data field	Do not operate master reception in broadcast communication, because the slave unit cannot be defined and data transfer cannot be performed correctly.	p.553 ☐

(17/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 16	Soft	IEBus controller	BCR register	While IEBus is operating as the master, writing to the BCR register (including bit manipulation instructions) is disabled until either the end of that communication or frame, or until communication is stopped by the occurrence of an arbitration-loss communication error. Master requests cannot therefore be multiplexed. However, the case when communication has been forcibly stopped (ENIEBUS flag = 0) is not a problem.	p.561 ☐
				If a bit manipulation instruction for the BCR register conflicts with a hardware reset of the MSTRQ bit, the BCR register may not operate normally. The following countermeasures are recommended in this case. • Because the hardware reset is instigated in the acknowledgment period of the slave address field, be sure to observe Caution 1 of (b) Master request flag (MSTRQ) below. • Be sure to observe the caution above regarding writing to the BCR register.	p.561 ☐
			Communication enable flag (ENIEBUS)	Before setting the ENIEBUS bit (1), the following registers must be set depending on the mode of communication to be started.	p.562 ☐
			Master request flag (MSTRQ)	If the IEBus controller has lost in arbitration, issue the master request again by software. In doing so, set (1) the MSTRQ bit at a timing other than that illustrated below.	p.563 ☐
				When a master request has been sent and bus mastership acquired, do not set the MSTRQ, ENSLVTX, or ENSLVRX bit until the end of communication (i.e. the communication end flag (ISR.ENDTRNS bit) or frame end flag (ISR.ENDFRAM bit) is set (1)) as setting these flags disables interrupt request signal generation. However, these flags can be set if communication has been aborted.	p.563 ☐
			Broadcast request flag of BCR register	When requesting broadcast communication, always set (1) the ALLRQ bit, then the MSTRQ bit.	p.563 ☐
			Slave transmission enable flag of BCR register	The ENSLVTX bit must be set before the parity bit in the control field is received.	p.564 ☐
				Clear the ENSLVTX bit (0) before setting the MSTRQ bit (1) when making a master request. This is to avoid transmission of the data of the DR register that tries master transmission if the controller loses in arbitration after master operation and if slave transmission is requested by the master.	p.564 ☐
			Slave reception enable flag of BCR register	The ENSLVRX bit must be set before the parity bit in the control field is received.	p.564 ☐
			PSR register	Do not set the PSR register while communication is enabled (BCR.ENIEBUS bit = 1).	p.565 ☐
				Be sure to clear bits 5 to 0 to “0”.	p.565 ☐
			Arbitration result flag of USR register	The flag is cleared (0) at the detection timing of the start bit if the other unit outputs the start bit earlier and the unit does not output the start bit after the master request.	p.568 ☐
			Lock status flag of USR register	Lock specification/release is not possible in broadcast communication. In the lock status, individual communication from a unit other than the one that requests locking is not acknowledged. However, even communication from a unit other than the one that requests locking is acknowledged as long as the communication is a slave status request.	p.569 ☐
			ESR register	Each bit can only be cleared (0). It cannot be set (1) even if 1 is written to it.	p.573 ☐
				The value of the ESR register is updated when an error occurs. If the ESR register is read at this time, however, an undefined value is read. It is recommended to read the ESR register in error interrupt servicing.	p.573 ☐

(18/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 16	Soft	IEBus controller	ESR register	If a communication error occurs, the IEBus controller returns to the default status and makes preparation for communication. If communication is started without the error corrected, the error flag accumulates the error. Correct the error before the next communication is started.	p.573 ☐
			Overrun error occurrence flag of ESR register	The overrun status is cleared only when the DR register is read and when the system is reset. Therefore, be sure to read the DR register in the communication error interrupt processing program.	p.576 ☐
				The next data cannot be transmitted in the overrun status if it is 2 bytes or more. Because the data request interrupt request signal (INTIE1) does not occur, the transmit data cannot be set and an underrun error occurs. Therefore, be sure to execute transmission after clearing the overrun status.	p.576 ☐
			Timing of write error occurrence	Even when the WERR bit is set (1), the INTIE1 interrupt request signal may be generated.	p.577 ☐
			UAR register	Do not set the UAR register while communication is enabled (BCR.ENIEBUS bit = 1).	p.578 ☐
			SAR register	Be sure to set the SAR register in the next timing. • When BCR.ENIEBUS bit = 0 • Between ENIEBUS bit = 1 and the first master request (set BCR.MSTRQ bit to (1)) • In ENIEBUS bit = 1, and MSTRQ bit = 0, between either the end of that communication, frame, or error and the next master request (set MSTRQ bit to (1))	p.578 ☐
			PAR register	The PAR register stores an address value if the parity is correct and the unit is not locked when the parity period of the master address field expires. If the PAR register is read at this time, an undefined value is read.	p.579 ☐
			RSA register	The RSA register stores an address value if the parity is correct and the unit is not locked when the parity period of the slave address field expires. If the RSA register is read at this time, an undefined value is read.	p.579 ☐
			CDR register	Because the slave unit must judge whether the received data is a “command” or “data”, read the value of the CDR register after completing communication.	p.581 ☐
				If the master unit sets an undefined value, the slave unit returns the NACK signal and communication is aborted. During broadcast communication, the master unit ignores the acknowledge bit and continues communication. Therefore, do not set an undefined value.	p.581 ☐
			DLR register	When the master issues a request (0H, 4H, 5H, or 6H) for transmission of a slave status or a lock address (higher 4 bits and lower 8 bits), 01H is transmitted as the telegraph length regardless of the contents of the DLR register. It is therefore not necessary to set the DLR register by software.	p.585 ☐
				When the IEBus controller serves as a receiver unit, the DLR register stores a telegraph length if the value of the parity bit of the telegraph length field is correct. If the DLR register is read at this time, an undefined value is read.	p.585 ☐
			DR register	When the IEBus controller serves as a receiver unit, the DR register stores receive data if the value of the parity bit of the data field is correct. If the DR register is read at this time, an undefined value is read.	p.586 ☐
			FSR register	The FSR register updates the status information when an interrupt request signal is generated. If the FSR register is read at this time, however, an undefined value is read.	p.587 ☐
				If another interrupt request signal is generated before the FSR register is read, the status information when the preceding interrupt occurred is updated by the status information when the new interrupt occurs.	p.587 ☐
				Use the FSR register only for problem analysis; do not use it with the actual software.	p.587 ☐

(19/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 16	Soft	IEBus controller	SCR register	The SCR register is updated when the parity period of the telegraph field expires and when the $\overline{\text{ACK}}$ signal of the data field is received. If the SCR register is read at this time, however, an undefined value is read.	p.588 ☐
			CCR register	The maximum number of transmit bytes is preset to the CCR register when the start bit is transmitted or received, and the register is decremented when the parity period of the data field expires. If the CCR register is read at this time, however, an undefined value is read.	p.589 ☐
Chapter 17	Soft	CAN controller	Programmable peripheral I/O area setting	The CAN controller is allocated in the programmable peripheral I/O area. Before using the CAN controller, enable use of the programmable peripheral I/O area by using the BPC register. For details, refer to 3.4.7 Programmable peripheral I/O registers.	p.612 ☐
			Arbitration field (in standard format mode)	An identifier is transmitted MSB first.	p.618 ☐
			Arbitration field (in extended format mode)	An identifier is transmitted MSB first.	p.618 ☐
			Data length setting	In the remote frame, there is no data field even if the data length code is not 0000B.	p.619 ☐
			Forced recovery operation that skips bus-off recovery sequence	This function is not defined by the CAN protocol ISO 11898. When using this function, thoroughly evaluate its effect on the network system.	p.631 ☐
			Initializing CAN module error counter register in initialization mode	This function is enabled only in the initialization mode. Even if the CCERC bit is set to 1 in a CAN operation mode, the C0ERC and C0INFO registers are not initialized.	p.631 ☐
			Register	Accessing the CAN controller registers is prohibited in the following statuses. For details, refer to 3.4.9 (2) Accessing specific on-chip peripheral I/O registers. • When the CPU operates with the subclock and the main clock oscillation is stopped • When the CPU operates with the internal oscillation clock	p.659 ☐
			C0GMCTRL register	While the MBON bit is cleared (to 0), software access to the message buffers (C0MDATA0m, C0MDATA1m, C0MDATA01m, C0MDATA2m, C0MDATA3m, C0MDATA23m, C0MDATA4m, C0MDATA5m, C0MDATA45m, C0MDATA6m, C0MDATA7m, C0MDATA67m, C0MDLCm, C0MCONFm, C0MIDLm, C0MIDHm, and C0MCTRLm), or registers related to transmit history or receive history (C0LOPT, C0TGPT, C0LIPT, and C0RGPT) is disabled.	p.659 ☐
				To request forced shut down, clear the GOM bit to 0 immediately after the EFSD bit has been set to 1. If access to another register (including reading the C0GMCTRL register) is executed without clearing the GOM bit immediately after the EFSD bit has been set to 1, the EFSD bit is forcibly cleared to 0, and the forced shut down request is invalid.	p.660 ☐
				The GOM bit is cleared to 0 only in the initialization mode or immediately after the EFSD bit is set to 1.	p.660 ☐
				Be sure to set the GOM bit and EFSD bit separately.	p.660 ☐

