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

Preliminary User's Manual

μPD780701Y Subseries

8-Bit Single-Chip Microcontrollers

μPD780701Y
μPD780702Y
μPD78F0701Y

Document No. U13781EJ2V0UM00 (2nd edition)
Date Published October 2004 N CP(K)

© NEC Electronics Corporation 1999
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.

FIP, IEBus, and Inter Equipment Bus are trademarks of NEC Electronics Corporation.
Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
PC/AT is a trademark of International Business Machines Corporation.
HP9000 Series 700 and HP-UX are trademarks of Hewlett-Packard Company.
SPARCstation is a trademark of SPARC International, Inc.
SunOS and Solaris are trademarks of Sun Microsystems, Inc.
Ethernet is a trademark of Xerox Corporation.
NEWS and NEWS-OS are trademarks of Sony Corporation.
OSF/Motif is a trademark of OpenSoftware Foundation, Inc.
TRON is an abbreviation of The Realtime Operating System Nucleus.
ITRON is an abbreviation of Industrial TRON.

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 contained in this document is being issued in advance of the production cycle for the product. The parameters for the product may change before final production or NEC Electronics Corporation, at its own discretion, may withdraw the product prior to its production.**
- **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 products 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).

M5D 02. 11-1

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC Electronics product in your application, please contact the NEC Electronics office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

[GLOBAL SUPPORT]
http://www.necel.com/en/support/support.html

NEC Electronics America, Inc. (U.S.)
Santa Clara, California
Tel: 408-588-6000
800-366-9782

NEC Electronics (Europe) GmbH
Duesseldorf, Germany
Tel: 0211-65030

- **Sucursal en España**
  Madrid, Spain
  Tel: 091-504 27 87
- **Succursale Française**
  Vélizy-Villacoublay, France
  Tel: 01-30-67 58 00
- **Filiale Italiana**
  Milano, Italy
  Tel: 02-66 75 41
- **Branch The Netherlands**
  Eindhoven, The Netherlands
  Tel: 040-244 58 45
- **Tyskland Filial**
  Taeby, Sweden
  Tel: 08-63 80 820
- **United Kingdom Branch**
  Milton Keynes, UK
  Tel: 01908-691-133

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-558-3737

NEC Electronics Shanghai Ltd.
Shanghai, P.R. China
Tel: 021-5888-5400

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-2719-2377

NEC Electronics Singapore Pte. Ltd.
Novena Square, Singapore
Tel: 6253-8311

J04.1

INTRODUCTION

Readers

This manual has been prepared for users who understand the functions of the μPD780701Y Subseries and wish to design and develop application systems and programs for these devices.

μPD780701Y Subseries: μPD780701Y, 780702Y, 78F0701Y

Purpose

This manual is intended for users to understand the functions described in the organization below.

Organization

The μPD780701Y Subseries manual is divided into two parts: this manual and the instruction manual (common to the 78K/0 Series).

μPD780701Y Subseries User's Manual (This Manual)	**78K/0 Series User's Manual Instructions**
• Pin functions • Internal block functions • Interrupt • Other on-chip peripheral functions	• CPU functions • Instruction set • Explanation of each instruction

How To Read This Manual

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

- To gain a general understanding of functions:
  → Read this manual in the order of the contents. The mark ★ shows major revised points.
- How to interpret the register format:
  → For a bit number enclosed in a square, the bit name is defined as a reserved word in RA78K/0, and in CC78K/0, already defined in the header file named sfrbit.h.
- To check the details of a register when you know the register name:
  → Refer to **APPENDIX C REGISTER INDEX.**

Conventions

Data significance:	Higher digits on the left and lower digits on the right
Active low representation:	$\overline{\times\times\times}$ (overscore over pin or signal name)
Note:	Footnote for item marked with **Note** in the text
Caution:	Information requiring particular attention
Remark:	Supplementary information
Numerical representation:	Binary ··· ×××× or ××××B
	Decimal ··· ××××
	Hexadecimal ··· ××××H

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

- **Device-related documents**

Document Name	Document No.
μPD780701Y, 780702Y Preliminary Product Information	U13920E
μPD78F0701Y Preliminary Product Information	U13563E
μPD780701Y Subseries User's Manual	This manual
78K/0 Series User's Manual Instructions	U12326E

- **Related documents for development tools (User's Manuals)**

Document Name		Document No.
RA78K0 Assembler Package	Operation	U11802E
	Language	U11801E
	Structured Assembly Language	U11789E
CC78K0 C Compiler	Operation	U11517E
	Language	U11518E
IE-78K0-R-EX1 In-circuit Emulator		To be prepared
IE-780701-NS-EM1 Emulation Board		To be prepared
SM78K0S, SM78K0 System Simulator Windows™ Based	Operation	U14611E
SM78K Series System Simulator	External Part User Open Interface Specifications	U10092E
ID78K0-NS Integrated Debugger Ver. 2.00 or later Windows Based	Operation	U14379E
ID78K0-NS, ID78K0S-NS Integrated Debugger Ver 2.20 or later Windows Based	Operation	U14910E
ID78K0 Integrated Debugger Windows Based	Reference	U11539E
	Guide	U11649E

- **Related documents for embedded software (User's Manuals)**

Document Name		Document No.
78K/0 Series Real-Time OS	Fundamental	U11537E
	Installation	U11536E
78K/0 Series OS MX78K0	Fundamental	U12257E

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

- **Other documents**

Document Name	Document No.
SEMICONDUCTOR SELECTION GUIDE - Products and Packages -	X13769X
Semiconductor Device Mount Manual	**Note**
Quality Grades on NEC Semiconductor Devices	C11531E
NEC Semiconductor Device Reliability/Quality Control System	C10983E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E

Note See the “Semiconductor Device Mount Manual” website (http://www.necel.com/pkg/en/mount/index.html).

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

CONTENTS

CHAPTER 1 OUTLINE 17
1.1 Features 17
1.2 Applications 18
1.3 Ordering Information 18
1.4 Pin Configuration (Top View) 19
1.5 78K/0 Series Lineup 23
1.6 Block Diagram 24
1.7 Overview of Functions 27

CHAPTER 2 PIN FUNCTIONS 28
2.1 Pin Function List 28
2.2 Description of Pin Functions 32
2.2.1 P00 to P07 (Port 0) 32
2.2.2 P20 to P27 (Port 2) 32
2.2.3 P30 to P36 (Port 3) 33
2.2.4 P40 to P47 (Port 4) 33
2.2.5 P50 to P57 (Port 5) 33
2.2.6 P64 to P67 (Port 6) 33
2.2.7 P70 to P77 (Port 7) 34
2.2.8 P80 to P87 (Port 8) 34
2.2.9 P90 to P97 (Port 9) 35
2.2.10 CTXD (μPD780701Y and 78F0701Y only) 35
2.2.11 CRXD (μPD780701Y and 78F0701Y only) 35
2.2.12 $\overline{\text{ITX0}}$ (μPD780702Y and 78F0701Y only) 35
2.2.13 $\overline{\text{IRX0}}$ (μPD780702Y and 78F0701Y only) 35
2.2.14 AV_{REF} 35
2.2.15 AV_{SS} 35
2.2.16 $\overline{\text{RESET}}$ 35
2.2.17 X1 and X2 35
2.2.18 CPUREG 35
2.2.19 V_{DD0} and V_{DD1} 35
2.2.20 V_{SS0} and V_{SS1} 35
2.2.21 V_{PP} (μPD78F0701Y only) 36
2.2.22 IC (μPD780701Y and 780702Y) 36
2.3 Pin Input/Output Circuits and Recommended Connection of Unused Pins 37

CHAPTER 3 CPU ARCHITECTURE 40
3.1 Memory Space 40
3.1.1 Internal program memory space 43
3.1.2 Internal data memory space 45
3.1.3 Special function register (SFR) area 45
3.1.4 External memory space 45
3.1.5 Data memory addressing 46
3.2 Processor Registers 49
3.2.1 Control registers 49
3.2.2 General registers 52
3.2.3 Special function register (SFR) 54
3.3 Instruction Address Addressing 59
3.3.1 Relative addressing 59
3.3.2 Immediate addressing 60
3.3.3 Table indirect addressing 61

3.3.4 Register addressing 62
3.4 Operand Address Addressing 63
3.4.1 Implied addressing 63
3.4.2 Register addressing 64
3.4.3 Direct addressing 65
3.4.4 Short direct addressing 66
3.4.5 Special function register (SFR) addressing 67
3.4.6 Register indirect addressing 68
3.4.7 Based addressing 69
3.4.8 Based indexed addressing 70
3.4.9 Stack addressing 70

CHAPTER 4 PORT FUNCTIONS 71
4.1 Port Functions 71
4.2 Port Configuration 74
4.2.1 Port 0 74
4.2.2 Port 2 76
4.2.3 Port 3 77
4.2.4 Port 4 79
4.2.5 Port 5 80
4.2.6 Port 6 81
4.2.7 Port 7 82
4.2.8 Port 8 84
4.2.9 Port 9 85
4.3 Port Function Control Registers 86
4.4 Port Function Operations 89
4.4.1 Writing to input/output port 89
4.4.2 Reading from input/output port 89
4.4.3 Operations on input/output port 89

CHAPTER 5 CLOCK GENERATOR 90
5.1 Clock Generator Functions 90
5.2 Clock Generator Configuration 90
5.3 Clock Generator Control Register 91
5.4 System Clock Oscillator 92
5.4.1 System clock oscillator 92
5.4.2 Frequency divider 94
5.5 Clock Generator Operations 95
5.6 Changing CPU Clock Setting 95
5.6.1 Time required for switchover between CPU clocks 95
5.6.2 CPU clock switching procedure 96

CHAPTER 6 16-BIT TIMER/EVENT COUNTER 97
6.1 Outline of Timer Integrated in μPD780701Y Subseries 97
6.2 16-Bit Timer/Event Counter Functions 98
6.3 16-Bit Timer/Event Counter Configuration 99
6.4 16-Bit Timer/Event Counter Control Registers 102
6.5 16-Bit Timer/Event Counter Operations 112
6.5.1 Interval timer operation 112
6.5.2 PPG output operation 114
6.5.3 Pulse width measurement operation 115
6.5.4 External event counter operation 122
6.5.5 Square-wave output operation 123

6.5.6 One-shot pulse output operation 125
6.6 Cautions for 16-Bit Timer/Event Counter 130

CHAPTER 7 8-BIT TIMER/EVENT COUNTER 134
7.1 Function of 8-Bit Timer/Event Counters 134
7.2 Configuration of 8-Bit Timer/Event Counter 135
7.3 Registers Controlling 8-Bit Timer/Event Counters 139
7.4 Operation of 8-Bit Timer/Event Counter 146
7.4.1 Interval timer (8-bit) operation 146
7.4.2 External event counter operation 149
7.4.3 Square-wave output (8-bit resolution) operation 150
7.4.4 8-bit PWM output operation 151
7.4.5 Interval timer (16-bit) operation 154
7.5 Cautions for 8-Bit Timer/Event Counter 155

CHAPTER 8 WATCH TIMER 156
8.1 Watch Timer Functions 156
8.2 Watch Timer Configuration 157
8.3 Watch Timer Control Registers 158
8.4 Watch Timer Operations 159
8.4.1 Watch timer operation 159
8.4.2 Interval timer operation 159

CHAPTER 9 WATCHDOG TIMER 161
9.1 Watchdog Timer Functions 161
9.2 Watchdog Timer Configuration 162
9.3 Watchdog Timer Control Registers 163
9.4 Watchdog Timer Operations 166
9.4.1 Operation as watchdog timer 166
9.4.2 Operation as interval timer 167

CHAPTER 10 CLOCK OUTPUT/BUZZER OUTPUT CONTROL CIRCUITS 168
10.1 Clock Output/Buzzer Output Control Circuit Functions 168
10.2 Clock Output/Buzzer Output Control Circuit Configuration 169
10.3 Clock Output/Buzzer Output Control Circuit Control Registers 169
10.4 Clock Output/Buzzer Output Control Circuit Operations 172
10.4.1 Operation as clock output 172
10.4.2 Operation as buzzer output 172

CHAPTER 11 A/D CONVERTER 173
11.1 Function of A/D Converter 173
11.2 A/D Converter Configuration 175
11.3 A/D Converter Control Registers 177
11.4 A/D Converter Operations 180
11.4.1 Basic operations of A/D converter 180
11.4.2 Input voltage and conversion results 182
11.4.3 A/D converter operation mode 183
11.5 Cautions for A/D Converter 185

CHAPTER 12 SERIAL INTERFACE (UART0) 188
12.1 Serial Interface Functions 188
12.2 Serial Interface Configuration 189
12.3 Serial Interface Control Registers 190
12.4 Serial Interface Operations 194

12.4.1 Operation stop mode 194
12.4.2 Asynchronous serial interface (UART) mode 194

CHAPTER 13 SERIAL INTERFACE (SIO30, SIO31) 205
13.1 Serial Interface Functions 206
13.2 Serial Interface Configuration 207
13.3 Serial Interface Control Registers 208
13.4 Serial Interface Operations 209
13.4.1 Operation stop mode 209
13.4.2 3-wire serial I/O mode 210

CHAPTER 14 SERIAL INTERFACE (IIC0) 212
14.1 Serial Interface Functions 212
14.2 Serial Interface Configuration 215
14.3 Serial Interface Control Registers 216
14.4 I^2C Bus Mode Functions 225
14.4.1 Pin configuration 225
14.5 I^2C Bus Definitions and Control Methods 226
14.5.1 Start conditions 226
14.5.2 Addresses 227
14.5.3 Transfer direction specification 227
14.5.4 Acknowledge ($\overline{ACK}$) signal 228
14.5.5 Stop condition 229
14.5.6 Wait signal ($\overline{WAIT}$) 230
14.5.7 I^2C interrupt requests (INTIIC0) 232
14.5.8 Interrupt request (INTIIC0) generation timing and wait control 251
14.5.9 Address match detection method 252
14.5.10 Error detection 252
14.5.11 Extension code 252
14.5.12 Arbitration 253
14.5.13 Wake-up function 254
14.5.14 Communication reservation 255
14.5.15 Other cautions 257
14.5.16 Communication operations 258
14.6 Timing Charts 260

CHAPTER 15 DCAN CONTROLLER (μPD780701Y, 78F0701Y ONLY) 267
15.1 Protocol 268
15.1.1 Protocol mode function 268
15.1.2 Message format 268
15.1.3 Data frame/remote frame 269
15.1.4 Error frame 275
15.1.5 Overload frame 276
15.2 Functions 277
15.2.1 Bus priority decision 277
15.2.2 Bit stuffing 277
15.2.3 Multi master 277
15.2.4 Multi cast 277
15.2.5 Sleep mode/stop mode function 278
15.2.6 Error control function 278
15.2.7 Baud rate control function 281
15.2.8 State shift chart 284
15.3 Outline 287

15.4 Connection with Target System 288
15.5 DCAN Controller Configuration 288
15.6 Special Function Registers (SFR) for DCAN Controller 289
15.7 Message and Buffer Configuration 290
15.8 Transmit Buffer Configuration 291
15.9 Transmit Message 291
15.9.1 Transmit message specification section 292
15.9.2 Transmit identifier section 293
15.9.3 Transmit data section 294
15.10 Receive Message Buffer Configuration 295
15.11 Receive Message 296
15.11.1 Receive control specification section 297
15.11.2 Receive status section 298
15.11.3 Receive identifier specification section 300
15.11.4 Receive message data section 301
15.12 Mask Function 302
15.12.1 Identifier mask 302
15.12.2 Mask identifier control bit setting 304
15.12.3 Mask identifier setting 304
15.13 DCAN Controller Control Registers 305
15.13.1 CAN control register (CANC) 305
15.13.2 CAN error status register (CANES) 309
15.13.3 Transmit error counter (TEC) 311
15.13.4 Receive error counter (REC) 311
15.13.5 Message count register (MCNT) 312
15.13.6 Bit rate prescaler (BRPRS) 313
15.13.7 Synchronization control register n (SYNCn: n = 0, 1) 314
15.13.8 Transmit control register (TCR) 318
15.13.9 Receive message register (RMES) 320
15.13.10 Mask control register (MASKC) 321
15.13.11 Redefinition control register (REDEF) 323
15.14 Interrupt Function 325
15.14.1 Interrupt vectors 325
15.14.2 Transmit interrupt 325
15.14.3 Receive interrupt 325
15.14.4 Error interrupt 325
15.15 Standby Function 326
15.15.1 HALT mode 326
15.15.2 STOP mode 326
15.15.3 DCAN sleep mode 326
15.15.4 DCAN stop mode 326
15.16 Description of Functions by Flowcharts 327
15.16.1 Initialization procedure 327
15.16.2 Transmit procedure 328
15.16.3 Transmit abort procedure 329
15.16.4 Receive processing by DCAN controller 330
15.16.5 Receive procedure (when receive interrupt request signal (INTCR) used) 331
15.16.6 Receive procedure (when receive interrupt request signal (INTCR) not used) 332

CHAPTER 16 IEBus CONTROLLER (μPD780702Y, 78F0701Y ONLY) 333
16.1 IEBus Controller Functions 333
16.1.1 Communication protocol of IEBus 333