(20/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 17	Soft	CAN controller	C0GMABT register	Before changing the normal operation mode with ABT to the initialization mode, be sure to set the C0GMABT register to the default value (0000H). After setting, confirm that the C0GMABT register is initialized to 0000H.	p.662 ☐
				Do not set the ABTTRG bit to 1 in the initialization mode. If the ABTTRG bit is set to 1 in the initialization mode, the operation is not guaranteed after the CAN module has entered the normal operation mode with ABT.	p.662 ☐
				Do not set the ABTTRG bit to 1 while the C0CTRL.TSTAT bit is set to 1. Directly confirm that the TSTAT bit = 0 before setting the ABTTRG bit to 1.	p.662 ☐
			C0GMABTD register	Do not change the contents of the C0GMABTD register while the ABTTRG bit is set to 1.	p.664 ☐
			C0CTRL register	Transition to and from the CAN stop mode must be made via CAN sleep mode. A request for direct transition to and from the CAN stop mode is ignored.	p.669 ☐
				After releasing the power save mode, the C0GMCTRL.MBON flag must be checked before accessing the message buffer again.	p.669 ☐
				It may take time to change the mode to the initialization mode or power save mode. Therefore, be sure to check if the mode has been successfully changed, by reading the register value before executing the processing.	p.669 ☐
			C0INTS register	The status bit of this register is not automatically cleared. Clear (0) it by software if each status must be checked in the interrupt servicing.	p.676 ☐
			C0BRP register	The C0BRP register can be write-accessed only in the initialization mode.	p.677 ☐
			C0MDLCm register	Be sure to set bits 7 to 4 to 0000B.	p.686 ☐
			C0MCONFm register	Be sure to write 0 to bits 2 and 1.	p.688 ☐
			C0MIDLm, C0MIDHm registers	Be sure to write 0 to bits 14 and 13 of the C0MIDHm register.	p.688 ☐
				Be sure to arrange the ID values to be registered in accordance with the bit positions of this register. For the standard ID, shift the bit positions of ID28 to ID18 of the ID value.	p.688 ☐
			C0MCTRLm register	Do not set the TRQ bit and RDY bit to 1 at the same time. Be sure to set the RDY bit to 1 before setting the TRQ bit to 1.	p.690 ☐
				Do not clear (0) the RDY bit during message transmission. Follow transmission abort procedures in order to clear the RDY bit for redefinition.	p.690 ☐
				If the RDY bit is not cleared (0) even when the processing to clear it is executed, execute the clearing processing again.	p.690 ☐
				Confirm, by reading the RDY bit again, that the RDY bit has been cleared (0) before writing data to the message buffer. However, it is unnecessary to confirm that the TRQ or RDY bit has been set (1) or that the DN or MOW bit has been cleared (0).	p.690 ☐
				Be sure to set the IE and RDY bits separately.	p.690 ☐
				Do not set the DN bit to 1 by software. Be sure to write 0 to bit 10.	p.690 ☐
				Be sure to set the TRQ and RDY bits separately.	p.691 ☐
			Message buffer	When a message is received, reception filtering is performed in accordance with the ID and mask set to each receive message buffer. If the procedure in Figure 17-39 is not observed, the contents of the message buffer after it has been redefined may contradict the result of reception (result of reception filtering). If this happens, check that the ID and IDE received first and stored in the message buffer following redefinition are those stored after the message buffer has been redefined. If no ID and IDE are stored after redefinition, redefine the message buffer again.	p.695 ☐

(21/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 17	Soft	CAN controller	Message buffer	When a message is transmitted, the transmission priority is checked in accordance with the ID, IDE, and RTR bits set to each transmit message buffer to which a transmission request was set. The transmit message buffer having the highest priority is selected for transmission. If the procedure in Figure 17-27 is not observed, a message with an ID not having the highest priority may be transmitted after redefinition.	p.695 ☐
			Receive history list function	Even if the receive history list overflows (C0RGPT.ROVF bit = 1), the receive history can be read until no more history is left unread and the C0RGPT.RHPM bit is set (1). However, the ROVF bit is kept set (1) (= overflow occurs) until cleared (0) by software. In this status, the RHPM bit is not cleared (0), unless the ROVF bit is cleared (0), even if a new receive history is stored and written to the list. If ROVF bit = 1 and RHPM bit = 1 and the receive history list overflows, therefore, the RHPM bit indicates that no more history is left unread even if new receive history is stored.	p.699 ☐
			Multi buffer receive block function	MBRB can be configured for each of the same message buffer types. Therefore, even if a message buffer of another MBRB whose ID matches but whose message buffer type is different has a vacancy, the received message is not stored in that message buffer, but instead discarded.	p.703 ☐
			Transmit history list function	If the TOVF bit = 1 and the THPM bit = 1 and the transmit history list overflows, therefore, the THPM bit indicates that no more history is left unread even if new transmit history is stored.	p.707 ☐
			Automatic block transmission	To resume the normal operation mode with ABT from the message buffer 0, set the ABTCLR bit to 1 while the ABTTRG bit is cleared to 0. If the ABTCLR bit is set to 1 while the ABTTRG bit is set to 1, the subsequent operation is not guaranteed.	p.710 ☐
				Do not set the ABTTRG bit in the initialization mode. If the ABTTRG bit is set in the initialization mode, the proper operation is not guaranteed after the mode is changed from the initialization mode to the ABT mode.	p.710 ☐
				Do not set the TRQ bit of the ABT message buffers to 1 by software in the normal operation mode with ABT. Otherwise, the operation is not guaranteed.	p.710 ☐
				Do not clear the RDY bit to 0 when the ABTTRG bit = 1.	p.710 ☐
				If a message is received from another node in the normal operation mode with ABT, the message may be transmitted after the time of one frame has elapsed even when C0GMABTD register = 00H.	p.710 ☐
			Transmission abort in normal operation mode with automatic block transmission	Be sure to abort ABT by clearing the ABTTRG bit to 0. The operation is not guaranteed if aborting transmission is requested by clearing RDY.	p.711 ☐
			Releasing CAN sleep mode	Even if the falling edge belongs to the SOF of a receive message, this message will not be received and stored. If the CPU has turned off the clock to the CAN while the CAN was in sleep mode, later on the CAN sleep mode will not be released and PSMODE1 and PSMODE0 bits will continue to be 01B unless the clock for the CAN is provided again. In addition to this, the receive message will not be received afterwards.	p.713 ☐
				If the falling edge is detected on the CAN reception pin (CRXD0) while the CAN clock is supplied, the PSMODE0 bit must be cleared by software (for details, refer to the processing in Figure 17-52).	p.713 ☐

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 17	Soft	CAN controller	Releasing CAN sleep mode	When the CAN sleep mode is released by an event of the CAN bus, a wakeup interrupt occurs even if the event of the CAN bus occurs immediately after the mode has been changed to the sleep mode. Note that the interrupt can occur at any time.	p.713 ☐
			Entering CAN stop mode	To set the CAN module to the CAN stop mode, the module must be in the CAN sleep mode. To confirm that the module is in the sleep mode, check that the PSMODE1 and PSMODE0 bits = 01B, and then request the CAN stop mode. If a bus change occurs at the CAN reception pin (CRXD0) while this process is being performed, the CAN sleep mode is automatically released. In this case, the CAN stop mode transition request cannot be acknowledged (when the CAN clock is supplied, however, the PSMODE0 bit must be cleared by software after a bus change occurs at the CAN reception pin (CRXD0)).	p.714 ☐
			Receive-only mode	If only two CAN nodes are connected to the CAN bus and one of them is operating in the receiveonly mode, there is no ACK on the CAN bus. Due to the missing ACK, the transmitting node will transmit an active error flag, and repeat transmitting a message frame. The transmitting node becomes error passive after transmitting the message frame 16 times (assuming that the error counter was 0 in the beginning and no other errors have occurred). When the message frame is transmitted for the 17th time, the transmitting node generates a passive error flag. The receiving node in the receive-only mode detects the first valid message frame at this point, and the VALID bit is set to 1 for the first time.	p.718 ☐
			Settable bit rate combinations	The values in Table 17-22 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.	p.724 ☐
				The values in Table 17-23 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.	pp.725, 726 ☐
				The values in Table 17-24 do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.	pp.727, 728 ☐
			Re-initialization	After setting the CAN module to the initialization mode, avoid setting the module to another operation mode immediately after. If it is necessary to immediately set the module to another operation mode, be sure to access registers other than the C0CTRL and C0GMCTRL registers (e.g., set a message buffer).	p.730 ☐
			Message buffer initialization	Before a message buffer is initialized, the RDY bit must be cleared.	p.731 ☐
				Make the following settings for message buffers not used by the application. • Clear the C0MCTRLm.RDY, C0MCTRLm.TRQ, and C0MCTRLm.DN bits to 0. • Clear the C0MCONFm.MA0 bit to 0.	p.731 ☐
			Message transmit processing	The TRQ bit should be set after the RDY bit is set.	p.734 ☐
				The RDY bit and TRQ bit should not be set at the same time.	p.734 ☐
			ABT message transmit processing	The ABTTRG bit should be set to 1 after the TSTAT bit is cleared to 0. The checking of the TSTAT bit and the setting for the ABTTRG bit to 1 must be continuous.	p.735 ☐
			Transmission via interrupt (Using C0LOPT register)	The TRQ bit should be set after the RDY bit is set.	p.736 ☐
				The RDY bit and TRQ bit should not be set at the same time.	p.736 ☐

(23/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 17	Soft	CAN controller	Transmission via interrupt (Using C0TGPT register)	The TRQ bit should be set after the RDY bit is set.	p.737 ☐
				The RDY bit and TRQ bit should not be set at the same time.	p.737 ☐
				If the TOVF bit is set (1) again, the transmit history list contradicts. Therefore, scan all the transmit message buffers that have completed transmission.	p.737 ☐
			Transmission via software polling	The TRQ bit should be set after the RDY bit is set.	p.738 ☐
				The RDY bit and TRQ bit should not be set at the same time.	p.738 ☐
				If the TOVF bit is set (1) again, the transmit history list contradicts. Therefore, scan all the transmit message buffers that have completed transmission.	p.738 ☐
			Transmission abort processing (Other than in normal operation mode with ABT)	Execute transmission abort processing by clearing the TRQ bit, not the RDY bit.	p.739 ☐
				Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.	p.739 ☐
				The TSTAT bit can be periodically checked by a user application or can be checked after the transmit completion interrupt.	p.739 ☐
				Do not execute a new transmission request that includes other message buffers while transmission abort processing is in progress.	p.739 ☐
				If data of the same message buffer are successively transmitted or if only one message buffer is used, judgments whether transmission has been successfully executed or failed may contradict. In such a case, make a judgment by using the history information of the C0TGPT register.	p.739 ☐
			Transmission abort processing (Other than in normal operation mode with ABT)	Execute transmission abort processing by clearing the TRQ bit, not the RDY bit.	p.740 ☐
				Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.	p.740 ☐
				Do not execute a new transmission request that includes other message buffers while transmission abort processing is in progress.	p.740 ☐
				If data of the same message buffer are successively transmitted or if only one message buffer is used, judgments whether transmission has been successfully executed or failed may contradict. In such a case, make a judgment by using the history information of the C0TGPT register.	p.740 ☐
			Transmission abort processing except for ABT transmission (Normal operation mode with ABT)	Do not set any transmission requests while ABT transmission abort processing is in progress.	pp.741, 742 ☐
				Make a CAN sleep mode/CAN stop mode transition request after the ABTTRG bit is cleared (after ABT mode is stopped) following the procedure shown in Figure 17-47 (a) or (b). When clearing a transmission request in an area other than the ABT area, follow the procedure shown in Figure 17-45.	pp.741, 742 ☐
			Reception via interrupt processing (Using C0LIPT register)	Check the MBON bit at the start and end of the interrupt routine to see if the message buffer and receive history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again. It is therefore recommended to cancel the CAN sleep mode transition request before executing reception interrupt servicing.	p.743 ☐
			Reception via interrupt processing (Using C0RGPT register)	Check the MBON bit at the start and end of the interrupt routine to see if the message buffer and receive history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again. It is therefore recommended to cancel the CAN sleep mode transition request before executing reception interrupt servicing.	p.744 ☐
				If the ROVF bit has been once set (1), the receive history list contradicts. Therefore, scan all the receive message buffers that have completed reception.	p.744 ☐

(24/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 17	Soft	CAN controller	Reception via software polling processing	Check the MBON bit at the start and end of the polling routine to see if the message buffer and receive history register can be accessed, because a CAN sleep mode transition request which has been held pending may be under execution. If the MBON bit is cleared (0), stop the processing under execution. Re-execute the processing after the MBON bit is set (1) again.	p.745 ☐
				If the ROVF bit has been once set (1), the receive history list contradicts. Therefore, scan all the receive message buffers that have completed reception.	p.745 ☐
			Setting CAN sleep mode/stop mode	To abort transmission before making a request for the CAN sleep mode, perform processing according to Figures 17-45 to 17-47.	p.746 ☐
			Bus-off recovery	If a request to change the mode from the initialization mode to any operation mode is made to execute the bus-off recovery sequence again during a bus-off recovery sequence, the receive error counter (C0ERC.REC0 to REC6 bits) is cleared. It is therefore necessary to detect 11 contiguous recessive bits 128 times on the bus again.	pp.748, 749 ☐
			Forced shutdown process	Do not read- or write-access any registers by software between setting the EFSD bit and clearing the GOM bit.	p.751 ☐
			Setting CPU standby (from CAN sleep mode)	Check if the CPU is in the CAN sleep mode before setting it to the standby mode. The CAN sleep mode may be released by wakeup after it is checked if the CPU is in the CAN sleep mode and before the CPU is set in the standby mode.	p.753 ☐
			Setting CPU standby (from CAN stop mode)	The CAN stop mode can only be released by writing 01 to the C0CTRL.PSMODE1 and C0CTRL.PSMODE0 bits. It cannot be released by changing the CAN bus.	p.754 ☐
Chapter 18	Soft	DMA function (DMA controller)	DSAn register	Be sure to clear bits 14 to 10 of the DSAnH register to 0.	p.757 ☐
				Set the DSAnH and DSAnL registers at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0). • Period from after reset to start of first DMA transfer • Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer • Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer	p.757 ☐
				When the value of the DSAn register is read, two 16-bit registers, DSAnH and DSAnL, are read. If reading and updating conflict, the value being updated may be read (see 18.13 Cautions).	p.757 ☐
				Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer. If these registers are not set, the operation when DMA transfer is started is not guaranteed.	p.757 ☐
			DDAn register	Be sure to clear bits 14 to 10 of the DDAnH register to 0.	p.758 ☐
				Set the DDAnH and DDAnL registers at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0). • Period from after reset to start of first DMA transfer • Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer • Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer	p.758 ☐
				When the value of the DDAn register is read, two 16-bit registers, DDAnH and DDAnL, are read. If reading and updating conflict, a value being updated may be read (see 18.13 Cautions).	p.758 ☐

(25/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 18	Soft	DMA function (DMA controller)	DDAn register	Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer. If these registers are not set, the operation when DMA transfer is started is not guaranteed.	p.758 ☐
			DBCn register	Set the DBCn register at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0). • Period from after reset to start of first DMA transfer • Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer • Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer	p.759 ☐
				Following reset, set the DSAnH, DSAnL, DDAnH, DDAnL, and DBCn registers before starting DMA transfer. If these registers are not set, the operation when DMA transfer is started is not guaranteed.	p.759 ☐
			DADCn register	Be sure to clear bits 15, 13 to 8, and 3 to 0 of the DADCn register to “0”.	p.760 ☐
				Set the DADCn register at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0). • Period from after reset to start of first DMA transfer • Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer • Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer	p.760 ☐
				The DS0 bit specifies the size of the transfer data, and does not control bus sizing. If 8-bit data (DS0 bit = 0) is set, therefore, the lower data bus is not always used.	p.760 ☐
				If the transfer data size is set to 16 bits (DS0 bit = 1), transfer cannot be started from an odd address. Transfer is always started from an address with the first bit of the lower address aligned to 0.	p.760 ☐
				If DMA transfer is executed on an on-chip peripheral I/O register (as the transfer source or destination), be sure to specify the same transfer size as the register size. For example, to execute DMA transfer on an 8-bit register, be sure to specify 8-bit transfer.	p.760 ☐
			DCHCn register	Be sure to clear bits 6 to 3 of the DCHCn register to 0.	p.761 ☐
				When DMA transfer is completed (when a terminal count is generated), the Enn bit is cleared to 0 and then the TCn bit is set to 1. If the DCHCn register is read while its bits are being updated, a value indicating “transfer not completed and transfer is disabled” (TCn bit = 0 and Enn bit = 0) may be read.	p.761 ☐
			DTFR0 register	Set the IFCn5 to IFCn0 bits at the following timing when DMA transfer is disabled (DCHCn.Enn bit = 0). • Period from after reset to start of first DMA transfer • Period from after channel initialization by DCHCn.INITn bit to start of DMA transfer • Period from after completion of DMA transfer (DCHCn.TCn bit = 1) to start of the next DMA transfer	p.762 ☐
				Be sure to follow the steps shown in the Caution 2 in 18.3 (6) when changing the DTFRn register settings.	p.763 ☐
				An interrupt request that is generated in the standby mode (IDEL1, IDLE2, STOP, or sub-IDLE mode) does not start the DMA transfer cycle (nor is the DFn bit set to 1).	p.763 ☐
				If a DMA start factor is selected by the IFCn5 to IFCn0 bits, the DFn bit is set to 1 when an interrupt occurs from the selected on-chip peripheral I/O, regardless of whether the DMA transfer is enabled or disabled. If DMA is enabled in this status, DMA transfer is immediately started.	p.763 ☐
			Transfer targets	The operation is not guaranteed for combinations of transfer destination and source marked with “×” in Table 18-2.	p.765 ☐