16.1.2 Determination of bus mastership (arbitration) 334
16.1.3 Communication mode 334
16.1.4 Communication address 334
16.1.5 Broadcast communication 335
16.1.6 Transfer format of IEBus 335
16.1.7 Transfer data 345
16.1.8 Bit format 348
16.2 Simple IEBus Controller 349
16.3 Configuration of IEBus Controller 350
16.4 Internal Registers of IEBus Controller 352
16.4.1 Internal register list 352
16.4.2 Description of internal registers 353
16.5 Interrupt Operations of IEBus Controller 372
16.5.1 Interrupt control block 372
16.5.2 Interrupt source list 373
16.5.3 Communication error source processing list 374
16.6 Interrupt Generation Timing and Main CPU Processing 377
16.6.1 Master transmission 377
16.6.2 Master reception 379
16.6.3 Slave transmission 381
16.6.4 Slave reception 383
16.6.5 Interval of occurrence of interrupt for IEBus control 385

CHAPTER 17 INTERRUPT FUNCTIONS 389
17.1 Interrupt Function Types 389
17.2 Interrupt Sources and Configuration 389
17.3 Interrupt Function Control Registers 394
17.4 Interrupt Servicing Operations 401
17.4.1 Non-maskable interrupt request acknowledge operation 401
17.4.2 Maskable interrupt acknowledge operation 404
17.4.3 Software interrupt request acknowledge operation 406
17.4.4 Multiple interrupt servicing 407
17.4.5 Interrupt request hold 410

CHAPTER 18 STANDBY FUNCTION 411
18.1 Standby Function and Configuration 411
18.1.1 Standby function 411
18.1.2 Standby function control register 412
18.2 Standby Function Operations 413
18.2.1 HALT mode 413
18.2.2 STOP mode 416

CHAPTER 19 RESET FUNCTION 419

CHAPTER 20 μPD78F0701Y 424
20.1 Internal Bus Controller (DCAN/IEBus) Switching 425
20.2 Internal Memory Size Switching Register (IMS) 426
20.3 Internal Expansion RAM Size Switching Register (IXS) 426
20.4 Flash Memory Programming 427
20.4.1 Selection of communication mode 427
20.4.2 Flash memory programming function 428
20.4.3 Flashpro II and Flashpro III connections 428
20.5 Flash Memory Programming by Self Write 429

20.5.1 Flash memory configuration 429
20.5.2 Flash programming mode control register 430
20.5.3 Self-write procedure 430
20.5.4 CPU resources 433
20.5.5 Entry RAM area 433
20.5.6 Self-write subroutines 435
20.5.7 Self-write circuit configuration 445

CHAPTER 21 INSTRUCTION SET 446
21.1 Conventions 447
21.1.1 Operand identifiers and description methods 447
21.1.2 Description of “operation” column 448
21.1.3 Description of “flag operation” column 448
21.2 Operation List 449
21.3 Instructions Listed by Addressing Type 457

APPENDIX A DEVELOPMENT TOOLS 461
A.1 Language Processing Software 464
A.2 Flash Memory Writing Tools 465
A.3 Debugging Tools 466
A.3.1 Hardware 466
A.3.2 Software 468
A.4 Upgrading from Former In-circuit Emulator for 78K/0 Series to IE-78001-R-A 470

APPENDIX B EMBEDDED SOFTWARE 473

APPENDIX C REGISTER INDEX 475
C.1 Register Index (In Alphabetical Order with Respect to Register Names) 475
C.2 Register Index (In Alphabetical Order with Respect to Register Symbol) 479

★ **APPENDIX D REVISION HISTORY 483**
D.1 Major Revisions in This Edition 483

CHAPTER 1 OUTLINE

1.1 Features

- Internal high-capacity ROM and RAM

Item / Part Number	Program Memory	Data Memory		
		Internal High-speed RAM	Internal Expansion RAM	DCAN Buffer RAM
μPD780701Y	60 Kbytes (Mask ROM)	1,024 bytes	2,048 bytes	288 bytes
μPD780702Y				—
μPD78F0701Y	60 Kbytes (Flash memory)			288 bytes

- External memory expansion space: 64 Kbytes
- Minimum instruction execution time can be changed from high-speed (0.32 μs: when system clock is operated at 6.29 MHz) to low-speed (5.09 μs)
- 67 I/O ports: 3 N-ch open-drain ports
- 8-bit resolution A/D converter: 16 channels
- DCAN controller: 1 channel (μPD780701Y and 78F0701Y only)
- IEBus™ controller: 1 channel (μPD780702Y and 78F0701Y only)
- Serial interface: 4 channels
  - 3-wire serial I/O mode: 2 channels
  - UART mode: 1 channel
  - I^2C mode: 1 channel
- Timer: 7 channels
  - 16-bit timer/event counter: 2 channels
  - 8-bit timer/event counter: 3 channels
  - Watch timer: 1 channel
  - Watchdog timer: 1 channel
- Vectored interrupt sources: 30
- Power supply voltage: V_{DD} = 3.5 to 5.5 V

1.2 Applications

Car audios, etc.

1.3 Ordering Information

Part Number	Package	Internal ROM
μPD780701YGC-×××-8BT	80-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780702YGC-×××-8BT	80-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780F0701YGC-8BT	80-pin plastic QFP (14 × 14 mm)	Flash memory

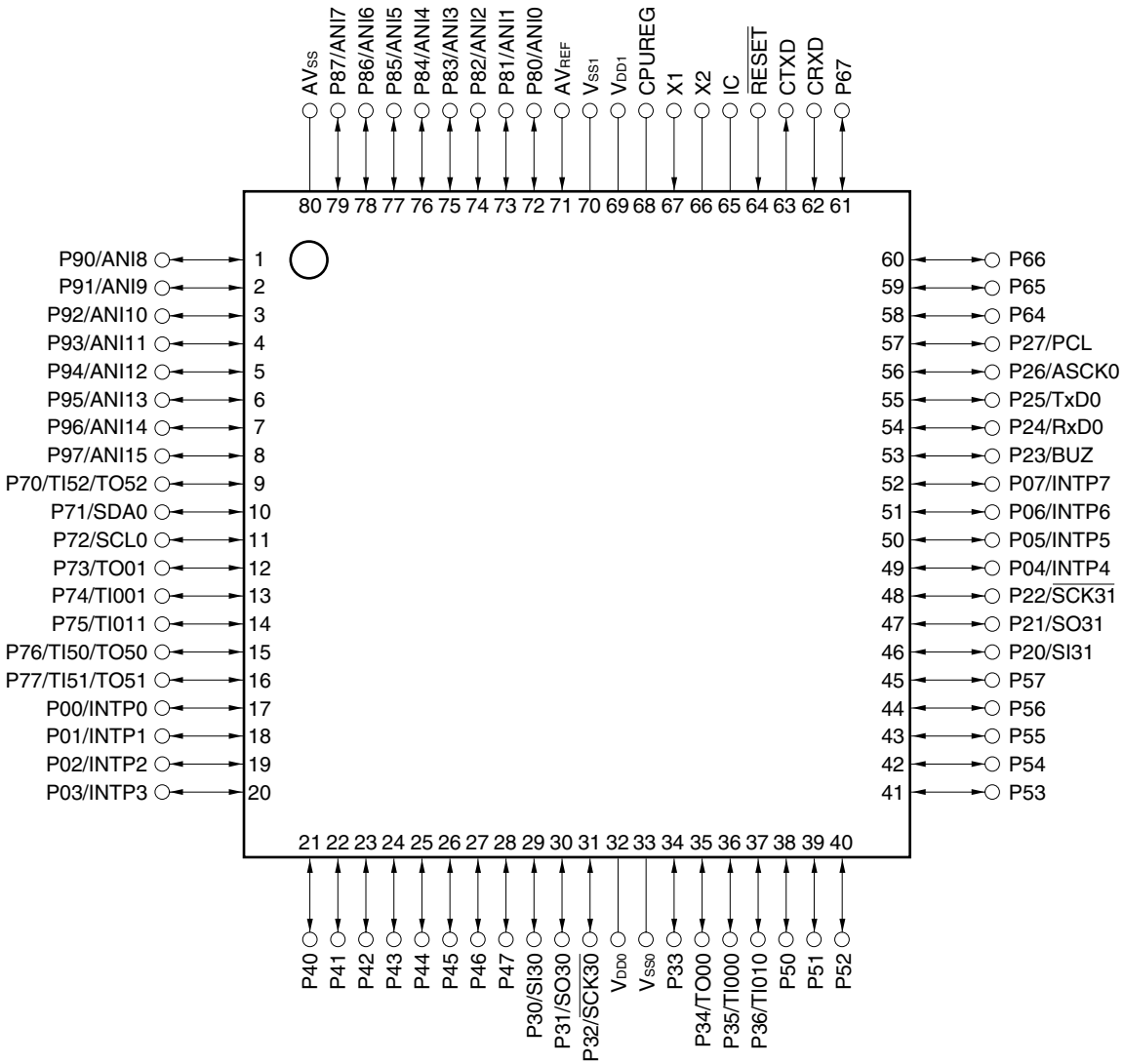
Remark ××× indicates ROM code suffix.

1.4 Pin Configuration (Top View)

(1) μPD780701Y

- **80-pin plastic QFP (14 × 14 mm)**

  μPD780701YGC-×××-8BT



Cautions 1. Connect the IC (Internally Connected) pin directly to V_{SS0} or V_{SS1}.

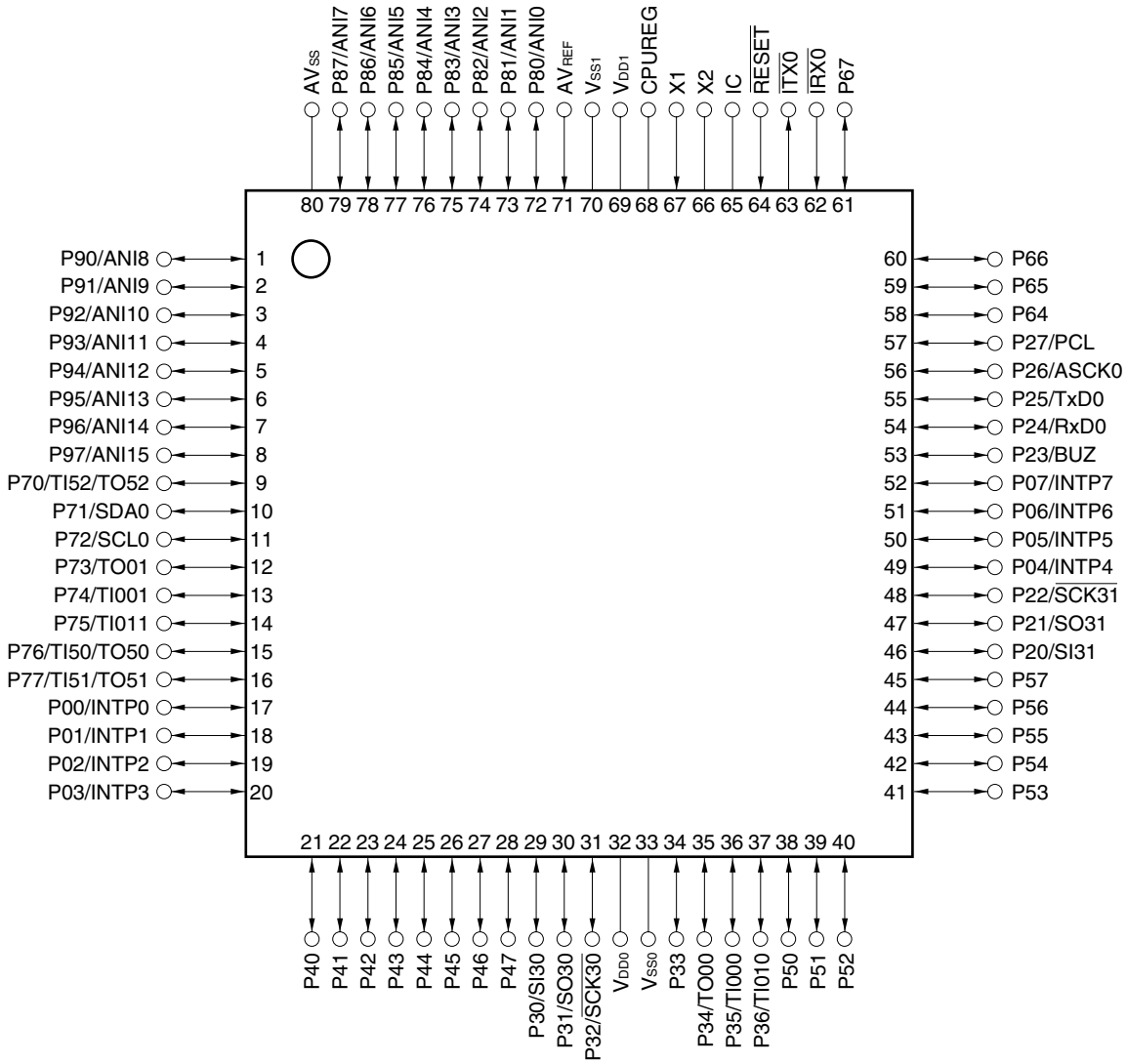
2. Connect the AV_{SS} pin to V_{SS0}.

3. Connect the AV_{REF} pin to V_{DD0}.

(2) μPD780702Y

- **80-pin plastic QFP (14 × 14 mm)**

  μPD780702YGC-×××-8BT



Cautions 1. Connect the IC (Internally Connected) pin directly to V_{SS0} or V_{SS1}.

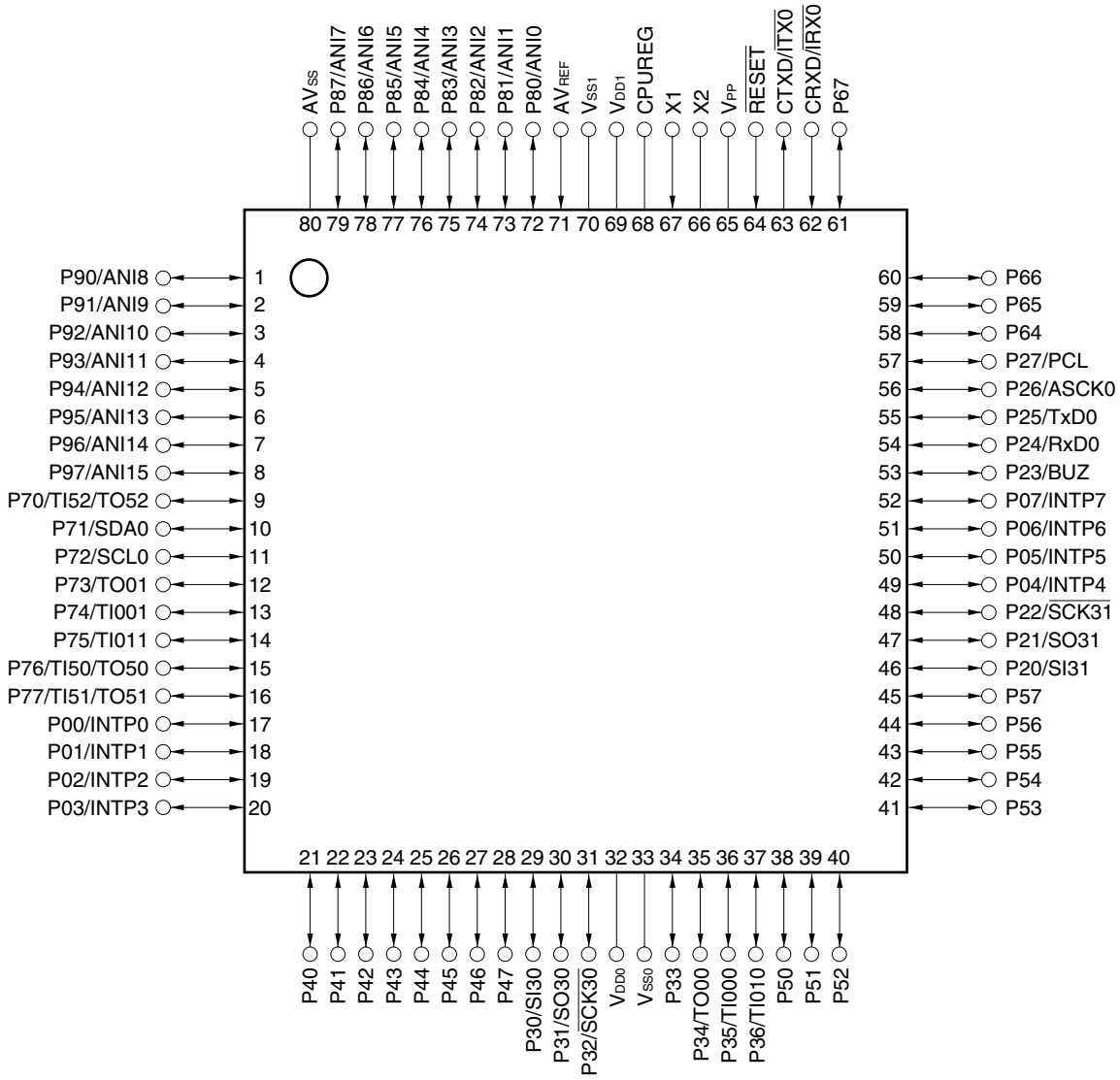
2. Connect the AV_{SS} pin to V_{SS0}.

3. Connect the AV_{REF} pin to V_{DD0}.

(3) μPD78F0701Y

- **80-pin plastic QFP (14 × 14 mm)**

μPD78F0701YGC-8BT



Cautions 1. In normal operation mode, connect the V_{PP} pin directly to V_{SS0} or V_{SS1}.

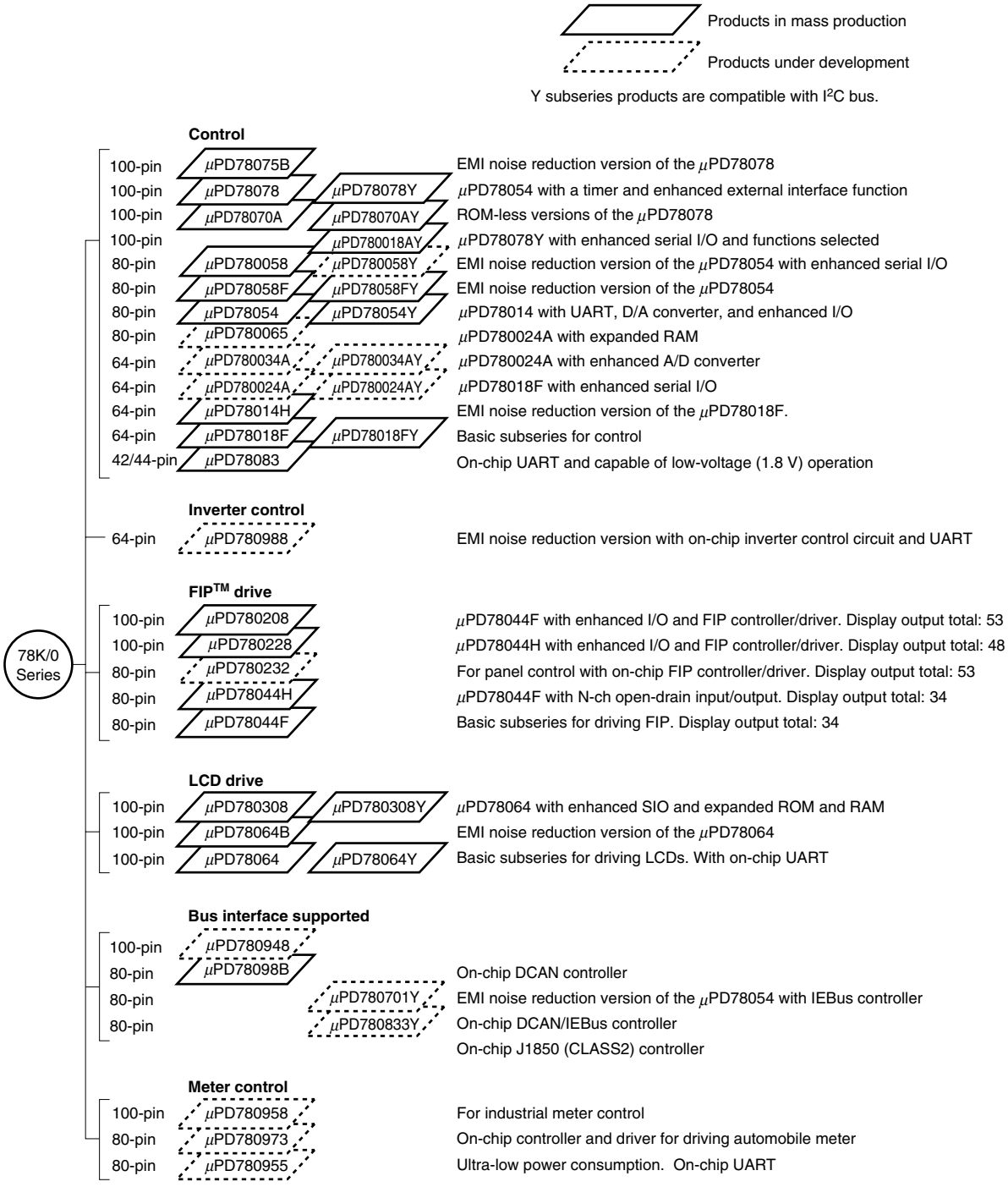
2. Connect the AV_{SS} pin to V_{SS0}.

3. Connect the AV_{REF} pin to V_{DD0}.

ANI0 to ANI15	:	Analog Input	P90 to P97	:	Port 9
ASCK0	:	Asynchronous Serial Clock	PCL	:	Programmable Clock
AV_{REF}	:	Analog Reference Voltage	$\overline{RESET}$	:	Reset
AV_{SS}	:	Analog Ground	RxD0	:	Receive Data (for UART0)
BUZ	:	Buzzer Output	$\overline{SCK30}$, $\overline{SCK31}$	:	Serial Clock (for SIO30, 31)
CPUREG	:	Regulator for CPU Power Supply	SCL0	:	Serial Clock (for IIC0)
CRXD	:	CAN Receive Data	SDA0	:	Serial Data
CTXD	:	CAN Transmit Data	SI30, SI31	:	Serial Input
IC	:	Internally Connected	SO30, SO31	:	Serial Output
INTP0 to INTP7	:	Interrupt from Peripherals	TI000, TI010,		
$\overline{IRX0}$	:	IEBus Receive Data	TI001, TI011,		
$\overline{ITX0}$	:	IEBus Transmit Data	TI50, TI51, TI52	:	Timer Input
P00 to P07	:	Port 0	TO00, TO01,		
P20 to P27	:	Port 2	TO50, TO51,		
P30 to P36	:	Port 3	TO52	:	Timer Output
P40 to P47	:	Port 4	TxD0	:	Transmit Data (for UART0)
P50 to P57	:	Port 5	V_{DD0}, V_{DD1}	:	Power Supply
P64 to P67	:	Port 6	V_{PP}	:	Programming Power Supply
P70 to P77	:	Port 7	V_{SS0}, V_{SS1}	:	Ground
P80 to P87	:	Port 8	X1, X2	:	Crystal

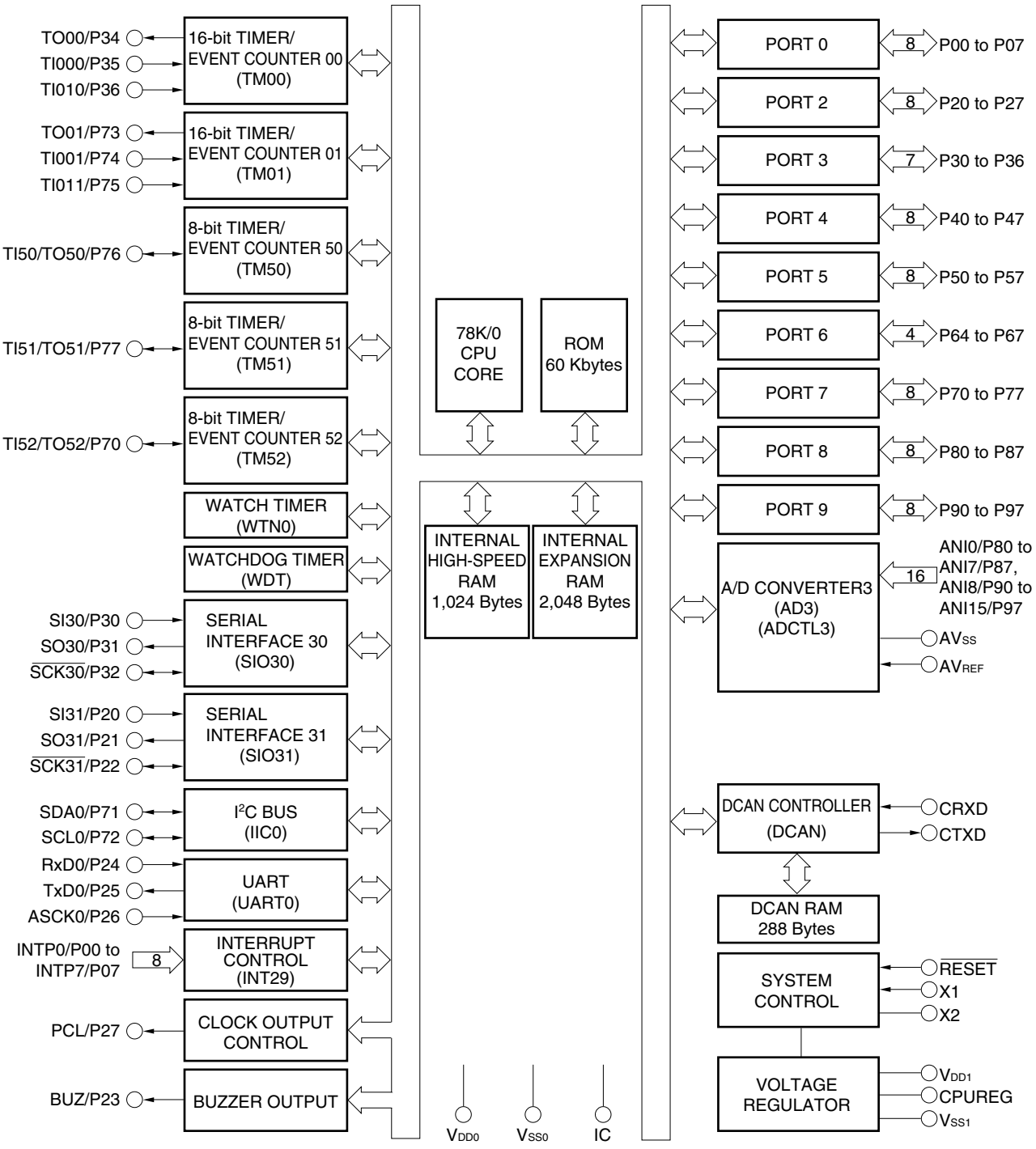
1.5 78K/0 Series Lineup

The products in the 78K/0 Series are listed below. The names enclosed in boxes are Subseries names.

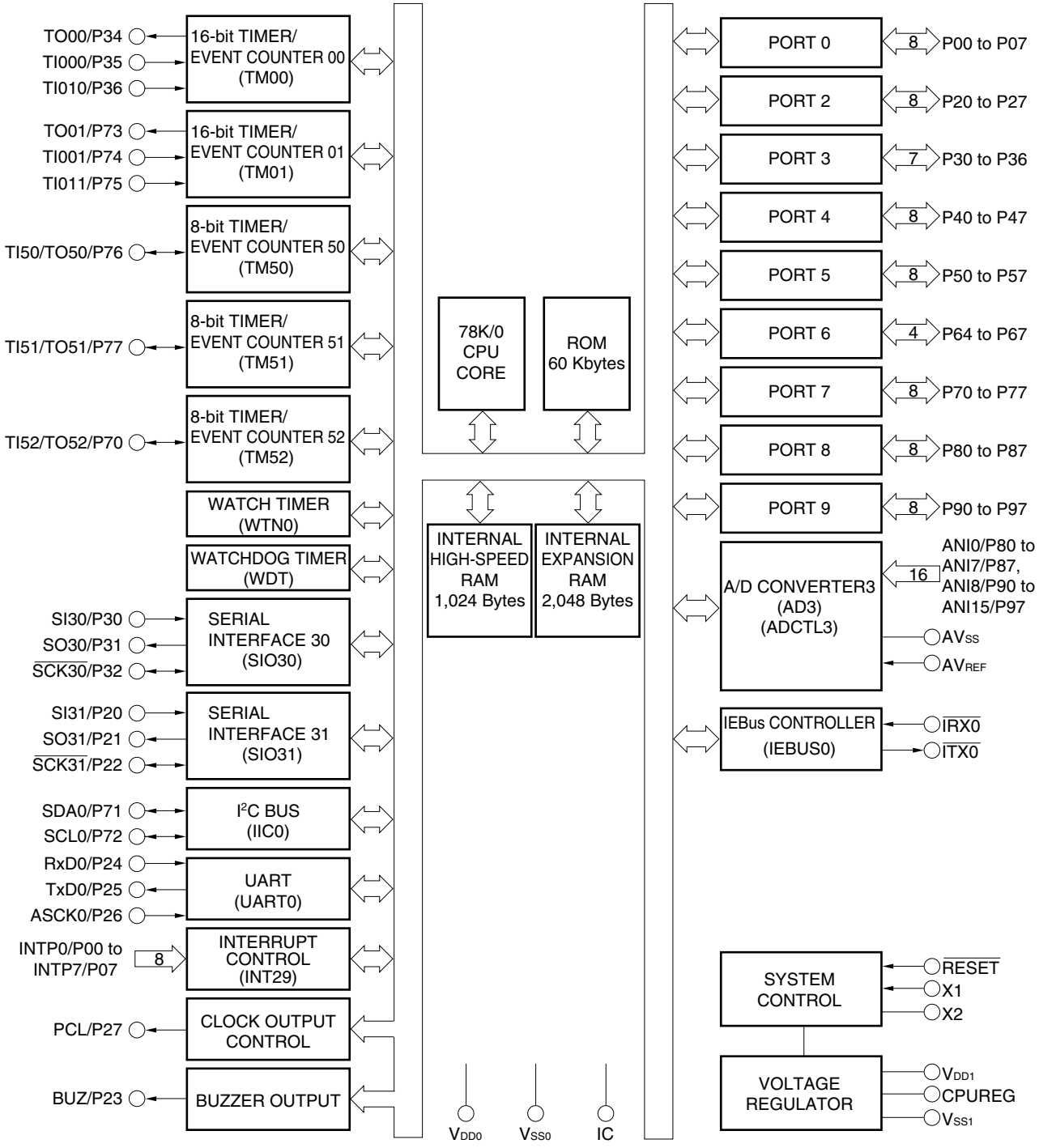


1.6 Block Diagram

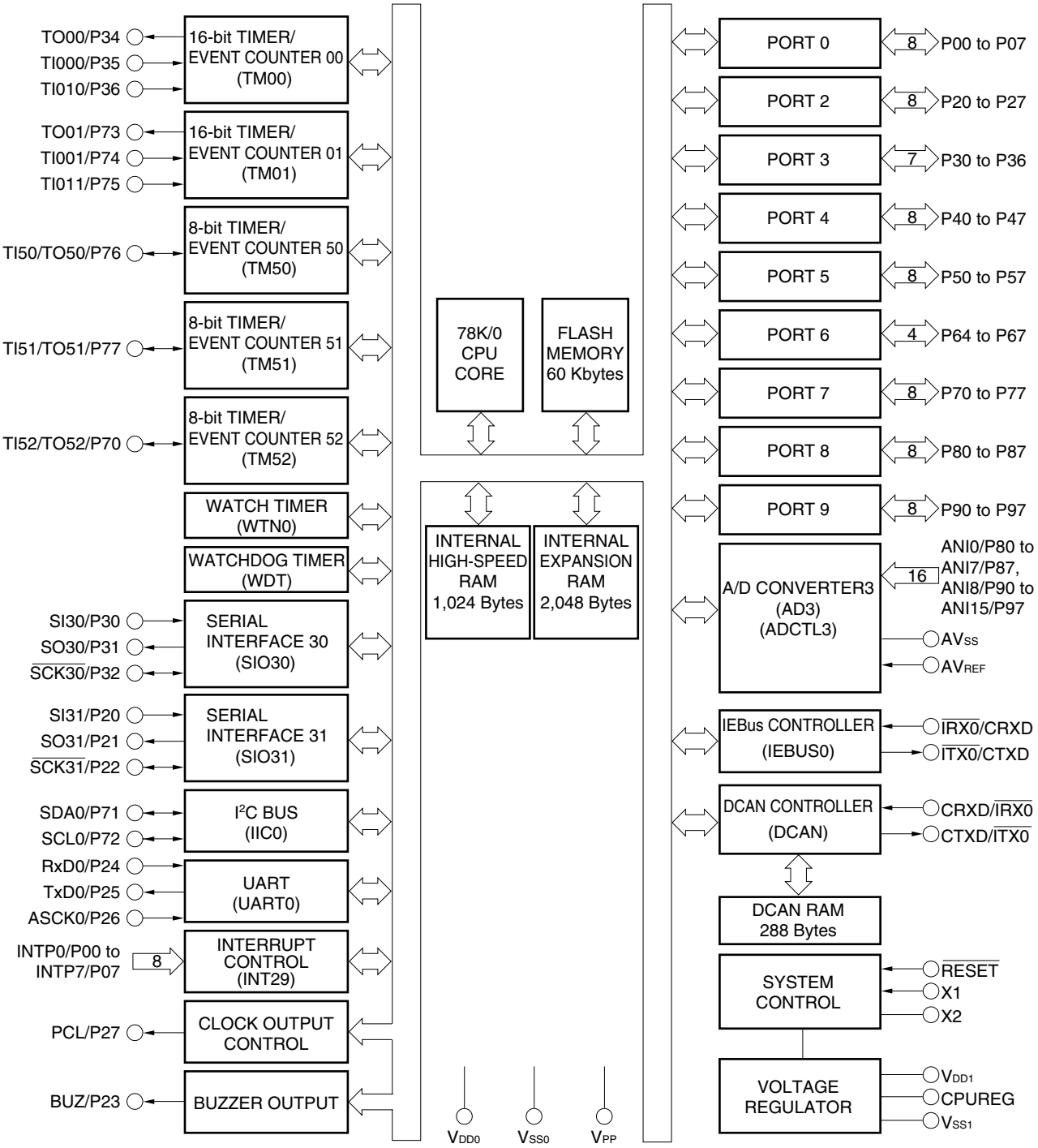
(1) μPD780701Y



(2) μPD780702Y



(3) μPD78F0701Y



1.7 Overview of Functions

Item		μPD780701Y	μPD780702Y	μPD78F0701Y
Internal memory	Flash memory	60 Kbytes (Mask ROM)		60 Kbytes (Flash memory)
	High-speed RAM	1,024 bytes		
	Expansion RAM	2,048 bytes		
	DCAN buffer RAM	288 bytes	—	288 bytes
Minimum instruction execution time		On-chip minimum instruction execution time variable function • 0.32 μs/0.64 μs/1.27 μs/2.54 μs/5.09 μs (System clock @ 6.29-MHz operation)		
General registers		8 bits × 32 registers (8 bits × 8 registers × 4 banks)		
Instruction set		• 16-bit operation • Multiply/divide (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, and Boolean operation) • BCD adjust, etc.		
I/O ports		Total: 67 • CMOS I/O: 56 • TTL input/CMOS output: 8 • N-ch open-drain I/O: 3		
A/D converter		• 8-bit resolution × 16 channels • Power-fail detection function		
Serial interface		• 3-wire serial I/O mode: 2 channels • UART mode: 1 channel • I^2C bus mode: 1 channel		
Timer		• 16-bit timer/event counter: 2 channels • 8-bit timer/event counter: 3 channels • Watch timer: 1 channel • Watchdog timer: 1 channel		
Timer outputs		5 (8-bit PWM output: 3)		
Bus controller		DCAN controller	IEBus controller	DCAN controller/ IEBus controller[Note]
Clock output		49.2 kHz, 98.3 kHz, 197 kHz, 393 kHz, 786 kHz, 1.57 MHz, 3.15 MHz, and 6.29 MHz (System clock @ 6.29-MHz operation)		
Buzzer output		0.768 kHz, 1.54 kHz, 3.07 kHz and, 6.14 kHz (System clock @ 6.29-MHz operation)		
Vectored interrupt sources	Maskable	Internal: 20, external: 8	Internal: 19, external: 8	Internal: 20, external: 8
	Non-maskable	Internal: 1		
	Software	1		
Power supply voltage		V_{DD} = 3.5 to 5.5 V		
Operating ambient temperature		T_A = –40 to +85°C		
Package		80-pin plastic QFP (14 × 14 mm)		

Note The μPD78F0701Y incorporates a DCAN controller and IEBus controller, however, they cannot be used at the same time.