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 18	Soft	DMA function (DMA controller)	DMA transfer start factors	Two start factors (software trigger and hardware trigger) cannot be used for one DMA channel. If two start factors are simultaneously generated for one DMA channel, only one of them is valid. The start factor that is valid cannot be identified.	p.768 ☐
				A new transfer request that is generated after the preceding DMA transfer request was generated or in the preceding DMA transfer cycle is ignored (cleared).	p.768 ☐
				Therefore, the transfer request intervals for the same DMA channel must be sufficiently separated by the system. When the software trigger is used, completion of the DMA transfer cycle that was generated before can be checked by updating the DBCn register.	p.768 ☐
			Caution for VSWC register	When using the DMAC, be sure to set an appropriate value, in accordance with the operating frequency, to the VSWC register. When the default value (77H) of the VSWC register is used, or if an inappropriate value is set to the VSWC register, the operation is not correctly performed (for details of the VSWC register, see 3.4.9 (1) (a) System wait control register (VSWC)).	p.774 ☐
			Caution for DMA transfer executed on internal RAM	When executing the following instructions located in the internal RAM, do not execute a DMA transfer that transfers data to/from the internal RAM (transfer source/destination), because the CPU may not operate correctly afterward. • Bit manipulation instruction located in internal RAM (SET1, CLR1, or NOT1) • Data access instruction to misaligned address located in internal RAM Conversely, when executing a DMA transfer to transfer data to/from the internal RAM (transfer source/destination), do not execute the above two instructions.	p.774 ☐
			Caution for reading DCHCn.TCn bit (n = 0 to 3)	The TCn bit is cleared to 0 when it is read, but it is not automatically cleared even if it is read at a specific timing. To accurately clear the TCn bit, add the following processing. (a) When waiting for completion of DMA transfer by polling TCn bit Confirm that the TCn bit has been set to 1 (after TCn bit = 1 is read), and then read the TCn bit three more times. (b) When reading TCn bit in interrupt servicing routine Execute reading the TCn bit three times.	p.774 ☐
			DMA transfer initialization procedure (setting DCHCn.INITn bit to 1)	Even if the INITn bit is set to 1 when the channel executing DMA transfer is to be initialized, the channel may not be initialized. To accurately initialize the channel, execute either of the two procedures shown in 18.13 (4) (a) and (b).	pp.775, 776 ☐
			Procedure of temporarily stopping DMA transfer (clearing Enn bit)	Stop and resume the DMA transfer under execution using the following procedure. <1> Suppress a transfer request from the DMA request source (stop the operation of the on-chip peripheral I/O). <2> Check the DMA transfer request is not held pending, by using the DFn bit (check if the DFn bit = 0). If a request is pending, wait until execution of the pending DMA transfer request is completed. <3> If it has been confirmed that no DMA transfer request is held pending, clear the Enn bit to 0 (this operation stops DMA transfer). <4> Set the Enn bit to 1 to resume DMA transfer. <5> Resume the operation of the DMA request source that has been stopped (start the operation of the onchip peripheral I/O).	p.776 ☐
			Memory boundary	The operation is not guaranteed if the address of the transfer source or destination exceeds the area of the DMA target (external memory, internal RAM, or on-chip peripheral I/O) during DMA transfer.	p.776 ☐

(27/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 18	Soft	DMA function (DMA controller)	Transferring misaligned data	DMA transfer of misaligned data with a 16-bit bus width is not supported. If an odd address is specified as the transfer source or destination, the least significant bit of the address is forcibly assumed to be 0.	p.776 ☐
			Registers/bits that must not be rewritten during DMA operation	Set the following registers at the following timing when a DMA operation is not under execution. [Registers] • DSAnH, DSAnL, DDAnH, DDAnL, DBCn, and DADCn registers • DTFRn.IFCn5 to DTFRn.IFCn0 bits [Timing of setting] • Period from after reset to start of the first DMA transfer • Time after channel initialization to start of DMA transfer • Period from after completion of DMA transfer (TCn bit = 1) to start of the next DMA transfer	p.777 ☐
			Be sure to set the following register bits to 0.	Be sure to set the following register bits to 0. • Bits 14 to 10 of DSAnH register • Bits 14 to 10 of DDAnH register • Bits 15, 13 to 8, and 3 to 0 of DADCn register • Bits 6 to 3 of DCHCn register	p.777 ☐
			DMA start factor	Care must be exercised when setting the same start trigger for multiple DMA channels. If DMA transfers via such DMA channels are activated, the DMA channel with a lower priority may be acknowledged prior to the DMA channel with a higher priority.	p.777 ☐
			Read values of DSAn and DDAn registers	Values in the middle of updating may be read from the DSAn and DDAn registers during DMA transfer (n = 0 to 3).	p.778 ☐
Chapter 19	Soft	Interrupt/ exception processing function	Non-maskable interrupts	For the non-maskable interrupt servicing executed by the non-maskable interrupt request signal (INTWDT2), see 19.2.2 (2) INTWDT2 signal.	p.783 ☐
				When the EP and NP bits are changed by the LDSR instruction during non-maskable interrupt servicing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 0 and the NP bit back to 1 using the LDSR instruction immediately before the RETI instruction.	p.786 ☐
			Maskable interrupts	When the EP and NP bits are changed by the LDSR instruction during maskable interrupt servicing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 0 and the NP bit back to 0 using the LDSR instruction immediately before the RETI instruction.	p.790 ☐
				To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.	p.792 ☐
			xxICn register	Disable interrupts (DI) or mask the interrupt to read the xxICn.xxIFn bit. If the xxIFn bit is read while interrupts are enabled (EI) or while the interrupt is unmasked, the correct value may not be read when acknowledging an interrupt and reading the bit conflict.	p.795 ☐
				When manipulating the xxICn.xxMKn bit while interrupt requests may occur (including the state in which interrupts are disabled (DI)), be sure to use a bit manipulation instruction or use the IMRm.xxMKn bit (m = 0 to 3).	p.795 ☐

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 19	Soft	Interrupt/ exception processing function	IMR0 to IMR3 registers	Set bits 0 and 9 to 13 of the IMR0 register, bits 5, 6, and 14 of the IMR2 register, and bits 11 to 15 of the IMR3 register to “1”. If the setting of these bits is changed, the operation is not guaranteed.	p.798 ☐
			ISPR register	If an interrupt is acknowledged while the ISPR register is being read in the interrupt enabled (EI) status, the value of the ISPR register after the bits of the register have been set by acknowledging the interrupt may be read. To accurately read the value of the ISPR register before an interrupt is acknowledged, read the register while interrupts are disabled (DI).	p.799 ☐
			Software exception	When the EP and NP bits are changed by the LDSR instruction during the software exception processing, in order to restore the PC and PSW correctly during recovery by the RETI instruction, it is necessary to set the EP bit back to 1 and the NP bit back to 0 using the LDSR instruction immediately before the RETI instruction.	p.802 ☐
			Exception trap	Since it is possible to assign this instruction to an illegal opcode in the future, it is recommended that it not be used.	p.804 ☐
				DBPC and DBPSW can be accessed only during the interval between the execution of the illegal opcode and DBRET instruction.	p.805 ☐
				DBPC and DBPSW can be accessed after the DBTRAP instruction is executed and before the DBRET instruction is executed.	p.807 ☐
			INTF0, INTR0 registers	When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF0n and INTR0n bits to 00, and then set the port mode.	p.809 ☐
				Be sure to clear the INTF0n and INTR0n bits to 00 when these registers are not used as the NMI or INTP0 to INTP3 pins.	p.809 ☐
			INTF3, INTR3 registers	When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF31 and INTR31 bits to 00, and then set the port mode.	p.810 ☐
				The INTP7 pin and RXDA0 pin are alternate-function pins. When using the pin as the RXDA0 pin, disable edge detection for the INTP7 alternate-function pin (clear the INTF3.INTF31 bit and the INRT3.INTR31 bit to 0). When using the pin as the INTP7 pin, stop UARTA0 reception (clear the UA0CTL0.UA0RXE bit to 0).	p.810 ☐
				Be sure to clear the INTF31 and INTR31 bits to 00 when these registers are not used as INTP7 pin.	p.810 ☐
			INTF9H, INTR9H registers	When the function is changed from the external interrupt function (alternate function) to the port function, an edge may be detected. Therefore, clear the INTF9n and INTR9n bits to 0, and then set the port mode.	p.811 ☐
				Be sure to clear the INTF9n and INTR9n bits to 00 when these registers are not used as INTP4 to INTP6 pins.	p.811 ☐
			NFC register	Therefore, be careful about the following points when using the interrupt and DMA functions (μPD703253Y only). • When using the interrupt function, after the 3 sampling clocks have elapsed, enable interrupts after the interrupt request flag (PIC3.PIF3 bit) has been cleared. • When using the DMA function (started by INTP3) in the μPD703253Y, enable DMA after 3 sampling clocks have elapsed.	p.812 ☐
			NMI pin	The NMI pin and P02 pin are an alternate-function pin, and function as a normal port pin after being reset. To enable the NMI pin, validate the NMI pin with the PMC0 register. The initial setting of the NMI pin is “No edge detected”. Select the NMI pin valid edge using the INTF0 and INTR0 registers.	p.815 ☐