CHAPTER 2 PIN FUNCTIONS

2.1 Pin Function List

(1) Port Pins (1/2)

Pin Name	I/O	Function		After Reset	Alternate Function
P00 to P07	I/O	Port 0 8-bit input/output port Input/output can be specified in 1-bit units. An on-chip pull-up resistor can be specified by means of software.		Input	INTP0 to INTP7
P20	I/O	Port 2 8-bit input/output port Input/output can be specified in 1-bit units. An on-chip pull-up resistor can be specified by means of software.		Input	SI31
P21					SO31
P22					$\overline{\text{SCK31}}$
P23					BUZ
P24					RxD0
P25					TxD0
P26					ASCK0
P27					PCL
P30	I/O	Port 3 7-bit input/output port Input/output can be specified in 1-bit units.	An on-chip pull-up resistor can be specified by means of software.	Input	SI30
P31					SO30
P32					$\overline{\text{SCK30}}$
P33			N-ch open-drain input/output port (15-V breakdown voltage). LEDs can be driven directly.		—
P34			An on-chip pull-up resistor can be specified by means of software.		TO00
P35					TI000
P36					TI010
P40 to P47	I/O	Port 4 8-bit input/output port Input/output can be specified in 1-bit units. An on-chip pull-up resistor can be specified by means of software. Interrupt request flag KRIF is set to 1 by falling edge detection.		Input	—
P50 to P57	I/O	Port 5 8-bit input/output port TTL level input/CMOS output Input/output can be specified in 1-bit units. An on-chip pull-up resistor can be specified by means of software.		Input	—
P64 to P67	I/O	Port 6 4-bit input/output port Input/output can be specified in 1-bit units. An on-chip pull-up resistor can be specified by means of software.		Input	—

(1) Port Pins (2/2)

Pin Name	I/O	Function		After Reset	Alternate Function
P70	I/O	Port 7 8-bit input/output port Input/output can be specified in 1-bit units.	An on-chip pull-up resistor can be specified by means of software.	Input	TI52/TO52
P71			N-ch open-drain input/output port (5-V breakdown voltage).		SDA0
P72					SCL0
P73			An on-chip pull-up resistor can be specified by means of software.		TO01
P74					TI001
P75					TI011
P76					TI50/TO50
P77					TI51/TO51
P80 to P87	I/O	Port 8 8-bit input/output port Input/output can be specified in 1-bit units.		Input	ANI0 to ANI7
P90 to P97	I/O	Port 9 8-bit input/output port Input/output can be specified in 1-bit units.		Input	ANI8 to ANI15

(2) Non-Port Pins (1/2)

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0 to INTP7	Input	External interrupt request input for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified	Input	P00 to P07
SI30	Input	Serial interface serial data input	Input	P30
SI31				P20
SO30	Output	Serial interface serial data output	Input	P31
SO31				P21
SDA0	I/O	Serial interface serial data input/output	Input	P71
$\overline{\text{SCK30}}$	I/O	Serial interface serial clock input/output	Input	P32
$\overline{\text{SCK31}}$				P22
SCL0				P72
RxD0	Input	Serial data input for asynchronous serial interface	Input	P24
TxD0	Output	Serial data output for asynchronous serial interface	Input	P25
ASCK0	Input	Serial clock input for asynchronous serial interface	Input	P26
CRXD	Input	DCAN controller (DCAN) data input (μPD780701Y, 78F0701Y only)	Input	$\overline{\text{IRX0}}$[Note]
CTXD	Output	DCAN controller (DCAN) data output (μPD780701Y, 78F0701Y only)	Output	$\overline{\text{ITX0}}$[Note]
$\overline{\text{IRX0}}$	Input	IEBus controller (IEBUS0) data input (μPD780702Y, 78F0701Y only)	Input	CRXD[Note]
$\overline{\text{ITX0}}$	Output	IEBus controller (IEBUS0) data output (μPD780702Y, 78F0701Y only)	Output	CTXD[Note]
TI000	Input	External clock input to 16-bit timer (TM00) and capture trigger input to capture register (CR010) of TM00.	Input	P35
TI010		Capture trigger input to capture register (CR000) of TM00.		P36
TI001		External clock input to 16-bit timer (TM01) and capture trigger input to capture register (CR011) of TM01.		P74
TI011		Capture trigger input to capture register (CR001) of TM01.		P75
TI50		External count clock input to 8-bit timer (TM50)		P76/TO50
TI51		External count clock input to 8-bit timer (TM51)		P77/TO51
TI52		External count clock input to 8-bit timer (TM52)		P70/TO52
TO00	Output	16-bit timer (TM00) output	Input	P34
TO01		16-bit timer (TM01) output		P73
TO50		8-bit timer (TM50) output		P76/TI50
TO51		8-bit timer (TM51) output		P77/TI51
TO52		8-bit timer (TM52) output		P70/TI52
PCL	Output	Clock output	Input	P27
BUZ	Output	Buzzer output	Input	P23
ANI0 to ANI7	Input	A/D converter (AD3) analog input	Input	P80 to P87
ANI8 to ANI15				P90 to P97

Note Applies to the μPD78F0701Y only.

(2) Non-Port Pins (2/2)

Pin Name	I/O	Function	After Reset	Alternate Function
AV_{REF}	Input	A/D converter (AD3) reference voltage and analog power supply input	—	—
AV_{SS}	—	A/D converter (AD3) ground potential	—	—
X1	Input	Connecting crystal resonator for system clock oscillation	—	—
X2	—		—	—
$\overline{\text{RESET}}$	Input	System reset input	Input	—
CPUREG	—	CPU power supply regulator. Connect to V_{SS0} or V_{SS1} via a 0.1-μF capacitor	—	—
V_{DD0}	—	Positive power supply for ports	—	—
V_{DD1}	—	Positive power supply (other than port and analog)	—	—
V_{SS0}	—	Ground potential for ports	—	—
V_{SS1}	—	Ground potential (other than port and analog)	—	—
IC	—	Internally connected. Connect directly to V_{SS0} or V_{SS1}	—	—
V_{PP}	—	High-voltage application for program write/verify. Connect directly to V_{SS0} or V_{SS1} in normal operation mode	—	—

2.2 Description of Pin Functions

2.2.1 P00 to P07 (Port 0)

These pins constitute an 8-bit input/output port. Besides serving as an input/output port, they function as external interrupt inputs.

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

(1) Port mode

In this mode, these pins function as an 8-bit input/output port.

P00 to P07 can be specified as input or output in 1-bit units with port mode register 0 (PM0). On-chip pull-up resistors can be specified by setting pull-up resistor option register 0 (PU0).

(2) Control mode

In this mode, these pins function as external interrupt request inputs (INTP0 to INTP7).

INTP0 to INTP7 are external interrupt request input pins whose valid edges (rising edge, falling edge, and both rising and falling edges) can be specified.

2.2.2 P20 to P27 (Port 2)

These pins constitute an 8-bit input/output port. Besides serving as an input/output port, they function as the serial data input/output and the serial clock input/output of the serial interface, buzzer output, and clock output.

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

(1) Port mode

In this mode, these pins function as an 8-bit input/output port.

They can be specified in 1-bit units as input or output with port mode register 2 (PM2). On-chip pull-up resistors can be specified in 1-bit units by setting pull-up resistor option register 2 (PU2).

(2) Control mode

In this mode, these pins function as serial data input/output and the serial clock input/output of the serial interface, buzzer output, and clock output functions.

(a) SI31 and SO31

Serial interface (SIO31) serial data input/output pins.

(b) $\overline{\text{SCK31}}$

Serial interface (SIO31) serial clock input/output pin.

(c) RxD0, TxD0

Asynchronous serial interface serial data input/output pins.

(d) ASCK0

Asynchronous serial interface serial clock input pin.

(e) BUZ

Buzzer output pin.

(f) PCL

Clock output pin.

2.2.3 P30 to P36 (Port 3)

These pins constitute a 7-bit input/output port. Besides serving as an input/output port, they function as the serial data input/output and the serial clock input/output of the serial interface, and timer input/output.

P33 can drive LEDs directly.

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

(1) Port mode

In this mode, these pins function as a 7-bit input/output port.

They can be specified in 1-bit units as input/output with port mode register 3 (PM3). P33 is N-ch open-drain input/output port.

On-chip pull-up resistors can be specified for P30 to P32 and P34 to P36 in 1-bit units by setting pull-up resistor option register 3 (PU3). P35 and P36 are also 16-bit timer/event counter (TM00) capture trigger signal input pins with a valid edge input.

(2) Control mode

In this mode, these pins function as the serial clock input/output and the serial data input/output of the serial interface, and timer input/output.

(a) SI30 and SO30

Serial interface (SIO30) serial data input/output pins.

(b) $\overline{\text{SCK30}}$

Serial interface (SIO30) serial clock input/output pin.

(c) TO00

Timer output pin.

(d) TI000

External count clock input pin to the 16-bit timer/event counter (TM00) and capture trigger signal input pin to the TM00 capture register (CR010).

(e) TI010

Capture trigger signal input pin to the capture register (CR000) of the 16-bit timer/event counter (TM00).

2.2.4 P40 to P47 (Port 4)

These pins constitute an 8-bit input/output port.

The interrupt request flag (KRIF) can be set to 1 with the detection of a falling edge.

They can be specified in 1-bit units as input/output with port mode register 4 (PM4). On-chip pull-up resistors can be specified in 1-bit units by setting pull-up resistor option register 4 (PU4).

2.2.5 P50 to P57 (Port 5)

These pins constitute an 8-bit input/output port. TTL level input/CMOS output. They can be specified in 1-bit units as input/output with port mode register 5 (PM5). On-chip pull-up resistors can be specified in 1-bit units by setting pull-up resistor option register 5 (PU5).

2.2.6 P64 to P67 (Port 6)

These pins constitute a 4-bit input/output port. They can be specified in 1-bit units as input/output with port mode register 6 (PM6). On-chip pull-up resistors can be specified in 1-bit units by setting pull-up resistor option register 6 (PU6).

2.2.7 P70 to P77 (Port 7)

These pins constitute an 8-bit input/output port. Besides serving as input/output port, they function as the timer input/output, serial data input/output and serial clock input/output of the serial interface.

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

(1) Port mode

In this mode, these pins function as an 8-bit input/output port.

They can be specified in 1-bit units as input/output with port mode register 7 (PM7). P71 and P72 are N-ch open-drain input/output port.

On-chip pull-up resistors can be specified for P70, P73 to P77 in 1-bit units by setting pull-up resistor option register 7 (PU7). P74 and P75 are also 16-bit timer/event counter (TM01) capture trigger signal input pins with a valid edge input.

(2) Control mode

In this mode, these pins function as the timer input/output, serial data input/output and serial clock input/output of the serial interface.

(a) TO01, TO50 to TO52

Timer output pins.

(b) TI001

External count clock input pin to the 16-bit timer/event counter (TM01) and capture trigger signal input pin to the TM01 capture register (CR011).

(c) TI011

Capture trigger signal input pin to the capture register (CR001) of the 16-bit timer/event counter (TM01).

(d) TI50 to TI52

External count clock input pins to 8-bit timer/event counter.

(e) SDA0

Serial interface (IIC0) serial data input/output pin.

(f) SCL0

Serial interface (IIC0) serial clock input/output pin.

2.2.8 P80 to P87 (Port 8)

These pins constitute an 8-bit input/output port. Besides serving as input/output port, they function as A/D converter analog inputs.

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

(1) Port Mode

In this mode, these pins function as an 8-bit input/output port. They can be specified in 1-bit units as input/output with port mode register 8 (PM8).

(2) Control Mode

In this mode, they function as A/D converter analog input pins (ANI0 to ANI7).

2.2.9 P90 to P97 (Port 9)

These pins constitute an 8-bit input/output port. Besides serving as input/output port, they function as A/D converter analog inputs.

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

(1) Port Mode

In this mode, these pins function as an 8-bit input/output port. They can be specified in 1-bit units as input/output with port mode register 9 (PM9).

(2) Control Mode

In this mode, these pins function as A/D converter analog input pins (ANI8 to ANI15).

2.2.10 CTXD (μPD780701Y and 78F0701Y only)

Data output pin for the DCAN controller.

2.2.11 CRXD (μPD780701Y and 78F0701Y only)

Data input pin for the DCAN controller.

2.2.12 $\overline{\text{ITX0}}$ (μPD780702Y and 78F0701Y only)

Data output pin for the IEBus controller.

2.2.13 $\overline{\text{IRX0}}$ (μPD780702Y and 78F0701Y only)

Data input pin for the IEBus controller.

2.2.14 AV_{REF}

A/D converter reference voltage input pin. It can also be used as an analog power supply pin. Keep this pin at the same voltage as the V_{DD0} pin even when the A/D converter is not used.

2.2.15 AV_{SS}

A/D converter ground potential. Keep this pin at the same voltage as the V_{SS0} pin even when the A/D converter is not used.

2.2.16 $\overline{\text{RESET}}$

Low-level active system reset input pin.

2.2.17 X1 and X2

Crystal resonator connection pins for system clock oscillation.

For external clock supply, input clock signal to X1 and its inverted signal to X2.

2.2.18 CPUREG

Regulator pin for the CPU power supply. Connect this pin to V_{SS0} or V_{SS1} via a 0.1-μF capacitor.

2.2.19 V_{DD0} and V_{DD1}

V_{DD0} is a positive power supply pin for ports.

V_{DD1} is a positive power supply pin except for ports.

2.2.20 V_{SS0} and V_{SS1}

V_{SS0} is a ground potential pin for ports.

V_{SS1} is a ground potential pin except for ports.

2.2.21 V_{PP} (μPD78F0701Y only)

High-voltage apply pin for flash memory programming mode setting and program write/verify.

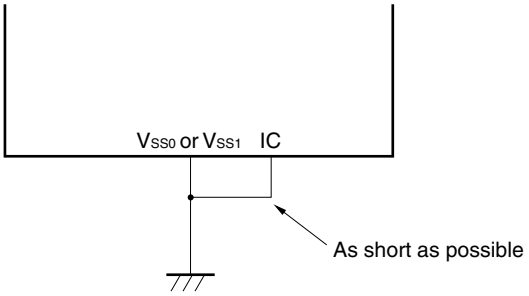
Connect directly to V_{SS0} or V_{SS1} in the normal operating mode.

2.2.22 IC (μPD780701Y and 780702Y)

The IC (Internally Connected) pin is provided to set the test mode to check the μPD780701Y Subseries at delivery. Connect this pin directly to V_{SS0} or V_{SS1} with the shortest possible wire in the normal operating mode.

When a voltage difference is produced between the IC pin and V_{SS0} pin or V_{SS1} pin, because the wiring between those two pins is too long or an external noise is input to the IC pin, the user's program may not operate normally.

- **Connect IC pins to V_{SS0} pins or V_{SS1} pins directly.**



2.3 Pin Input/Output Circuits and Recommended Connection of Unused Pins

Table 2-1 shows the types of pin input/output circuits and the recommended connections of unused pins. Refer to Figure 2-1 for the configuration of each type of input/output circuit.

Table 2-1. Pin Input/Output Circuit Types and Recommended Connection of Unused Pins (1/2)

Pin Name	Input/Output Circuit Type	Input/Output	Recommended Connection of Unused Pins
P00/INTP0 to P07/INTP7	8-C	Input/output	Independently connect to V_{SS0} via a resistor.
P20/SI31			Independently connect to V_{DD0} or V_{SS0} via a resistor.
P21/SO31	5-H		
P22/$\overline{SCK31}$	8-C		
P23/BUZ	5-H		
P24/RxD0	8-C		
P25/TxD0	5-H		
P26/ASCK0	8-C		
P27/PCL	5-H		
P30/SI30	8-C		
P31/SO30	5-H		
P32/$\overline{SCK30}$	8-C		
P33	13-P		Connect to V_{DD0} via a resistor.
P34/TO00	5-H		Independently connect to V_{DD0} or V_{SS0} via a resistor.
P35/TI000	8-C		
P36/TI010			
P40 to P47	5-H		Independently connect to V_{DD0} via a resistor.
P50 to P57	5-T		Independently connect to V_{DD0} or V_{SS0} via a resistor.
P64 to P67	5-H		
P70/TI52/TO52			
P71/SDA0	13-R		Independently connect to V_{DD0} via a resistor.
P72/SCL0			
P73/TO01	5-H		Independently connect to V_{DD0} or V_{SS0} via a resistor.
P74/TI001	8-C		
P75/TI011			
P76/TI50/TO50			
P77/TI51/TO51			
P80/ANI0 to P87/ANI7	11-E		
P90/ANI8 to P97/ANI15			

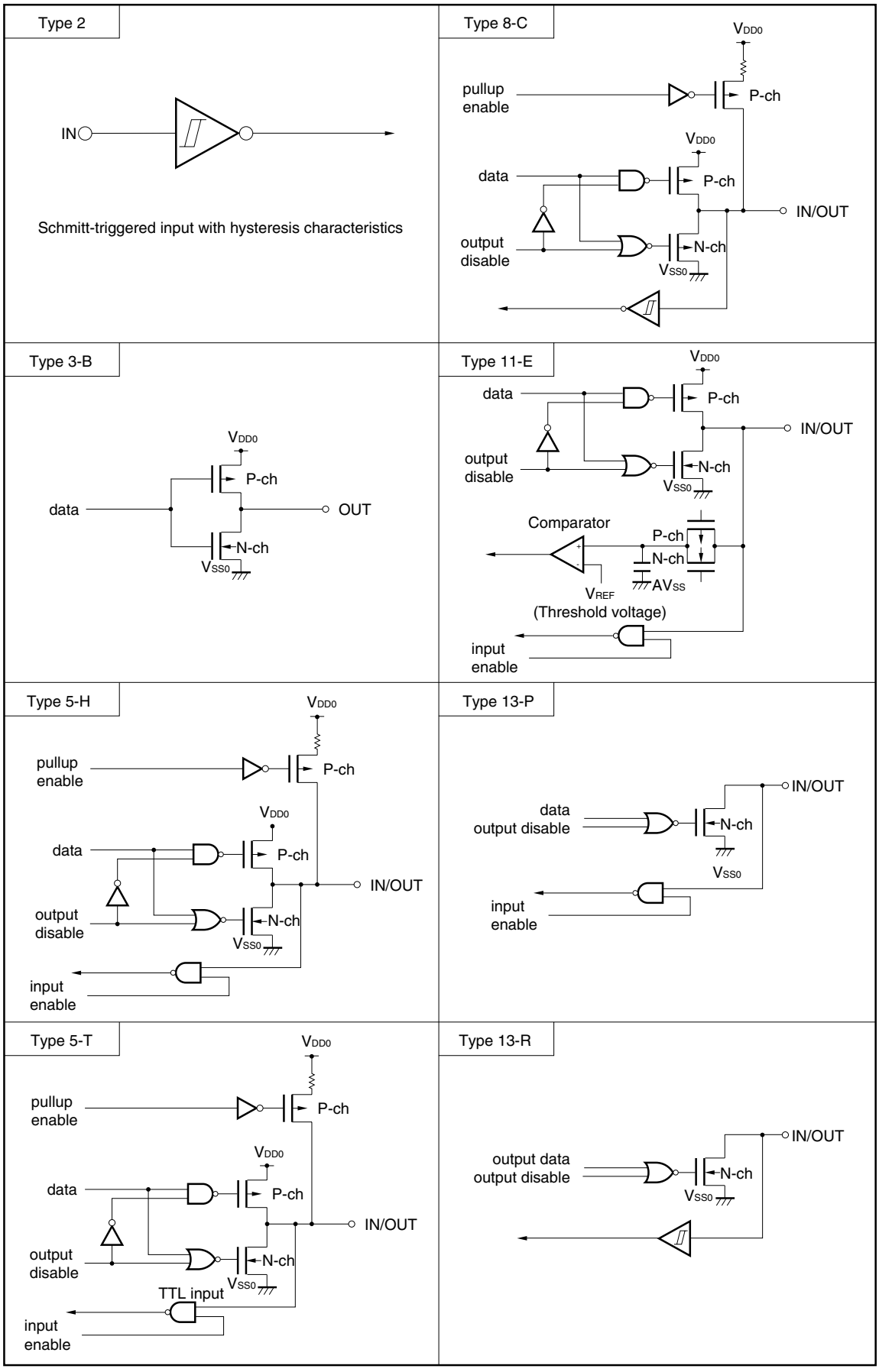
Table 2-1. Pin Input/Output Circuit Types and Recommended Connection of Unused Pins (2/2)

Pin Name	Input/Output Circuit Type	Input/Output	Recommended Connection of Unused Pins
CRXD[Note 1]/$\overline{\text{IRX0}}$[Note 2]	2	Input	Connect to V_{DD0} or V_{SS0} via a resistor.
CTXD[Note 1]/$\overline{\text{ITX0}}$[Note 2]	3-B	Output	Leave open.
$\overline{\text{RESET}}$	2	Input	—
AV_{REF}	—		Connect to V_{DD0}.
AV_{SS}		—	Connect to V_{SS0}.
V_{PP}			Connect directly to V_{SS0} or V_{SS1}.
IC			

Notes 1. Applies to the μPD780701Y and 78F0701Y only.

2. Applies to the μPD780702Y and 78F0701Y only.

Figure 2-1. Pin Input/Output Circuits



CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Space

Each product in the μPD780701Y Subseries can access a memory space of 64 Kbytes.
Figures 3-1 to 3-3 show the memory maps of the respective products.

Figure 3-1. Memory Map (μPD780701Y)

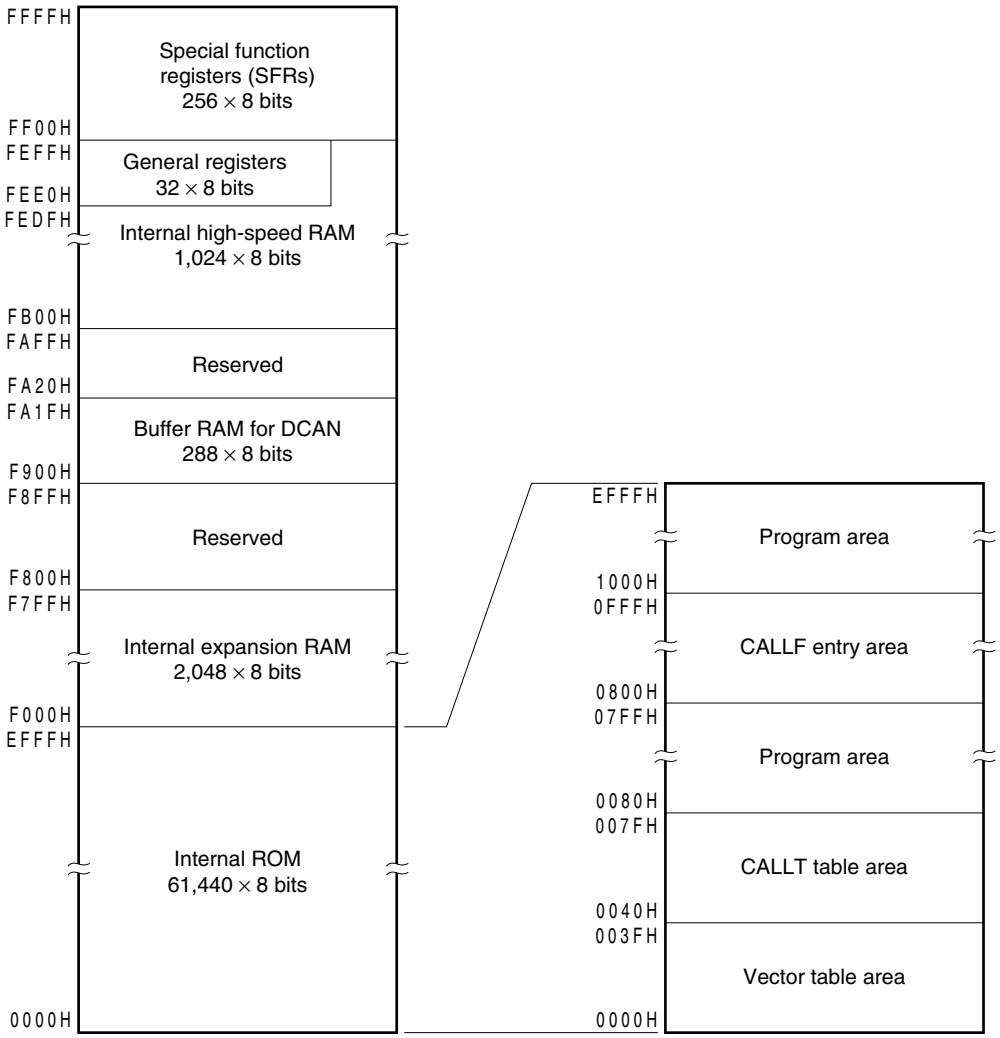


Figure 3-2. Memory Map (μPD780702Y)

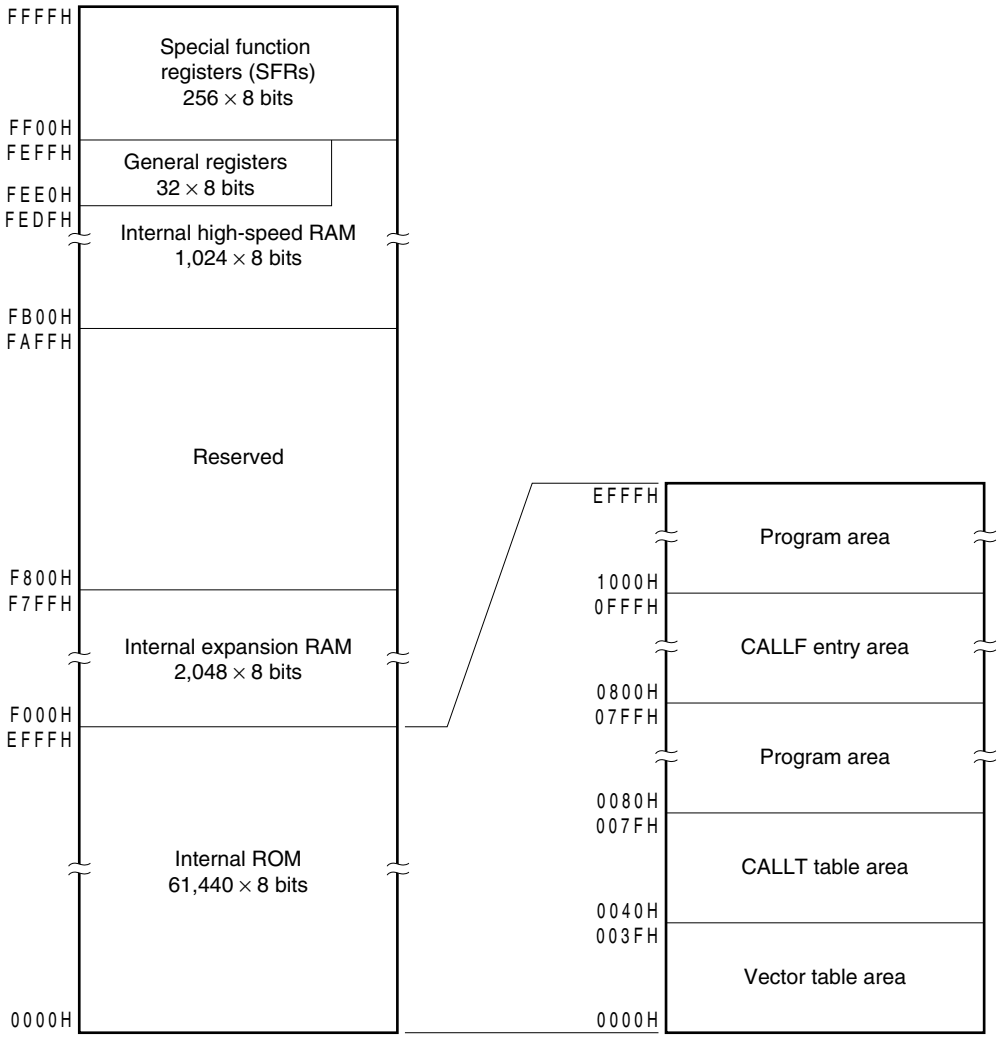
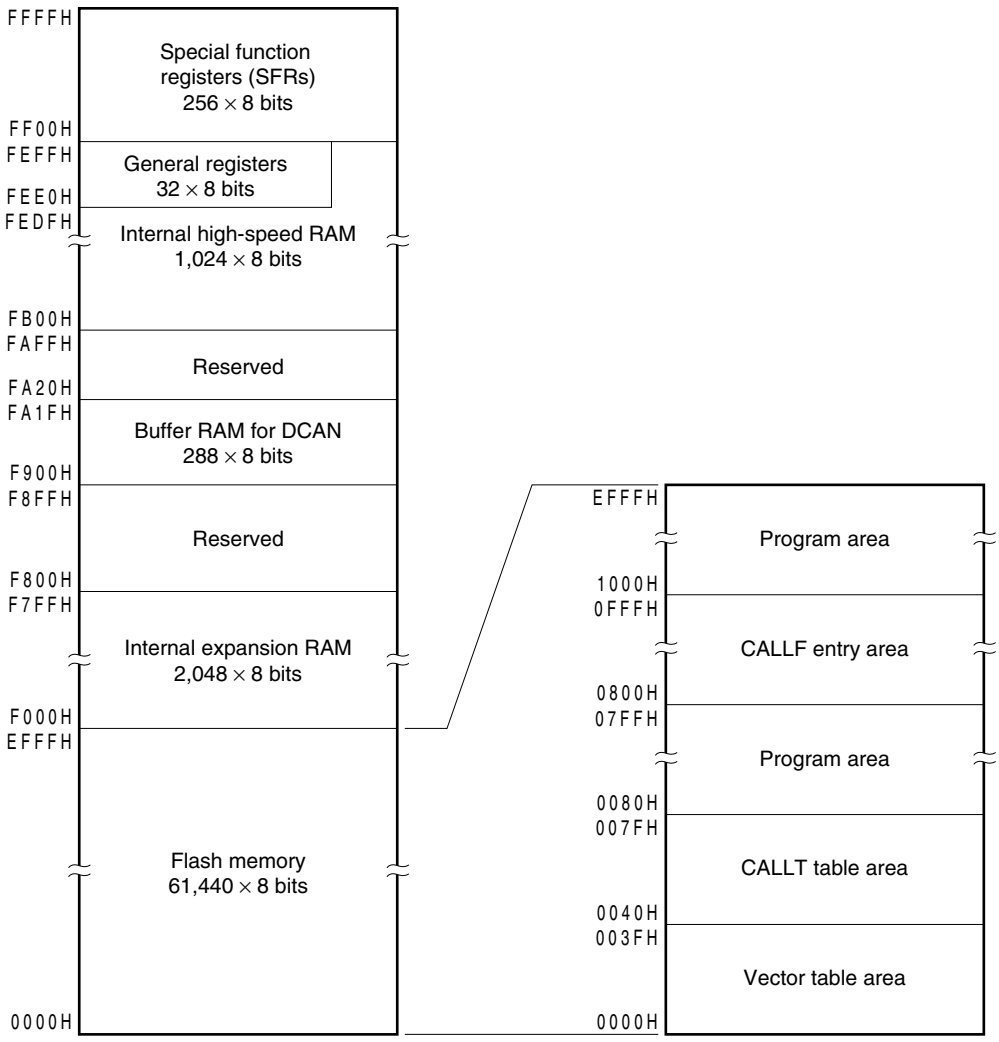


Figure 3-3. Memory Map (μPD78F0701Y)



3.1.1 Internal program memory space

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

The μPD780701Y Subseries products are provided with on-chip ROM (or flash memory).

Table 3-1. Internal Memory Capacity

Product	Internal ROM	
	Structure	Capacity
μPD780701Y, 780702Y	Mask ROM	61,440 × 8 bits (0000H to EFFFH)
μPD78F0701Y	Flash memory	

The following areas are allocated to the internal program memory space.

(1) Vector table area

The 64-byte area of 0000H to 003FH is reserved as a vector table area. The $\overline{\text{RESET}}$ input and program start addresses for branch upon generation of each interrupt request are stored in the vector table area.

Of the 16-bit addresses, low-order 8 bits are stored at even addresses and high-order 8 bits are stored at odd addresses.

Table 3-2. Vector Table

Vector Table Address	Interrupt Source
0000H	$\overline{\text{RESET}}$ input
0004H	INTWDT
0006H	INTP0
0008H	INTP1
000AH	INTP2
000CH	INTP3
000EH	INTP4
0010H	INTP5
0012H	INTP6
0014H	INTP7
0016H	INTSER0
0018H	INTSR0
001AH	INTST0
001CH	INTCSI30
001EH	INTCSI31
0020H	INTIIC0
0022H	INTCE**Note 1**
0024H	INTCR**Note 1**/INTIE1**Note 2**
0026H	INTCT**Note 1**/INTIE2**Note 2**
0028H	INTWTNI0
002AH	INTTM000
002CH	INTTM010
002EH	INTTM001
0030H	INTTM011
0032H	INTTM50
0034H	INTTM51
0036H	INTTM52
0038H	INTAD
003AH	INTWTN0
003CH	INTKR
003EH	BRK

Notes 1. Applies to μPD780701Y and 78F0701Y only.

2. Applies to μPD780702Y and 78F0701Y only.

(2) CALLT instruction table area

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

(3) CALLF instruction entry area

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

3.1.2 Internal data memory space

In the μPD780701Y Subseries, the following on-chip RAM is provided.