(29/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 19	Soft	Interrupt/ exception processing function	xxICn register	When manipulating the xxICn.xxMKn bit while interrupt requests may occur (including the state in which interrupts are disabled (DI)), be sure to use a bit manipulation instruction or use the IMRm.xxMKn bit (m = 0 to 3).	p.815 ☐
			ISPR register	If an interrupt is acknowledged while the ISPR register is being read in the interrupt enabled (EI) status, the value of the ISPR register after the bits of the register have been set by acknowledging the interrupt may be read. To accurately read the value of the ISPR register before an interrupt is acknowledged, read the register while interrupts are disabled (DI).	p.815 ☐
Chapter 20	Soft	Key interrupt function	KRM register	Rewrite the KRM register after once clearing the KRM register to 00H.	p.817 ☐
	Hard		KR0 to KR7 pins	If a low level is input to any of the KR0 to KR7 pins, the INTKR signal is not generated even if the falling edge of another pin is input.	p.817 ☐
	Hard, soft		RXDA1 pin KR7 pin	The RXDA1 and KR7 pins must not be used at the same time. To use the RXDA1 pin, do not use the KR7 pin. To use the KR7 pin, do not use the RXDA1 pin (it is recommended to set the PFC91 bit to 1 and clear PFCE91 bit to 0).	p.817 ☐
	Soft		KRM register	If the KRM register is changed, an interrupt request signal (INTKR) may be generated. To prevent this, change the KRM register after disabling interrupts (DI) or masking, then clear the interrupt request flag (KRIC.KRIF bit) to 0, and enable interrupts (EI) or clear the mask.	p.817 ☐
			Port pin	To use the key interrupt function, be sure to set the port pin to the key return pin and then enable the operation with the KRM register. To switch from the key return pin to the port pin, disable the operation with the KRM register and then set the port pin.	p.817 ☐
Chapter 21	Soft	Standby function	PSC register	Before setting the IDLE1, IDLE2, STOP, or sub-IDLE mode, set the PSMR.PSM1 and PSMR.PSM0 bits and then set the STP bit.	p.820 ☐
				If there is an unmasked interrupt request signal being held pending when the IDLE1/IDLE2/STOP mode is set, set the bit corresponding to the interrupt request signal (NMI1M, NMI0M, or INTM) to 1, and then set the STP bit to 1.	p.820 ☐
			PSMR register	Be sure to clear bits 2 to 7 to “0”.	p.821 ☐
				The PSM0 and PSM1 bits are valid only when the PSC.STP bit is 1.	p.821 ☐
			OSTS register	Be sure to clear bits 3 to 7 to “0”.	p.822 ☐
				The oscillation stabilization time following reset release is $2^{16}/f_X$ (because the initial value of the OSTS register = 06H).	p.822 ☐
			HALT mode	Insert five or more NOP instructions after the HALT instruction.	p.823 ☐
				If the HALT instruction is executed while an unmasked interrupt request signal is being held pending, the status shifts to HALT mode, but the HALT mode is then released immediately by the pending interrupt request.	p.823 ☐
			IDLE1 mode	Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE1 mode.	p.825 ☐
				If the IDLE1 mode is set while an unmasked interrupt request signal is being held pending, the IDLE1 mode is released immediately by the pending interrupt request.	p.825 ☐
			IDLE2 mode	Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the IDLE2 mode.	p.827 ☐
				If the IDLE2 mode is set while an unmasked interrupt request signal is being held pending, the IDLE2 mode is released immediately by the pending interrupt request.	p.827 ☐
			STOP mode	Insert five or more NOP instructions after the instruction that stores data in the PSC register to set the STOP mode.	p.830 ☐
				If the STOP mode is set while an unmasked interrupt request signal is being held pending, the STOP mode is released immediately by the pending interrupt request.	p.830 ☐

(30/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 21	Soft	Standby function	Subclock operation mode	When manipulating the CK3 bit, do not change the set values of the PCC.CK2 to PCC.CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended). For details of the PCC register, see 6.3 (1) Processor clock control register (PCC).	p.834 ☐
				If the following conditions are not satisfied, change the CK2 to CK0 bits so that the conditions are satisfied and set the subclock operation mode. Internal system clock (f_{CLK}) > Subclock (f_{XT} = 32.768 kHz) × 4	p.834 ☐
				When manipulating the CK3 bit, do not change the set values of the CK2 to CK0 bits (using a bit manipulation instruction to manipulate the bit is recommended). For details of the PCC register, see 6.3 (1) Processor clock control register (PCC).	p.834 ☐
				When the CPU is operating on the subclock and main clock oscillation is stopped, accessing a register in which a wait occurs is disabled. If a wait is generated, it can be released only by reset (see 3.4.9 (2)).	p.835 ☐
			Sub IDLE mode	Following the store instruction to set the PSC register to the sub-IDLE mode, insert the five or more NOP instructions.	p.836 ☐
				If the sub-IDLE mode is set while an unmasked interrupt request signal is being held pending, the sub-IDLE mode is then released immediately by the pending interrupt request.	p.836 ☐
				When the sub-IDLE mode is released, 12 cycles of the subclock (about 366 μs) elapse from when the interrupt request signal that releases the sub-IDLE mode is generated to when the mode is released.	p.836 ☐
Chapter 22	Soft	Reset function	Emergency operation mode	When the CPU operates on the internal oscillation clock, access to the register in which await state is generated is prohibited. For the register in which a wait state is generated, see 3.4.9 (2) Accessing specific on-chip peripheral I/O registers.	p.838 ☐
			Reset operation via $\overline{\text{RESET}}$ pin	The OCDM register is initialized by the $\overline{\text{RESET}}$ pin input. Therefore, if a high level is input to the P05/INTP2 pin after a reset release before the OCDM.OCDM0 bit is cleared, the pull-down resistor connected to the P05/INTP2 pin is controlled. For details, see CHAPTER 4 PORT FUNCTIONS.	p.840 ☐
			PEMU1 register	This bit is not automatically cleared.	p.850 ☐
Chapter 23	Soft	Clock monitor	CLM register	Once the CLME bit has been set to 1, it cannot be cleared to 0 by any means other than reset.	p.852 ☐
Chapter 24	Hard	Regulator	Regulator	Use the regulator with a setting of $V_{DD} = EV_{DD} = AV_{REF0} = AV_{REF1} \geq BV_{DD}$.	p.856 ☐
Chapter 25	Soft	ROM correction function	ROM correction operation and program flow	The software that performs <3> and <4> must be executed in the internal ROM/RAM.	p.860 ☐
			CORADn register	When setting an address to be corrected in the CORADn register, clear the higher bits to 0 in accordance with the capacity of the internal ROM.	p.862 ☐
			ROM correction function	The ROM correction function cannot be used to correct data in the internal ROM. It can only be used to correct instruction codes. If ROM correction is used to correct data, that data is replaced with a DBTRAP instruction code.	p.862 ☐
			ROM code	ROM correction is not performed in regards to the ROM code before writing in the CORCNn register ends.	p.862 ☐

(31/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 25	Soft	ROM correction function	DBTRAPP instruction	After executing a DBTRAP instruction, the PSW.NP, EP, and DI bits are set to 111, and interrupt/exception cannot be acknowledged. After executing a DBTRAP instruction, change the PSW register value as required.	p.862 ☐
			DBPC, DBPSW registers	The DBPC and DBPSW registers can be accessed while DBTRAP instructions are being executed.	p.862 ☐
			Correction address	If the addresses of the instructions executed immediately after the CORCNn register setting (enabled) are set as the correction addresses, normal operation may not be obtained (DBTRAP is not generated).	p.862 ☐
Chapter 26	Hard	Electrical specification	Absolute maximum ratings	Do not directly connect the output (or I/O) pins of IC products to each other, or to V_{DD}, V_{CC}, and GND. Open-drain pins or open-collector pins, however, can be directly connected to each other. Direct connection of the output pins between an IC product and an external circuit is possible, if the output pins can be set to the high-impedance state and the output timing of the external circuit is designed to avoid output conflict.	p.864 ☐
				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. The ratings and conditions indicated for DC characteristics and AC characteristics represent the quality assurance range during normal operation.	p.864 ☐
			Main clock oscillator characteristics	When using the main clock oscillator, wire as follows in the area enclosed by the broken lines in the above figure 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.	p.866 ☐
				When the main clock is stopped and the device is operating on the subclock, wait until the oscillation stabilization time has been secured by the program before switching back to the main clock.	p.866 ☐
				This oscillator constant is a reference value based on evaluation under a specific environment by the resonator manufacturer. If optimization of oscillator characteristics is necessary in the actual application, apply to the resonator manufacturer for evaluation on the implementation circuit. The oscillation voltage and oscillation frequency indicate only oscillator characteristics. Use the V850ES/SG1 so that the internal operating conditions are within the specifications of the DC and AC characteristics.	p.867 ☐
			Subclock oscillator characteristics	When using the subclock 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.	p.869 ☐

(32/32)

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 26	Hard	Electrical specification	Subclock oscillator characteristics	The subclock oscillator is designed as a low-amplitude circuit for reducing power consumption, and is more prone to malfunction due to noise than the main clock oscillator. Particular care is therefore required with the wiring method when the subclock is used.	p.869 ☐
				For the resonator selection and oscillator constant, customers are requested to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.	p.869 ☐
			Data retention characteristics	Shifting to STOP mode and restoring from STOP mode must be performed within the rated operating range.	p.874 ☐
			AC characteristics	If the load capacitance exceeds 50 pF due to the circuit configuration, bring the load capacitance of the device to 50 pF or less by inserting a buffer or by some other means.	p.875 ☐
	Soft		A/D converter	Do not set (read/write) alternate-function ports during A/D conversion; otherwise the conversion resolution may be degraded.	p.896 ☐
Chapter 28	Hard	Recommended soldering condition	Recommended soldering conditions	Do not use different soldering methods together (except for partial heating).	p.900 ☐
Appendix A	Soft	Development tool	Control software	PM+ is included in the C compiler package CA850. It can only be used in Windows.	p.906 ☐
			Embedded software	To purchase the RX850 or RX850 Pro, first fill in the purchase application form and sign the user agreement.	p.915 ☐

<R>

APPENDIX F REVISION HISTORY

F.1 Major Revisions in This Edition

(1/8)