(1) Internal high-speed RAM

This RAM is configured from the 1,024 × 8 bits of addresses FB00H to FEFFH. In this area, 4 general register banks with eight 8-bit registers as 1 bank are allocated to the 32-byte area of FEE0H to FEFFH.

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

(2) DCAN buffer RAM (μPD780701Y and 78F0701Y)

The DCAN buffer RAM is allocated to the 288-byte area of F900H to FA1FH.

This DCAN buffer RAM can also be used as normal RAM.

(3) Internal expansion RAM

The internal expansion RAM is allocated to the 2,048-byte area of F000H to F7FFH.

3.1.3 Special function register (SFR) area

On-chip peripheral hardware special-function registers (SFRs) are allocated to the area of FF00H to FFFFH. (Refer to **Table 3-4. Special Function Register List** in **3.2.3 Special function register (SFR)**.)

Caution Do not access addresses where SFRs are not assigned.

3.1.4 External memory space

This is external memory space that can be accessed by setting the memory expansion mode register (MEM). This external memory space is used to store programs and table data peripheral devices can be allocated to this space.

3.1.5 Data memory addressing

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

The address of the instruction to be executed next is addressed by the program counter (PC) (for details, see **3.3 Instruction Address Addressing**).

Several addressing modes are provided for addressing the memory relevant to the execution of instructions for the μPD780701Y Subseries, based on operability and other considerations. For areas containing data memory in particular, special addressing methods designed for the functions of special function registers (SFR) and general registers are available for use. Data memory addressing is illustrated in Figures 3-4 to 3-6. For the details of each addressing mode, see **3.4 Operand Address Addressing**.

Figure 3-4. Data Memory Addressing (μPD780701Y)

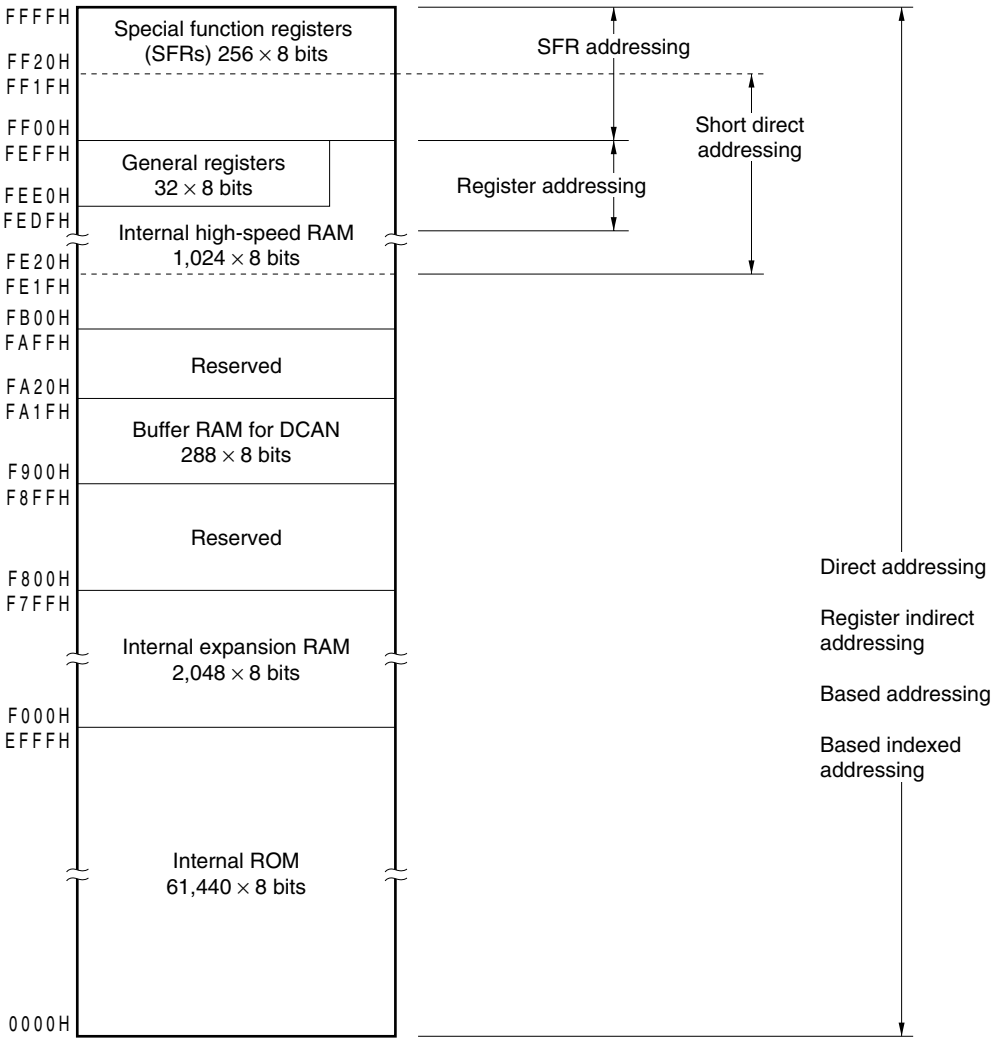


Figure 3-5. Data Memory Addressing (μPD780702Y)

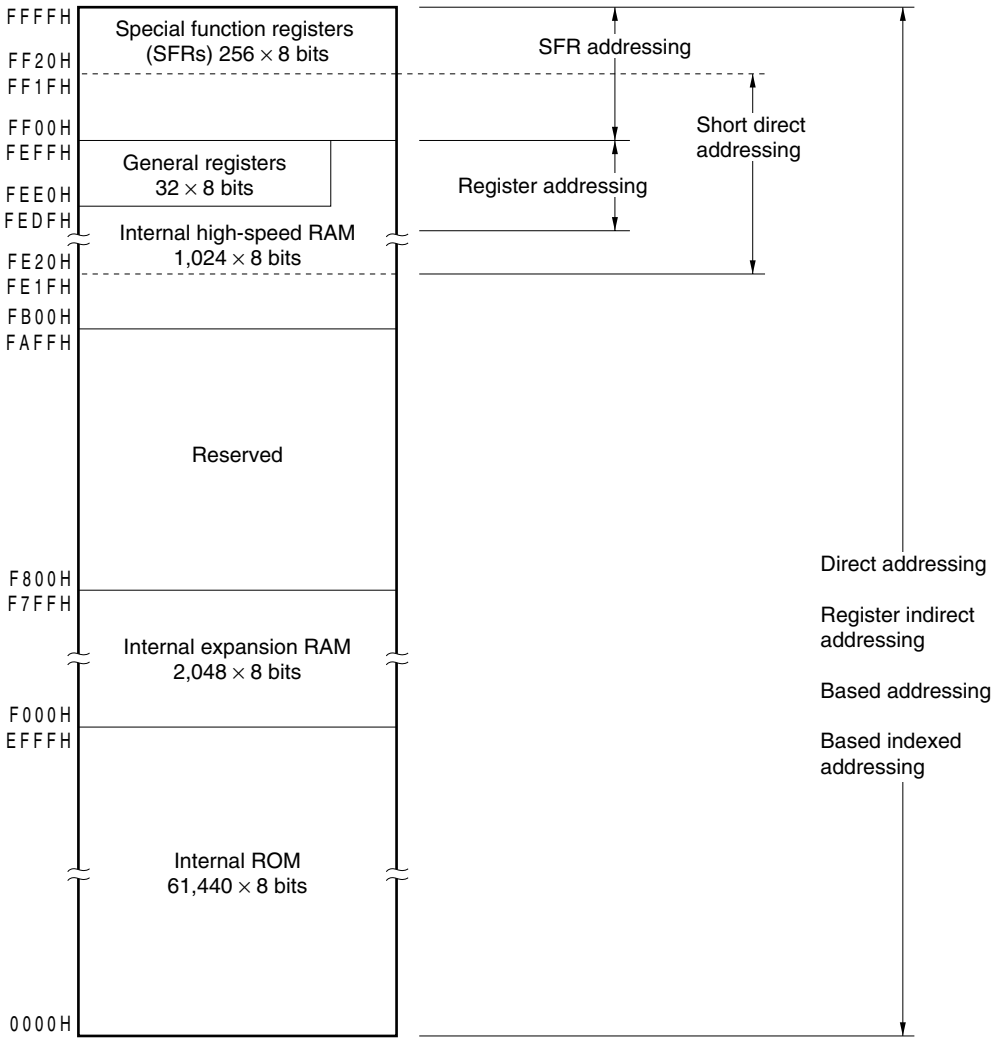
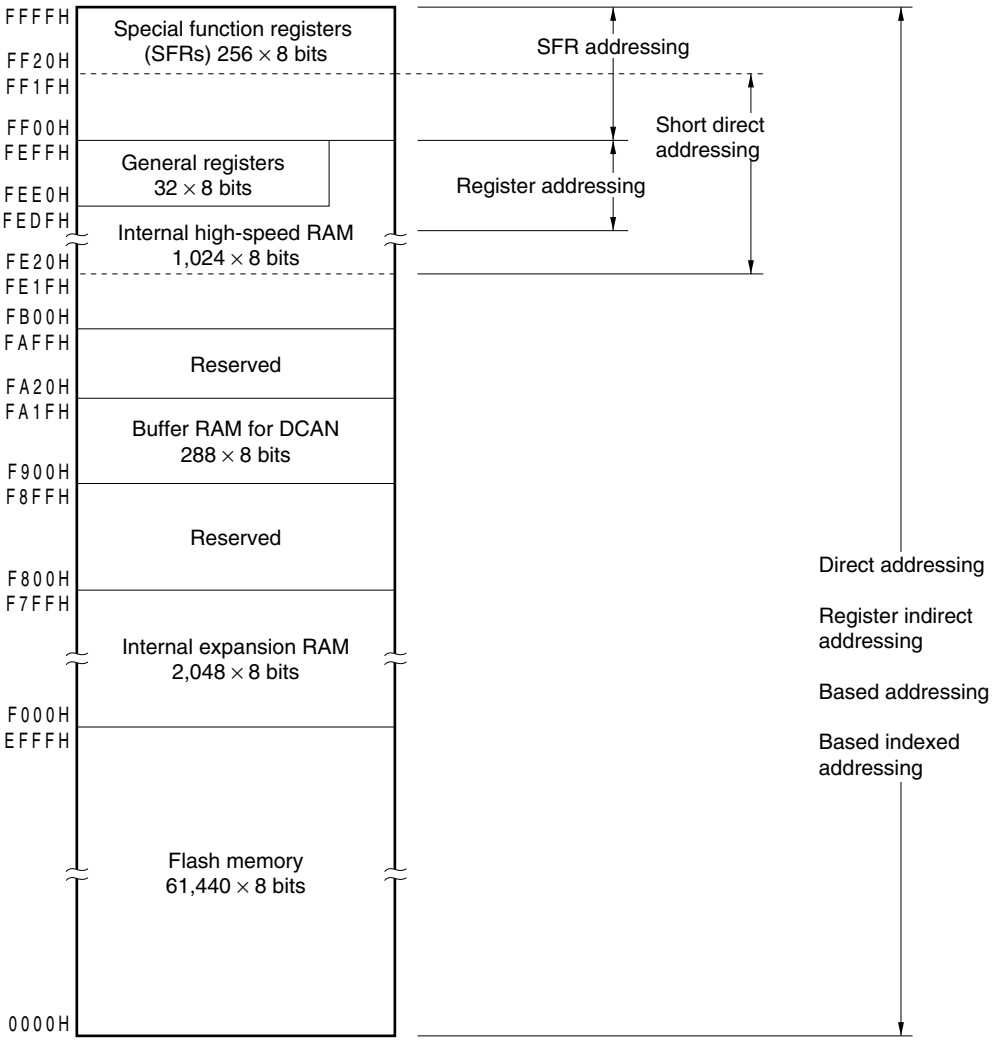


Figure 3-6. Data Memory Addressing (μPD78F0701Y)



3.2 Processor Registers

The μPD780701Y Subseries products incorporate the following processor registers.

3.2.1 Control registers

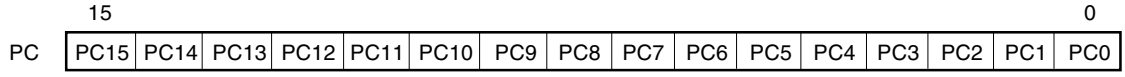
The control registers control the program sequence, statuses, and stack memory. The control registers consist of a program counter (PC), a program status word (PSW) and a stack pointer (SP).

(1) Program counter (PC)

The program counter is a 16-bit register that holds the address information of the next program to be executed. In normal operation, the PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data and register contents are set.

$\overline{\text{RESET}}$ input sets the reset vector table values of addresses 0000H and 0001H to the program counter.

Figure 3-7. Program Counter Format



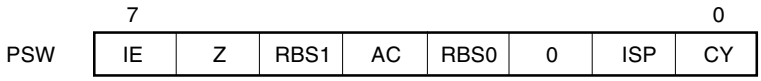
(2) Program status word (PSW)

The program status word is an 8-bit register consisting of various flags to be set/reset according to instruction execution.

Program status word contents are automatically stacked upon interrupt request generation or PUSH PSW instruction execution and are automatically reset upon execution of the RETB, RETI, and POP PSW instructions.

$\overline{\text{RESET}}$ input sets PSW to 02H.

Figure 3-8. Program Status Word Format



(a) Interrupt enable flag (IE)

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

When IE is 0, the CPU enters the disable interrupt (DI) state, and only non-maskable interrupt requests become acknowledgeable. Other interrupt requests are disabled.

When IE is 1, the CPU enters the enable interrupt (EI) state and the interrupt request acknowledge is controlled with an in-service priority flag (ISP), an interrupt mask flag for various interrupt sources, and a priority specification flag.

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

(b) Zero flag (Z)

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

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

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

They store the 2-bit information that indicates the register bank selected by SEL RBn instruction execution.

(d) Auxiliary carry flag (AC)

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

(e) In-service priority flag (ISP)

This flag manages the priority of acknowledgeable maskable vectored interrupts. When this flag is 0, low-level vectored interrupt requests specified with the priority specify flag registers (PR0L, PR0H, PR1L, PR1H) (refer to **17.3 (3) Priority specify flag register (PR0L, PR0H, PR1L, PR1H)**) are acknowledge-disabled. Actual request acknowledgment is controlled with the interrupt enable flag (IE).

(f) Carry flag (CY)

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

(3) Stack pointer (SP)

This is a 16-bit register that holds the start address of the memory stack area. Only the internal high-speed RAM area (FB00H to FEFFH) can be used as the stack area.

Figure 3-9. Stack Pointer Format

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

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

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

Caution Since $\overline{\text{RESET}}$ input makes the SP contents undefined, be sure to initialize the SP before instruction execution.

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

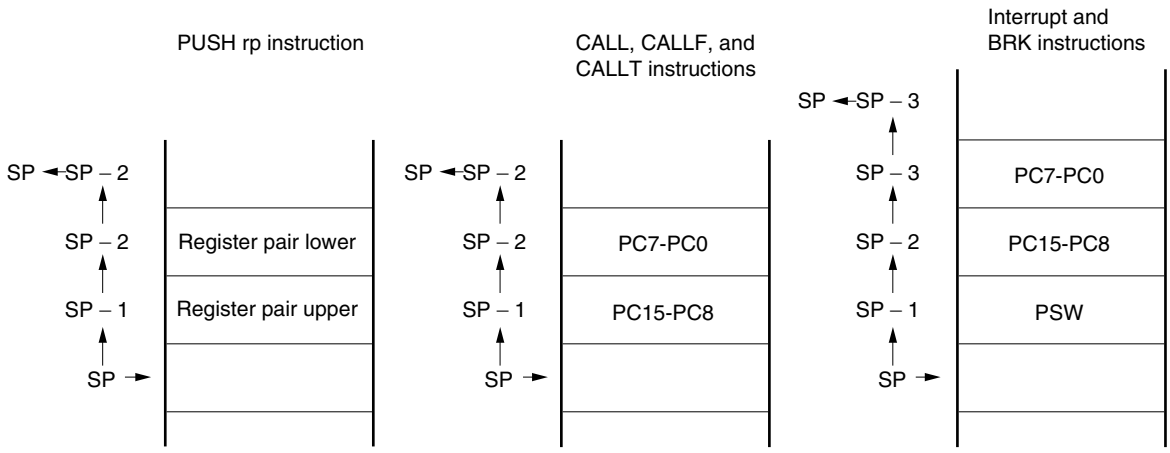
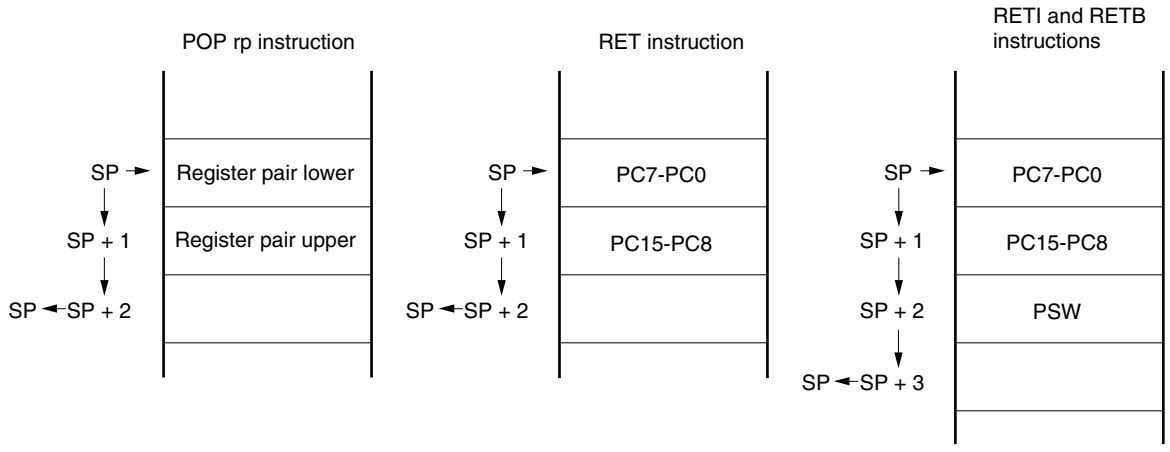


Figure 3-11. Data to Be Restored from Stack Memory



3.2.2 General registers

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

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

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

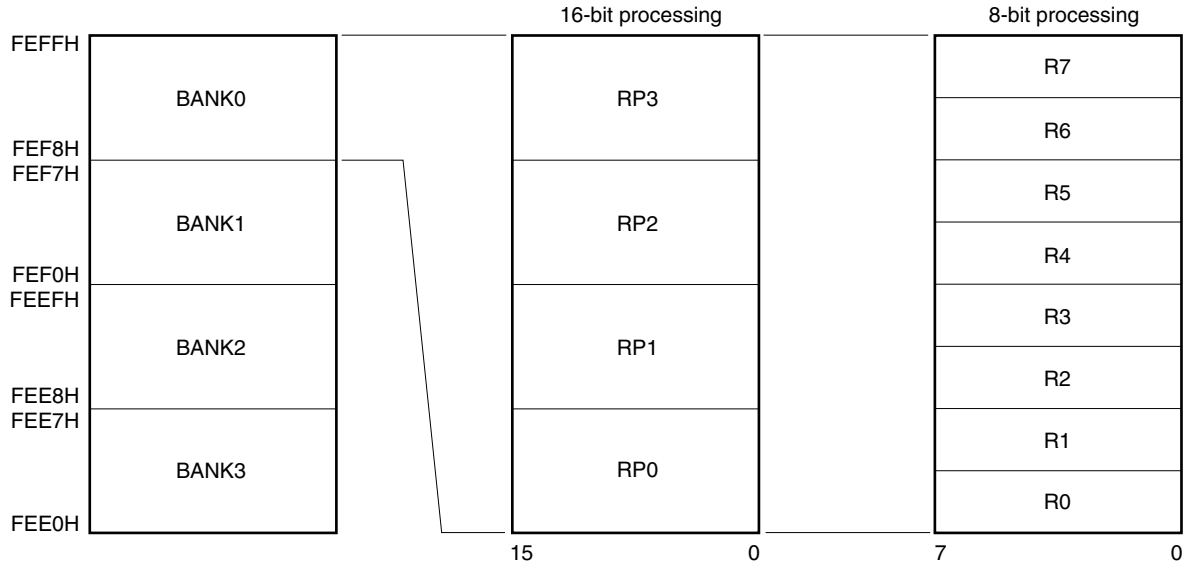
Register banks to be used for instruction execution are set with the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interrupts for each bank.

Table 3-3. Absolute Addresses of General Registers

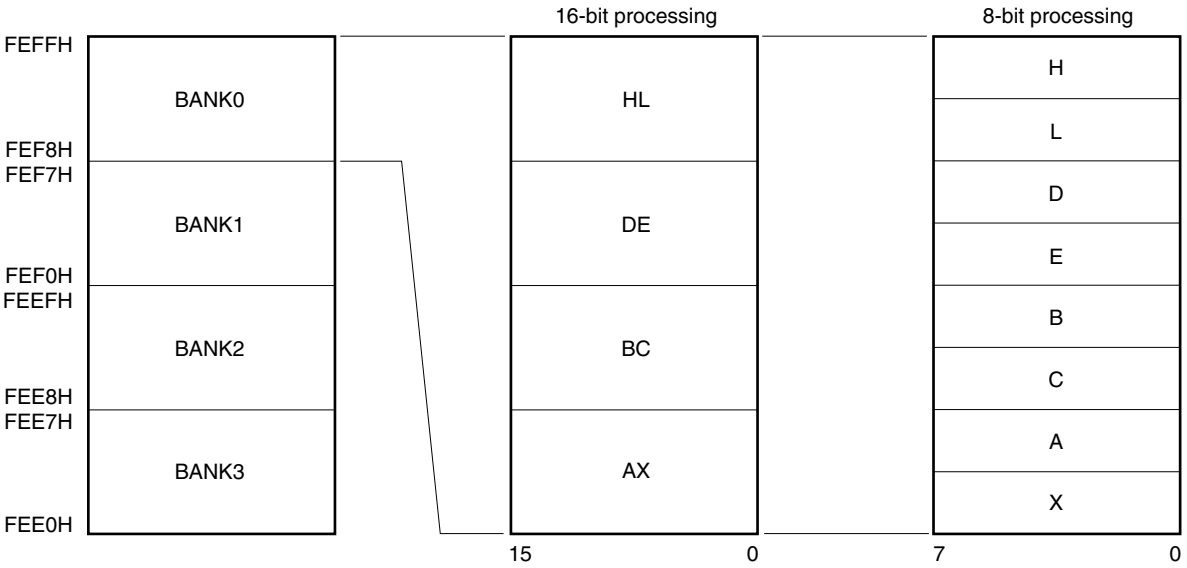
Bank Name	Register		Absolute Address		Bank Name	Register		Absolute Address
	Function Name	Absolute Name				Function Name	Absolute Name	
BANK0	H	R7	F E F F H		BANK2	H	R7	F E E F H
	L	R6	F E F E H			L	R6	F E E E H
	D	R5	F E F D H			D	R5	F E E D H
	E	R4	F E F C H			E	R4	F E E C H
	B	R3	F E F B H			B	R3	F E E B H
	C	R2	F E F A H			C	R2	F E E A H
	A	R1	F E F 9 H			A	R1	F E E 9 H
	X	R0	F E F 8 H			X	R0	F E E 8 H
BANK1	H	R7	F E F 7 H		BANK3	H	R7	F E E 7 H
	L	R6	F E F 6 H			L	R6	F E E 6 H
	D	R5	F E F 5 H			D	R5	F E E 5 H
	E	R4	F E F 4 H			E	R4	F E E 4 H
	B	R3	F E F 3 H			B	R3	F E E 3 H
	C	R2	F E F 2 H			C	R2	F E E 2 H
	A	R1	F E F 1 H			A	R1	F E E 1 H
	X	R0	F E F 0 H			X	R0	F E E 0 H

Figure 3-12. General Register Configuration

(a) Absolute name



(b) Function name



3.2.3 Special function register (SFR)

Unlike general registers, special-function registers have their own functions and are allocated to the FF00H to FFFFH area.

Special-function registers can be manipulated like general registers, with operation, transfer, and bit manipulation instructions. The bit units in which one register can be manipulated (1, 8, or 16 bits) depend on the special-function register type.

Each bit unit for manipulation is specified as follows.

- 1-bit manipulation
  A symbol reserved by the assembler is described as the operand (sfr.bit) of a 1-bit manipulation instruction. This manipulation can also be specified with an address.
- 8-bit manipulation
  A symbol reserved by the assembler is described as the operand (sfr) of an 8-bit manipulation instruction. This manipulation can also be specified with an address.
- 16-bit manipulation
  A symbol reserved by the assembler is described as the operand (sfrp) of a 16-bit manipulation instruction. When specifying an address, describe an even address.

Table 3-4 gives a list of special-function registers. The meanings of items in the table are as follows.

- Symbol
  The symbol indicates the address of a special function register. It is a reserved word in the RA78K/0, and is defined via the header file “sfrbit.h” in the CC78K/0. When using the RA78K/0, ID78K0-NS, ID78K0, or SM78K0, symbols can be written as instruction operands.
- R/W
  Indicates whether the corresponding special-function register can be read or written.
  R/W: Read/write enable
  R: Read only
  W: Write only
- Bit Unit for Manipulation
  “√” indicates the manipulatable bit unit (1, 8, or 16). “—” indicates a bit unit for which manipulation is not possible.
- After Reset
  Indicates each register status upon $\overline{\text{RESET}}$ input.

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

	Address	Special Function Register (SFR) Name	Symbol	R/W	Bit Units for Manipulation			After Reset
					1 bit	8 bits	16 bits	
	FF00H	Port 0	P0	R/W	√	√	—	00H
	FF02H	Port 2	P2		√	√	—	
	FF03H	Port 3	P3		√	√	—	
	FF04H	Port 4	P4		√	√	—	Undefined
	FF05H	Port 5	P5		√	√	—	
	FF06H	Port 6	P6		√	√	—	
	FF07H	Port 7	P7		√	√	—	00H
	FF08H	Port 8	P8		√	√	—	
	FF09H	Port 9	P9		√	√	—	
★	FF0AH	16-bit timer capture/compare register 000	CR000		—	—	√	0000H
	FF0BH				—	—		
★	FF0CH	16-bit timer capture/compare register 010	CR010		—	—	√	
	FF0DH				—	—		
	FF0EH	16-bit timer/counter 00	TM00	R	—	—	√	
	FF0FH				—	—		
★	FF10H	16-bit timer capture/compare register 001	CR001	R/W	—	—	√	
	FF11H				—	—		
★	FF12H	16-bit timer capture/compare register 011	CR011		—	—	√	
	FF13H				—	—		
	FF14H	16-bit timer/counter 01	TM01	R	—	—	√	
	FF15H				—	—		
	FF17H	A/D conversion result register 3	ADCR3		—	√	—	Undefined
	FF18H	Transmit shift register 0	TXS0	W	—	√	—	FFH
		Receive buffer register 0	RXB0	R	—	√	—	
★	FF1AH	Serial I/O shift register 30	SIO30	R/W	—	√	—	00H
★	FF1BH	Serial I/O shift register 31	SIO31		—	√	—	
	FF1FH	IIC0 shift register	IIC0		—	√	—	
	FF20H	Port mode register 0	PM0		√	√	—	FFH
	FF22H	Port mode register 2	PM2		√	√	—	
	FF23H	Port mode register 3	PM3		√	√	—	
	FF24H	Port mode register 4	PM4		√	√	—	
	FF25H	Port mode register 5	PM5		√	√	—	
	FF26H	Port mode register 6	PM6		√	√	—	
	FF27H	Port mode register 7	PM7		√	√	—	
	FF28H	Port mode register 8	PM8		√	√	—	
	FF29H	Port mode register 9	PM9		√	√	—	
	FF30H	Pull-up resistor option register 0	PU0		√	√	—	00H
	FF32H	Pull-up resistor option register 2	PU2		√	√	—	
	FF33H	Pull-up resistor option register 3	PU3		√	√	—	
	FF34H	Pull-up resistor option register 4	PU4		√	√	—	
	FF35H	Pull-up resistor option register 5	PU5		√	√	—	

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

Address	Special Function Register (SFR) Name	Symbol		R/W	Bit Units for Manipulation			After Reset
					1 bit	8 bits	16 bits	
FF36H	Pull-up resistor option register 6	PU6		R/W	√	√	—	00H
FF37H	Pull-up resistor option register 7	PU7			√	√	—	
FF40H	Clock output selection register	CKS			√	√	—	
FF41H	Watch timer mode register 0	WTNM0			√	√	—	
FF42H	Watchdog timer clock selection register	WDCS			—	√	—	
FF47H	Memory expansion mode register	MEM			—	√	—	
FF48H	External interrupt rising edge enable register	EGP			√	√	—	
FF49H	External interrupt falling edge enable register	EGN			√	√	—	
FF60H	16-bit timer mode control register 00	TMC00			√	√	—	
FF61H	Prescaler mode register 00	PRM00			—	√	—	
FF62H	Capture/compare control register 00	CRC00			√	√	—	
FF63H	16-bit timer output control register 00	TOC00			√	√	—	
FF68H	16-bit timer mode control register 01	TMC01			√	√	—	
FF69H	Prescaler mode register 01	PRM01			—	√	—	
FF6AH	Capture/compare control register 01	CRC01			√	√	—	
FF6BH	16-bit timer output control register 01	TOC01			√	√	—	
FF70H	8-bit timer compare register 50	CR5	CR50		—	√	√	Undefined
FF71H	8-bit timer compare register 51		CR51		—	√		
FF72H	8-bit timer compare register 52	CR52			—	√	—	
FF74H	8-bit timer/counter 50	TM5	TM50	R	—	√	√	00H
FF75H	8-bit timer/counter 51		TM51		—	√		
FF76H	8-bit timer/counter 52	TM52			—	√	—	
FF78H	Timer clock selection register 50	TCL50		R/W	—	√	—	
FF79H	8-bit timer mode control register 50	TMC50			√	√	—	04H**Note**
FF7AH	Timer clock selection register 51	TCL51			—	√	—	00H
FF7BH	8-bit timer mode control register 51	TMC51			√	√	—	04H**Note**
FF7CH	Timer clock selection register 52	TCL52			—	√	—	00H
FF7DH	8-bit timer mode control register 52	TMC52			√	√	—	04H**Note**
FF80H	A/D converter mode register 3	ADM3			√	√	—	00H
FF81H	Analog input channel specification register 3	ADS3			—	√	—	
FF83H	Power-fail comparison threshold register 3	PFT3			—	√	—	
FF84H	Power-fail comparison mode register 3	PFM3			√	√	—	
FFA0H	Asynchronous serial interface mode register 0	ASIM0			√	√	—	
FFA1H	Asynchronous serial interface status register 0	ASIS0		R	√	√	—	
FFA2H	Baud rate generator control register 0	BRGC0		R/W	—	√	—	
FFA8H	IIC0 control register	IICC0			√	√	—	
FFA9H	IIC0 status register	IICS0		R	√	√	—	

Note Although the initial value is 04H, the value is shown as 00H when read (because bits 2 and 3 are write only).

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

Address	Special Function Register (SFR) Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFAAH	IIC0 transfer clock selection register	IICCL0	R/W	√	√	—	00H
FFABH	Slave address register 0	SVA0		—	√	—	
FFAFH	IEBus control register**Note 1**	BCR0		√	√	—	
FFB0H	Serial operation mode register 30	CSIM30		√	√	—	
FFB1H	IEBus control data register**Note 1**	CDR		—	√	—	01H
FFB2H	IEBus unit address register**Note 1**	UAR		—	—	√	0000H
FFB3H				—	—		
FFB4H	IEBus slave address register**Note 1**	SAR		—	—	√	
FFB5H				—	—		
FFB6H	IEBus partner address register**Note 1**	PAR	R	—	—	√	
FFB7H				—	—		
FFB8H	Serial operation mode register 31	CSIM31	R/W	√	√	—	00H
FFB9H	IEBus telegraph length register**Note 1**	DLR		—	√	—	01H
FFBAH	IEBus data register**Note 1**	DR		—	√	—	00H
FFBBH	IEBus unit status register**Note 1**	USR	R	√	√	—	
FFBCH	IEBus interrupt status register**Note 1**	ISR	R/W	√	√	—	
FFBDH	IEBus slave status register**Note 1**	SSR	R	√	√	—	41H
FFBEH	IEBus successful communication counter**Note 1**	SCR		—	√	—	01H
FFBFH	IEBus transmission counter**Note 1**	CCR		—	√	—	20H
FFC0H	CAN control register**Note 2**	CANC	R/W	√	√	—	01H
FFC1H	Transmit control register**Note 2**	TCR		—	√	—	00H
FFC2H	Receive message register**Note 2**	RMES	R	—	√	—	
FFC3H	Redefinition control register**Note 2**	REDEF	R/W	√	√	—	
FFC4H	CAN error status register**Note 2**	CANES		—	√	—	
FFC5H	Transmit error counter**Note 2**	TEC	R	—	√	—	
FFC6H	Receive error counter**Note 2**	REC		—	√	—	
FFC7H	Message count register**Note 2**	MCNT		—	√	—	C0H
FFC8H	Bit rate prescaler**Note 2**	BRPRS	R/W	—	√	—	00H
FFC9H	Synchronous control register 0**Note 2**	SYNC0		—	√	—	18H
FFCAH	Synchronous control register 1**Note 2**	SYNC1		—	√	—	0EH
FFCBH	Mask control register**Note 2**	MASKC		—	√	—	00H
FFCDH	Flash programming mode control register**Note 3**	FLPMC		√	√	—	08H
FFD0H to FFDFH	External access area**Note 4**			√	√	—	Undefined

Notes 1. Applies to the μPD78F0701Y and 780702Y only.

2. Applies to the μPD78F0701Y and 780701Y only.

3. Applies to the μPD78F0701Y only.

4. The external access area cannot be accessed by SFR addressing. Access it with an instruction that specifies a 16-bit address.

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

Address	Special Function Register (SFR) Name	Symbol		R/W	Bit Units for Manipulation			After Reset
					1 bit	8 bits	16 bits	
FFE0H	Interrupt request flag register 0L	IF0	IF0L	R/W	√	√	√	00H
FFE1H	Interrupt request flag register 0H		IF0H		√	√		
FFE2H	Interrupt request flag register 1L	IF1	IF1L		√	√	√	
FFE3H	Interrupt request flag register 1H		IF1H		√	√		
FFE4H	Interrupt mask flag register 0L	MK0	MK0L		√	√	√	FFH
FFE5H	Interrupt mask flag register 0H		MK0H		√	√		
FFE6H	Interrupt mask flag register 1L	MK1	MK1L		√	√	√	
FFE7H	Internal mask flag register 1H		MK1H		√	√		DFH
FFE8H	Priority level specification flag register 0L	PR0	PR0L		√	√	√	FFH
FFE9H	Priority level specification flag register 0H		PR0H		√	√		
FFEAH	Priority level specification flag register 1L	PR1	PR1L		√	√	√	
FFEBH	Priority level specification flag register 1H		PR1H		√	√		
FFF0H	Internal memory size switching register	IMS[Note]			—	√	—	CFH
FFF4H	Internal expansion RAM size select register	IXS[Note]			√	√	—	0CH
FFF9H	Watchdog timer mode register	WDTM			√	√	—	00H
FFFAH	Oscillation stabilization time selection register	OSTS			—	√	—	04H
FFFBH	Processor clock control register	PCC			√	√	—	

Note Do not set a value other than the initial value.