Page	Description
Throughout	• All products are changed to lead-free products. • Addition of μPD703253Y • Change of number of TMP channels
p. 20	Addition of description to **1.1 General**
p. 22	Addition of description to **1.2 Features**
pp. 24, 25	Addition of **Note** to **1.5 Pin Configuration (Top View)**
pp. 30 to 32	Addition of description to **1.6.2 Internal units**
pp. 33 to 35	Addition of description to **2.1 (1) Port pins**
pp. 37 to 41	Addition of description to **2.1 (2) Non-port pins**
p. 42	Modification of description in **Table 2-2 Pin Operation States in Various Modes**
pp. 43, 44	Modification of description in **2.3 Pin I/O Circuit Types, I/O Buffer Power Supplies, and Connection of Unused Pins**
p. 46	Addition of description to **Figure 2-1 Pin I/O Circuits**
p. 47	Modification of description in **2.4 Cautions**
p. 51	Addition of **Note 2** to **Table 3-2 System Register Numbers**
p. 56	Addition of description to **3.2.2 (6) Exception/debug trap status saving registers (DBPC and DBPSW)**
p. 58	Addition of description to **Figure 3-1 Image on Address Space**
p. 60	Addition of description to **Figure 3-2 Data Memory Map (Physical Addresses)**
p. 62	Addition of **3.4.4 (1) (a) Internal ROM (128 KB)**
p. 63	Addition of **3.4.4 (2) (a) Internal RAM (8 KB)**
p. 64	Addition of **Note 4** to **3.4.4 (3) On-chip peripheral I/O area**
p. 65	Addition of **3.4.4 (4) Programmable peripheral I/O area**
p. 65	Modification of description in **3.4.4 (5) External memory area**
p. 66	Addition of description to **3.4.5 (1) Program space**
p. 68	Addition of description to **Figure 3-10 Recommended Memory Map**
pp 69 to 72, 75, 76	Addition of **Note** to **3.4.6 Peripheral I/O registers**
p. 79	Addition of **3.4.7 Programmable peripheral I/O registers**
p. 80	Modification of description in **3.4.8 Special registers**
p. 81	Addition of description to **3.4.8 (1) Setting data to special registers**
p. 83	Modification of description in **3.4.8 (3) (a) Set condition (PRERR flag = 1)**
pp. 85, 86	Modification of description in **3.4.9 (2) Accessing specific on-chip peripheral I/O registers**
p. 87	Modification of description in **3.4.9 (3) System reserved area**
p. 95	Addition of description of **Note** to **Table 4-4 Port 0 Alternate-Function Pins**
p. 97	Addition of **Caution** in **4.3.1 (5) Port 0 function register (PF0)**
p. 98	Modification of description in **Table 4-5 Port 1 Alternate-Function Pins**
p. 98	Addition of **Caution** to **4.3.2 (1) Port 1 register (P1)**
p. 99	Addition of **Caution** to **4.3.2 (2) Port 1 mode register (PM1)**

(2/8)

Page	Description
p. 100	Modification of description in **Table 4-6 Port 3 Alternate-Function Pins**
p. 102	Addition of description to **4.3.3 (1) Port 3 mode control register (PMC3)**
p. 104	Modification of description in **4.3.3 (6) Port 3 alternate function specifications**
p. 105	Addition of **Caution** to **4.3.3 (7) Port 3 function register (PF3)**
p. 106	Modification of description in **Table 4-7 Port 4 Alternate-Function Pins**
p. 108	Addition of **Caution** in **4.3.4 (5) Port 4 function register (PF4)**
p. 109	Modification of description in **Table 4-8 Port 5 Alternate-Function Pins**
p. 112	Addition of **Caution** to **4.3.5 (7) Port 5 function register (PF5)**
p. 114	Modification of description in **4.3.6 (1) Port 7 register H, port 7 register L (P7H, P7L)**
p. 115	Modification of description in **Table 4-10 Port 9 Alternate-Function Pins**
pp. 117, 118	Modification of description in **4.3.7 (3) Port 9 mode control register (PMC9)**
p. 119	Modification of **Caution** to **4.3.7 (4) Port 9 function control register (PFC9)**
p. 120	Modification of description in **4.3.7 (6) Port 9 alternate function specifications**
p. 122	Addition of **Caution** in **4.3.7 (7) Port 9 function register (PF9)**
p. 133	Addition of **Figure 4-4 Block Diagram of Type A-2**
p. 135	Modification of **Figure 4-7 Block Diagram of Type D-2**
p. 137	Modification of **Figure 4-9 Block Diagram of Type E-3**
p. 138	Addition of **Figure 4-10 Block Diagram of Type G-1**
p. 139	Modification of **Figure 4-11 Block Diagram of Type G-2**
p. 140	Modification of **Figure 4-12 Block Diagram of Type G-3**
p. 141	Modification of **Figure 4-13 Block Diagram of Type G-5**
p. 142	Modification of **Figure 4-14 Block Diagram of Type G-6**
p. 143	Modification of **Figure 4-15 Block Diagram of Type G-7**
p. 144	Modification of **Figure 4-16 Block Diagram of Type G-8**
p. 146	Modification of **Figure 4-18 Block Diagram of Type G-10**
p. 147	Modification of **Figure 4-19 Block Diagram of Type G-11**
p. 148	Addition of **Figure 4-20 Block Diagram of Type G-12**
p. 149	Addition of **Figure 4-21 Block Diagram of Type G-14**
p. 152	Modification of **Figure 4-24 Block Diagram of Type N-2**
p. 154	Modification of **Figure 4-26 Block Diagram of Type U-12**
p. 155	Modification of **Figure 4-27 Block Diagram of Type U-13**
p. 156	Modification of **Figure 4-28 Block Diagram of Type U-14**
p. 157	Addition of **Figure 4-29 Block Diagram of Type U-15**
p. 158	Modification of **Figure 4-30 Block Diagram of Type U-16**
p. 159	Modification of **Figure 4-31 Block Diagram of Type U-20**
p. 160	Modification of **Figure 4-32 Block Diagram of Type U-21**
p. 161	Modification of **Figure 4-33 Block Diagram of Type U-22**
p. 162	Modification of **Figure 4-34 Block Diagram of Type U-23**
p. 163	Modification of **Figure 4-35 Block Diagram of Type U-24**
p. 164	Modification of **Figure 4-36 Block Diagram of Type U-25**
p. 165	Modification of **Figure 4-37 Block Diagram of Type U-26**

(3/8)

Page	Description
p. 166	Modification of **Figure 4-38 Block Diagram of Type U-27**
p. 167	Modification of **Figure 4-39 Block Diagram of Type AA-1**
pp. 169 to 173	Modification of description in **Table 4-15 Using Port Pin as Alternate-Function Pin**
p. 179	Addition of description to **4.6.3 Cautions on on-chip debug pins**
p. 180	Modification of description in **4.6.4 Cautions on P10, P11, and P53 pins when power is turned on**
p. 180	Modification of description in **4.6.5 Hysteresis characteristics**
p. 180	Addition of **4.6.6 Cautions on separate bus mode**
p. 181	Deletion of description in **5.1 Features**
p. 183	Addition of description to **Figure 5-1 Data Memory Map: Physical Address**
p. 200	Addition of description to **5.9 Bus Priority**
p. 208	Addition of description to **Figure 6-1 Clock Generator**
p. 209	Addition of description to **6.2 (5) Prescaler 1**
p. 212	Addition of description to **6.3 (1) (a) Example of setting main clock operation → subclock operation**
p. 213	Addition of description to **6.3 (1) (b) Example of setting subclock operation → main clock operation**
p. 214	Addition of **Note** to **6.3 (3) CPU operation clock status register (CCLS)**
p. 216	Addition of **Caution** to **6.5.2 (1) PLL control register (PLLCTL)**
p. 220	Addition of description to **6.5.3 (1) (b)** When transiting to the IDLE2 or STOP mode while remaining in the PLL operation mode
p. 221	Addition of description to **7.1 Overview**
p. 225	Modification of description in **7.4 (1) TMPn control register 0 (TPnCTL0)**
p. 226	Addition of description to **7.4 (2) TMPn control register 1 (TPnCTL1)**
p. 228	Addition of **Note** and **Caution** to **7.4 (3) TMPn I/O control register 0 (TPnIOC0)**
p. 229	Modification of description to **7.4 (4) TMPn I/O control register 1 (TPnIOC1)**
p. 230	Addition of **Note** to **7.4 (5) TMPn I/O control register 2 (TPnIOC2)**
p. 231	Addition of description to **7.4 (6) TMPn option register 0 (TPnOPT0)**
p. 232	Addition of **Note** to **7.4 (7) TMPn capture/compare register 0 (TPnCCR0)**
p. 233	Addition of description to **7.4 (7) (a) Function as compare register**
p. 233	Addition of description to **7.4 (7) (b) Function as capture register**
p. 233	Addition of **Note** and **Remark** to **Table 7-2 Function of Capture/Compare Register in Each Mode and How to Write Compare Register**
p. 234	Addition of **Note** to **7.4 (8) TMPn capture/compare register 1 (TPnCCR1)**
p. 235	Addition of description in **7.4 (8) (a) Function as compare register**
p. 235	Addition of description in **7.4 (8) (b) Function as capture register**
p. 235	Addition of **Note** and **Remark** in **Table 7-3 Function of Capture/Compare Register in Each Mode and How to Write Compare Register**
p. 236	Addition of **Note** to **7.4 (9) TMPn counter read buffer register (TPnCNT)**
p. 237	Modification of description in **Table 7-4 Timer Output Control in Each Mode**
p. 239	Addition of **7.6 (1) Counter basic operation**
p. 240	Addition of **7.6 (2) Anytime write and batch write**
p. 245	Modification of description in **7.6.1 Interval timer mode (TPnMD2 to TPnMD0 bits = 000)**
pp. 246 to 248	Modification of description in **Figure 7-8 Register Setting for Interval Timer Mode Operation**
p. 249	Addition of description to **Figure 7-9 Software Processing Flow in Interval Timer Mode**

(4/8)

Page	Description
p. 250	Modification of description and Figure in **7.6.1 (2) (a) Operation if TPnCCR0 register is set to 0000H**
pp. 254, 255	Addition of description to **7.6.1 (2) (d) Operation of TPnCCR1 register**
p. 256	Addition of **7.6.1 (3) Operation by external event count input (TIPn0)**
p. 257, 259	Modification of description in **7.6.2 External event count mode (TPnMD2 to TPnMD0 bits = 001)**
p. 258	Modification of description in **Figure 7-14 Basic Timing in External Event Count Mode**
p. 260	Modification of description in **Figure 7-15 Register Setting for Operation in External Event Count Mode**
p. 262	Modification of description in **7.6.2 (2) (a) Operation if TPnCCR0 register is set to FFFFH**
p. 263	Modification of description in **7.6.2 (2) (b) Notes on rewriting the TPnCCR0 register**
p. 264	Addition of description in **7.6.2 (2) (c) Operation of TPnCCR1 register**
p. 266	Addition of **Note** and **Caution** in **Figure 7-20 Configuration in External Trigger Pulse Output Mode**
pp. 268, 269	Modification of description in **Figure 7-22 Setting of Registers in External Trigger Pulse Output Mode**
p. 274	Modification of description in **7.6.3 (2) (b) 0%/100% output of PWM waveform**
p. 278	Addition of **Note** and **Caution** to **Figure 7-24 Configuration in One-Shot Pulse Output Mode**
p. 279	Modification of description in **7.6.4 One-shot pulse output mode (TPnMD2 to TPnMD0 bits = 011)**
pp. 280, 281	Modification of description in **Figure 7-26 Setting of Registers in One-Shot Pulse Output Mode**
p. 285	Modification of **Figure 7-28 Configuration in PWM Output Mode**
p. 292	Modification of description in **7.6.5 (2) (b) 0%/100% output of PWM waveform**
p. 294	Addition of **Note** to **Figure 7-32 Configuration in Free-Running Timer Mode**
p. 298	Addition of **Note** to **Figure 7-35 Register Setting in Free-Running Timer Mode**
p. 311	Addition of **7.6.6 (3) Note on capture operation**
p. 312	Modification of **Figure** and addition of **Caution** to **Figure 7-38 Configuration in Pulse Width Measurement Mode**
pp. 314, 315	Modification of description in **Figure 7-40 Register Setting in Pulse Width Measurement Mode**
p. 316	Deletion of description in **Figure 7-41 Software Processing Flow in Pulse Width Measurement Mode**
p. 317	Addition of **7.6.7 (3) Notes**
p. 322	Modification of **Figure 8-3 Basic Timing of Operation in Interval Timer Mode**
p. 325	Modification of **8.4.1 (2) (a) Operation if TM0CMP0 register is set to 0000H**
p. 325	Modification of **8.4.1 (2) (b) Operation if TM0CMP0 register is set to N**
p. 326	Addition of **8.4.2 (3)**
p. 339	Addition of **Caution 4** to **10.3 (1) Watchdog timer mode register 2 (WDTM2)**
p. 341	Addition of **Caution 3** to **10.3 (2) Watchdog timer enable register (WDTE)**
p. 348	Modification of description in **11.4 (1) A/D converter mode register 0 (ADA0M0)**
pp. 349 to 351	Addition of **Caution** and deletion of **11.4 (2) A/D converter mode register 1 (ADA0M1)**
p. 350	Modification of description in **Table 11-2 Conversion Time Selection in Normal Conversion Mode (ADA0HS1 Bit = 0)**
p. 351	Modification of description in **Table 11-3 Conversion Time Selection in High-Speed Conversion Mode (ADA0HS1 Bit = 1)**
p. 352	Addition of **Caution** to **11.4 (3) A/D converter mode register 2 (ADA0M2)**
p. 353	Addition of **Caution** to **11.4 (4) Analog input channel specification register (ADA0S)**
p. 356	Addition of **Caution 3** to **11.4 (6) Power-fail compare mode register (ADA0PFM)**
p. 357	Addition of **Caution** to **11.4 (7) Power-fail compare threshold value register (ADA0PFT)**
p. 358	Addition of **Remark** to **11.5.1 Basic operation**

(5/8)

Page	Description
p. 359	Modification of description in **Figure 11-3 Conversion Operation Timing (Continuous Conversion)**
p. 360	Addition of description to **11.5.3 (1) Software trigger mode**
p. 360	Addition of description to **11.5.3 (2) External trigger mode**
p. 361	Addition of description to **11.5.3 (3) Timer trigger mode**
p. 372	Modification of description in **11.6 (4) Alternate I/O**
p. 373	Modification of description in **Figure 11-14 Internal Equivalent Circuit of ANIn Pin**
p. 374	Modification of description in **11.6 (8) Reading ADA0CRn register**
p. 375	Modification of description in **11.6 (10) Restriction for each mode**
p. 380	Addition of **CHAPTER 12 D/A CONVERTER**
p. 386	Modification of **Figure 13-1 Block Diagram of Asynchronous Serial Interface An**
p. 389	Modification of description in **13.3 (1) UARTAn control register 0 (UAnCTL0)**
p. 390	Addition of description to **13.3 (4) UARTAn option control register 0 (UAnOPT0)**
p. 391	Addition of **Caution** to **13.3 (5) UARTAn status register (UAnSTR)**
p. 393	Addition of description to **13.3 (6) UARTAn receive data register (UAnRX)**
p. 393	Addition of description to **13.3 (7) UARTAn transmit data register (UAnTX)**
p. 396	Modification of description in **Figure 13-2 UARTA Transmit/Receive Data Format**
p. 399	Modification of **Figure 13-5 Continuous Transmission Operation Timing**
p. 405	Addition of **Caution** to **Figure 13-9 Configuration of Baud Rate Generator**
p. 405	Addition of description to **13.6 (1) (a) Baud rate generator configuration**
p. 413	Addition of description to **13.7 Cautions**
pp. 418, 420	Addition of **Caution, Note**, and description of **Note** to **14.4 (1) CSIBn control register 0 (CBnCTL0)**
p. 421	Addition of **14.4 (1) (a) How to use CBnSCE bit**
p. 422	Addition of **Note** to **14.4 (2) CSIBn control register 1 (CBnCTL1)**
p. 425	Addition of **Caution** to **14.4 (4) CSIBn status register (CBnSTR)**
p. 426	Addition of **14.5 Interrupt Request Signals**
p. 427	Addition of **14.6.1 Single transfer mode (master mode, transmission mode)**
p. 429	Modification of description in **14.6.2 Single transfer mode (master mode, reception mode)**
p. 431	Modification of description in **14.6.3 Single transfer mode (master mode, transmission/reception mode)**
p. 433	Addition of **14.6.4 Single transfer mode (slave mode, transmission mode)**
p. 435	Addition of **14.6.5 Single transfer mode (slave mode, reception mode)**
p. 437	Addition of **14.6.6 Single transfer mode (slave mode, transmission/reception mode)**
p. 439	Addition of **14.6.7 Continuous transfer mode (master mode, transmission mode)**
p. 441	Modification of description in **14.6.8 Continuous transfer mode (master mode, reception mode)**
p. 444	Modification of description in **14.6.9 Continuous transfer mode (master mode, transmission/reception mode)**
p. 448	Addition of **14.6.10 Continuous transfer mode (slave mode, transmission mode)**
p. 450	Modification of description in **14.6.11 Continuous transfer mode (slave mode, reception mode)**
p. 453	Modification of description in **14.6.12 Continuous transfer mode (slave mode, transmission/reception mode)**
p. 457	Modification of description in **14.6.13 Reception error**
pp. 458, 459	Modification of description in **14.6.14 Clock timing**
p. 462	Addition of **Caution** to **14.8.1 Baud rate generation**

(6/8)

Page	Description
p. 463	Addition of description to **14.9 Cautions**
p. 470	Deletion of description in **15.4 (1) IIC control registers 0, 1 (IICC0, IICC1)**
p. 491	Modification of description in **15.6.6 Wait state**
p. 493	Addition of **Note** to **15.6.7 Wait state cancellation method**
p. 495	Addition of **Note** to **15.7.1 (1) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)**
p. 496	Addition of **Note** and deletion of description in **15.7.2 (2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)**
p. 497	Addition of **Note** to **15.7.1 (3) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)**
p. 501	Modification of description in **15.7.2 (4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop**
p. 502	Addition of description to **15.7.3 Slave device operation (when receiving extension code)**
p. 505	Modification of description in **15.7.3 (4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop**
p. 507	Addition of description to **15.7.5 Arbitration loss operation (operation as slave after arbitration loss)**
p. 509	Addition of description to **15.7.6 Operation when arbitration loss occurs (no communication after arbitration loss)**
p. 513	Modification of description in **15.7.6 (6) When arbitration loss occurs due to low level of SDA0n pin when attempting to generate a restart condition**
p. 515	Modification of description in **15.7.6 (8) When arbitration loss occurs due to low level of SDA0n pin when attempting to generate a stop condition**
p. 518	Addition of description to **15.11 Extension Code**
p. 521	Modification of description in **15.14.1 When communication reservation function is enabled (IICFn.IICRSVn bit = 0)**
p. 523	Modification of **Figure 15-13 Communication Reservation Timing**
p. 525	Modification of description in **Table 15-7 Wait Periods**
pp. 527 to 531	Modification of description in **15.16 Communication Operations**
p. 534	Modification of **Figure 15-19 Slave Operation Flowchart (1)**
p. 535	Modification of **Figure 15-20 Slave Operation Flowchart (2)**
pp. 537 to 539	Modification of **Figure 15-21 Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave)**
pp. 540 to 542	Modification of **Figure 15-22 Example of Slave to Master Communication (When 9-Clock Wait for Master and 8-Clock Wait for Slave Are Selected)**
p. 576	Deletion of description in **16.3 (6) (e) Overrun error occurrence flag (OERR)**
p. 578	Modification of description of **Caution** in **16.3 (8) IEBus slave address register (SAR)**
p. 586	Deletion of description in **16.3 (13) (b) When reception unit**
p. 597	Deletion of description in **16.4.3 Interrupt source list**
p. 597	Modification of description in **Table 16-17 Interrupt Source List**
p. 600	Deletion of description in **Figure 16-26 Master Transmission**
p. 600	Deletion of description in **16.5.1 Master transmission**
p. 602	Deletion of description in **Figure 16-27 Master Reception**
p. 604	Deletion of description in **Figure 16-28 Slave Transmission**
p. 606	Deletion of description in **Figure 16-29 Slave Reception**
p. 612	Addition of **CHAPTER 17 CAN CONTROLLER**
p. 755	Addition of **CHAPTER 18 DMA FUNCTION (DMA CONTROLLER)**
p. 779	Modification of description in **CHAPTER 19 INTERRUPT/EXCEPTION PROCESSING FUNCTION**

(7/8)

Page	Description
p. 779	Addition of description to **19.1 Features**
pp. 780, 781	Addition of description to **Table 19-1 Interrupt Source List**
p. 788	Modification of description in **19.3 Maskable Interrupts**
p. 795	Addition of **Caution 2** to **19.3.4 Interrupt control register (xxICn)**
pp. 796, 797	Addition of description to **Table 19-2 Interrupt Control Register (xxICn)**
p. 798	Modification of description in **19.3.5 Interrupt mask registers 0 to 3 (IMR0 to IMR3)**
p. 805	Addition of **Caution** to **19.5.1 (2) Restoration**
p. 812	Modification of description in **19.6.2 (4) Noise elimination control register (NFC)**
p. 813	Addition of description to **19.7 Interrupt Acknowledge Time of CPU**
p. 814	Addition of description to **Figure 19-15 Pipeline Operation at Interrupt Request Signal Acknowledgment (Outline)**
p. 815	Addition of **19.9 (2) Interrupt control register (xxICn)**
p. 815	Addition of **19.9 (3) In-service priority register (ISPR)**
p. 820	Addition of **Caution** to **21.2 (1) Power save control register (PSC)**
p. 824	Addition of description to **Table 21-3 Operating Status in HALT Mode**
p. 826	Addition of description to **Table 21-5 Operating Status in IDLE1 Mode**
p. 828	Addition of description to **Table 21-7 Operating Status in IDLE2 Mode**
p. 832	Addition of description to **Table 21-9 Operating Status in STOP Mode**
p. 835	Modification of description in **Table 21-10 Operating Status in Subclock Operation Mode**
p. 837	Addition of description to **Table 21-12 Operating Status in Sub-IDLE Mode**
p. 838	Addition of description to **22.2 (2) Emergency operation mode**
p. 840	Modification of description in **Table 22-1 Hardware Status on $\overline{\text{RESET}}$ Pin Input**
p. 844	Addition of **22.3.3 Reset operation by clock monitor (CLMRES)**
p. 848	Addition of description to **22.4 Valid/Invalid of Internal RAM Data**
p. 853	Deletion of description in **Table 23-2 Operation Status of Clock Monitor (When CLM.CLME Bit = 1, During Internal Oscillation Clock Operation)**
p. 856	Addition of description to **24.1 Outline**
p. 856	Modification of description in **Figure 24-1 Regulator**
p. 859	Addition of description to **25.2 (1) Correction address registers 0 to 3 (CORAD0 to CORAD3)**
p. 865	Modification of description in **26.2 Capacitance**
p. 865	Addition of description to **26.3 Operating Conditions**
p. 867	Addition of **26.4.1 (i) KYOCERA KINSEKI CORPORATION: Crystal resonator**
p. 868	Addition of **26.4.1 (ii) Murata Mfg. Co. Ltd.: Ceramic resonator**
p. 873	Addition of **26.6.2 (2) μPD703253Y**
p. 877	Modification of description in **26.8.2 (1) (a) Read/write cycle (CLKOUT asynchronous)**
p. 882	Modification of description in **26.8.2 (2) (a) Read cycle (CLKOUT asynchronous): In separate bus mode**
p. 891	Addition of **Note** to **26.9 (4) Timer timing**
p. 895	Addition of **26.9 (9) CAN timing (μPD703253Y only)**
p. 896	Addition of **26.9 (11) D/A converter (μPD703253Y only)**
p. 897	Addition of **26.9 (12) RAM retention detection (μPD703253Y only)**
p. 900	Modification of description in **CHAPTER 28 RECOMMENDED SOLDERING CONDITIONS**
p. 902	Addition of description of **Note** to **Figure A-1 (1) When using in-circuit emulator IE-V850ES-G1**

(8/8)

Page	Description
p. 903	Addition of description of **Note** to **Figure A-1 (2) When using IECUBE QB-V850ESSX2**
p. 904	Addition of **Figure A-1 (3) When using on-chip debug emulator IE-V850E1-CD-NW**
p. 905	Addition of **Figure A-1 (4) When using MINICUBE QB-V850MINI**
p. 906	Modification of description in **A.2 Language Processing Software**
p. 909	Modification of description in **Figure A-3 System Configuration (QB-V850ESSX2 Used)**
p. 909	Modification of **Note** in **A.4.2 When using IECUBE QB-V850ESSX2**
p. 912	Addition of **A.4.3 When using on-chip debug emulator IE-V850E1-CD-NW**
p. 913	Addition of **A.4.4 When using MINICUBE QB-V850MINI**
p. 914	Modification of description in **A.5 Debugging Tools (Software)**
p. 916	Modification of description in **A.7 Flash Memory Writing Tools**
p. 917	Modification of description in **APPENDIX B MAJOR DIFFERENCES BETWEEN V850ES/SG1 (μPD703249Y, 703252Y) AND V850ES/SG1 (μPD703253Y) AND V850ES/SG2 (μPD70F3261Y, 70F3271Y, 70F3281Y)**
p. 939	Addition of **APPENDIX E LIST OF CAUTIONS**
p. 972	Addition of **APPENDIX F REVISION HISTORY**

For further information,
please contact:

NEC Electronics Corporation
1753, Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa 211-8668,
Japan
Tel: 044-435-5111
http://www.necel.com/

[America]

NEC Electronics America, Inc.
2880 Scott Blvd.
Santa Clara, CA 95050-2554, U.S.A.
Tel: 408-588-6000
800-366-9782
http://www.am.necel.com/

[Europe]

NEC Electronics (Europe) GmbH
Arcadiastrasse 10
40472 Düsseldorf, Germany
Tel: 0211-65030
http://www.eu.necel.com/

Hanover Office
Podbielskistrasse 166 B
30177 Hannover
Tel: 0 511 33 40 2-0

Munich Office
Werner-Eckert-Strasse 9
81829 München
Tel: 0 89 92 10 03-0

Stuttgart Office
Industriestrasse 3
70565 Stuttgart
Tel: 0 711 99 01 0-0

United Kingdom Branch
Cygnus House, Sunrise Parkway
Linford Wood, Milton Keynes
MK14 6NP, U.K.
Tel: 01908-691-133

Succursale Française
9, rue Paul Dautier, B.P. 52
78142 Velizy-Villacoublay Cédex
France
Tel: 01-3067-5800

Sucursal en España
Juan Esplandiu, 15
28007 Madrid, Spain
Tel: 091-504-2787

Tyskland Filial
Täby Centrum
Entrance S (7th floor)
18322 Täby, Sweden
Tel: 08 638 72 00

Filiale Italiana
Via Fabio Filzi, 25/A
20124 Milano, Italy
Tel: 02-667541

Branch The Netherlands
Steijgerweg 6
5616 HS Eindhoven
The Netherlands
Tel: 040 265 40 10

[Asia & Oceania]

NEC Electronics (China) Co., Ltd
7th Floor, Quantum Plaza, No. 27 ZhiChunLu Haidian
District, Beijing 100083, P.R.China
Tel: 010-8235-1155
http://www.cn.necel.com/

Shanghai Branch
Room 2509-2510, Bank of China Tower,
200 Yincheng Road Central,
Pudong New Area, Shanghai, P.R.China P.C:200120
Tel:021-5888-5400
http://www.cn.necel.com/

Shenzhen Branch
Unit 01, 39/F, Excellence Times Square Building,
No. 4068 Yi Tian Road, Futian District, Shenzhen,
P.R.China P.C:518048
Tel:0755-8282-9800
http://www.cn.necel.com/

NEC Electronics Hong Kong Ltd.
Unit 1601-1613, 16/F., Tower 2, Grand Century Place,
193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: 2886-9318
http://www.hk.necel.com/

NEC Electronics Taiwan Ltd.
7F, No. 363 Fu Shing North Road
Taipei, Taiwan, R. O. C.
Tel: 02-8175-9600
http://www.tw.necel.com/

NEC Electronics Singapore Pte. Ltd.
238A Thomson Road,
#12-08 Novena Square,
Singapore 307684
Tel: 6253-8311
http://www.sg.necel.com/

NEC Electronics Korea Ltd.
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku,
Seoul, 135-080, Korea
Tel: 02-558-3737
http://www.kr.necel.com/

G0706