3.3 Instruction Address Addressing

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

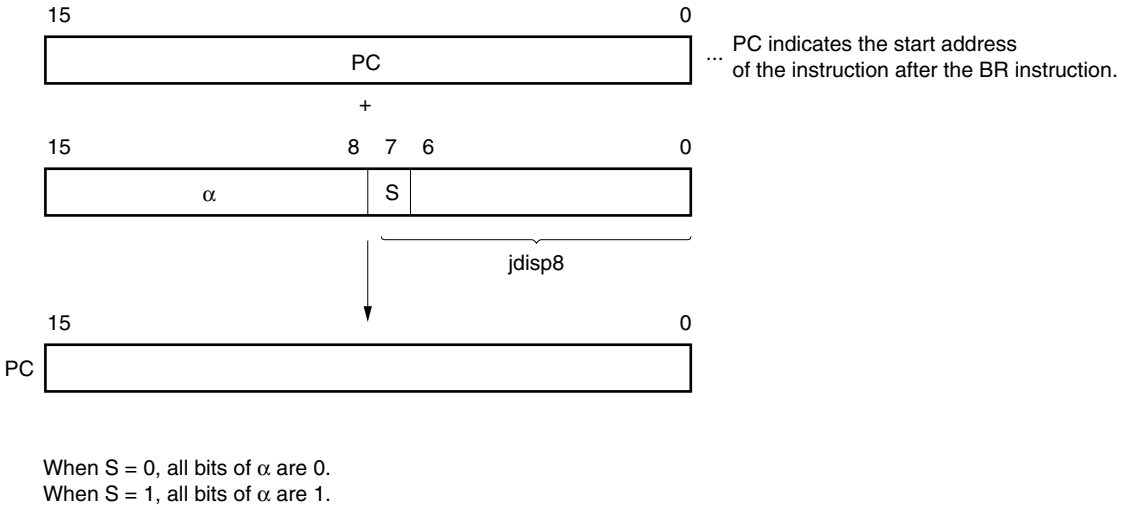
3.3.1 Relative addressing

[Function]

The value obtained by adding 8-bit immediate data (displacement value: jdisp8) of an instruction code to the start address of the following instruction is transferred to the program counter (PC) and branched. The displacement value is treated as signed two's complement data (–128 to +127) and bit 7 becomes a sign bit. In other words, relative addressing consists of relative branching from the start address of the following instruction to the –128 to +127 range.

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

[Illustration]



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

3.3.2 Immediate addressing

[Function]

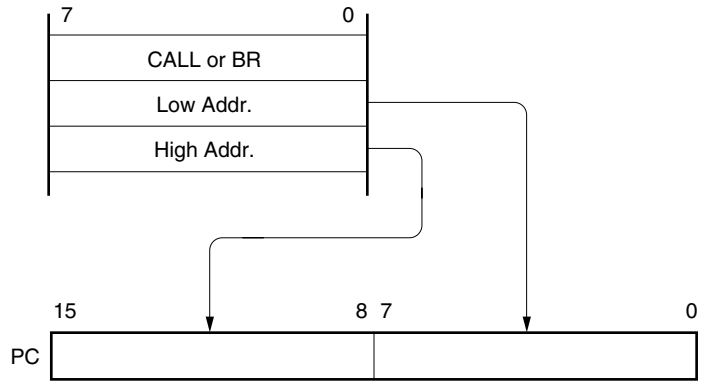
Immediate data in the instruction word is transferred to the program counter (PC) and branched.

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

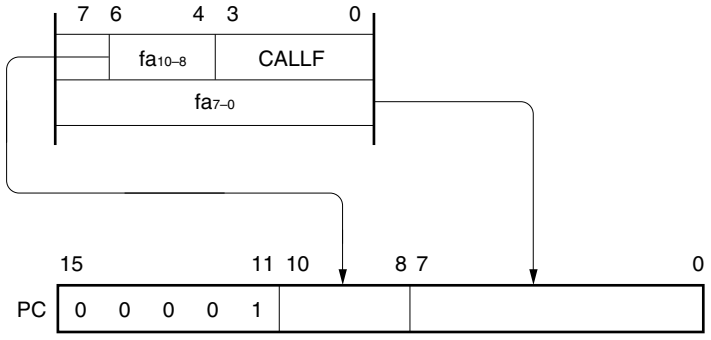
The entire memory space can be used as the destination when the program is branched by executing the CALL !addr16 or BR !addr16 instruction. However, when the CALLF !addr11 instruction is executed, only the area of 0800H to 0FFFH can be used as the program branch destination.

[Illustration]

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



In the case of CALLF !addr11 instruction



3.3.3 Table indirect addressing

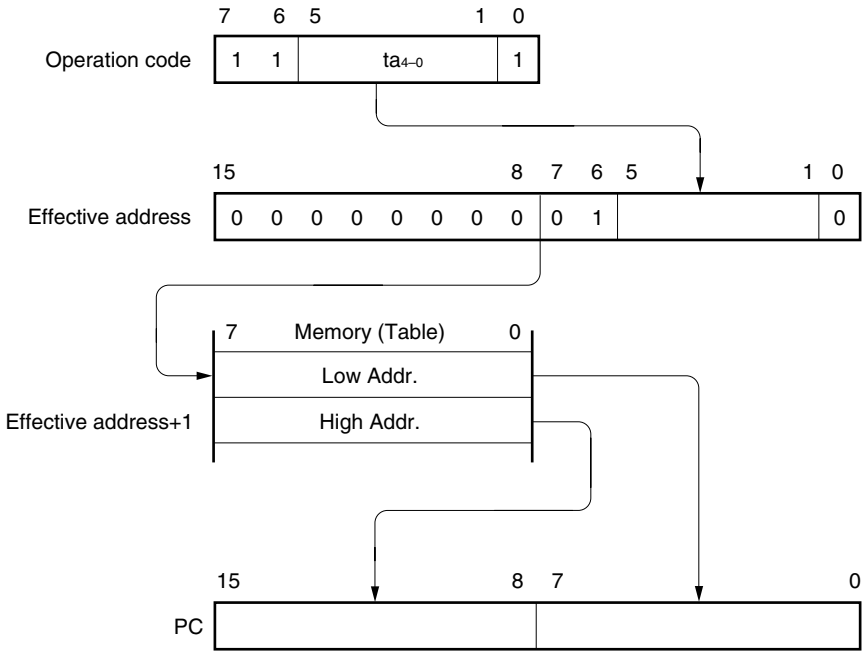
[Function]

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

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

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

[Illustration]



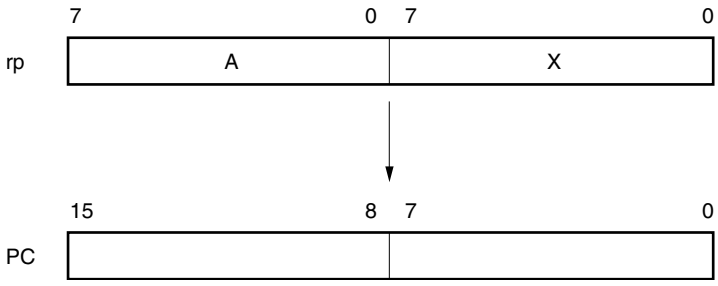
3.3.4 Register addressing

[Function]

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

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

[Illustration]



3.4 Operand Address Addressing

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

3.4.1 Implied addressing

[Function]

This addressing is to automatically (implicitly) address a register that functions as an accumulator (A or AX) in the general register area.

Of the μPD780701Y Subseries instruction words, the following instructions employ implied addressing.

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

[Operand format]

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

[Description example]

MULU X

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

3.4.2 Register addressing

[Function]

This addressing is used to access a general register.

The general register to be accessed is specified with the register bank select flags (RBS0 and RBS1) and with the register specification code (Rn and RPn) in an instruction code.

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

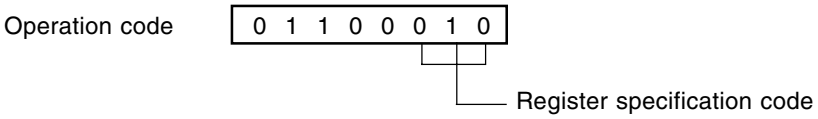
[Operand format]

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

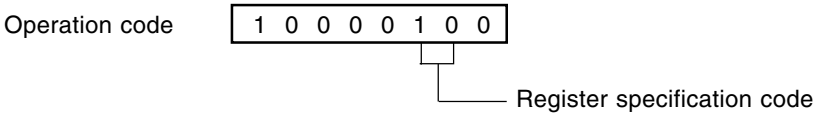
'r' and 'rp' can be described with absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

[Description example]

MOV A, C; when selecting C register as r



INCW DE; when selecting DE register pair as rp



3.4.3 Direct addressing

[Function]

This address is to directly address the memory indicated by the immediate data in an instruction word.

[Operand format]

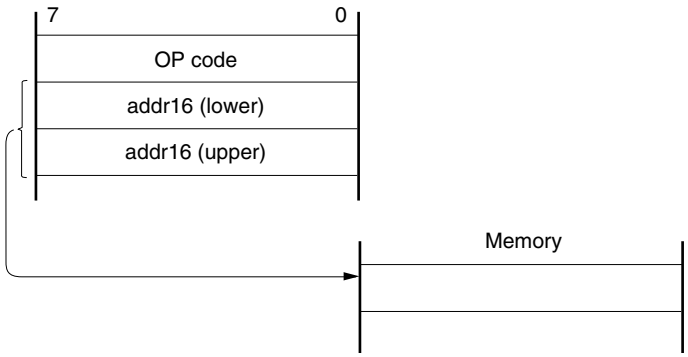
Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

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

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

[Illustration]



3.4.4 Short direct addressing

[Function]

The memory to be manipulated in the fixed space is directly addressed with the 8-bit data in an instruction word. This addressing is applied to the 256-byte space of FE20H to FF1FH. Internal high-speed RAM and special-function registers (SFRs) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

The SFR area (FF00H to FF1FH) where short direct addressing is applied is a part of the entire SFR area. Ports which are frequently accessed in a program and compare and capture registers of timer/event counters are mapped to the SFR area. These SFRs can be manipulated with a small number of bytes and clocks.

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

[Operand format]

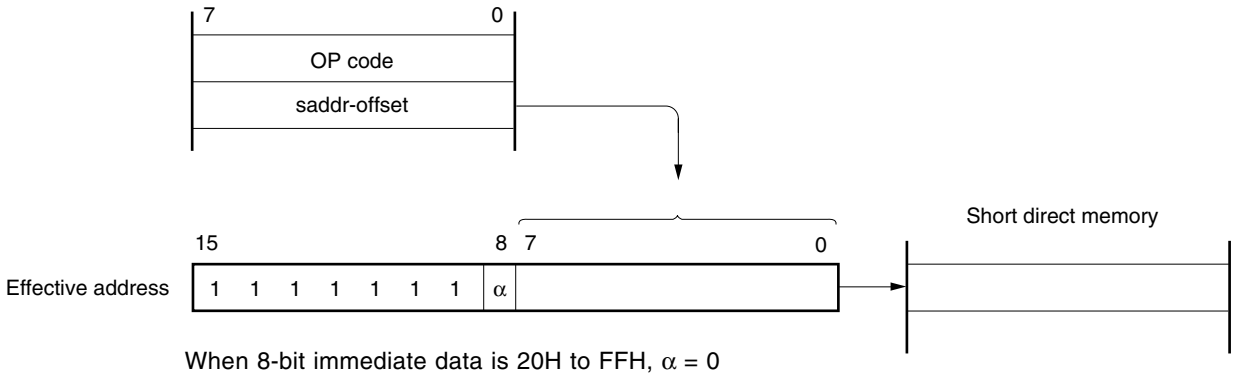
Identifier	Description
saddr	Label or immediate data of FE20H to FF1FH
saddrp	Label or immediate data of FE20H to FF1FH (even address only)

[Description example]

MOV 0FE30H, #50H; when setting saddr to FE30H and immediate data to 50H

Operation code	0 0 0 1 0 0 0 1	OP code
	0 0 1 1 0 0 0 0	30H (saddr-offset)
	0 1 0 1 0 0 0 0	50H (immediate data)

[Illustration]



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

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

3.4.5 Special function register (SFR) addressing

[Function]

The memory-mapped special function registers (SFRs) are addressed with the 8-bit immediate data in an instruction word.

This addressing is applied to the 240-byte space of FF00H to FFCFH plus FFE0H to FFFFH. However, the SFR mapped at FF00H to FF1FH can be accessed with short direct addressing.

[Operand format]

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

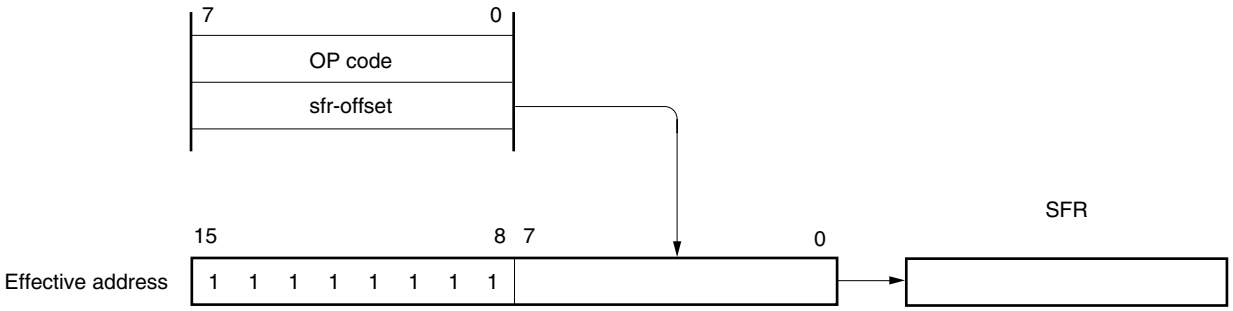
[Description example]

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

Operation code | 1 1 1 1 0 1 1 0 | OP code

| 0 0 1 0 0 0 0 0 | 20H (sfr-offset)

[Illustration]



3.4.6 Register indirect addressing

[Function]

This addressing is used to address memory by using the contents of a specified register pair as an operand. The register pair to be accessed is specified by the register bank select flags (RSB0 and RSB1) and the register pair specification code in an instruction code. This addressing can be carried out for the entire memory space.

[Operand format]

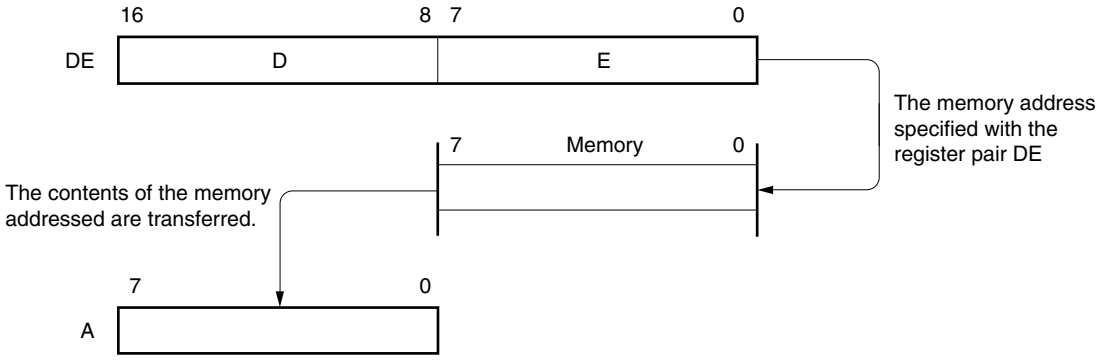
Identifier	Description
—	[DE], [HL]

[Description example]

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

Operation code | 1 0 0 0 0 1 0 1 |

[Illustration]



3.4.7 Based addressing

[Function]

This addressing is to address the memory by using the result of adding 8-bit immediate data to the contents of the HL register pair that is used as a base register. The HL register pair to be accessed is in the register bank specified with the register bank select flags (RBS0 and RBS1). Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be carried out for the entire memory space.

[Operand format]

Identifier	Description
—	[HL + byte]

[Description example]

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

Operation code	1	0	1	0	1	1	1	0
	0	0	0	1	0	0	0	0

3.4.8 Based indexed addressing

[Function]

This addressing is to address the memory by using the result of adding the contents of the B or C register specified in the instruction word to the contents of the HL register that is used as a base register. The HL, B, and C registers accessed are in the register bank specified by the register bank select flags (RBS0 and RBS1). Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be carried out for the entire memory space.

[Operand format]

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

[Description example]

MOV A, [HL + B]

Operation code

1	0	1	0	1	0	1	1

3.4.9 Stack addressing

[Function]

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

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

Stack addressing can access the internal high-speed RAM area only.

[Description example]

PUSH DE

Operation code

1	0	1	1	0	1	0	1

CHAPTER 4 PORT FUNCTIONS

4.1 Port Functions

The μPD780701Y Subseries products incorporate 67 input/output ports. Figure 4-1 shows the port configuration. Every port is capable of 1-bit and 8-bit manipulations and can carry out considerably varied control operations. Besides port functions, the ports can also serve as on-chip hardware input/output pins.

Figure 4-1. Types of Ports

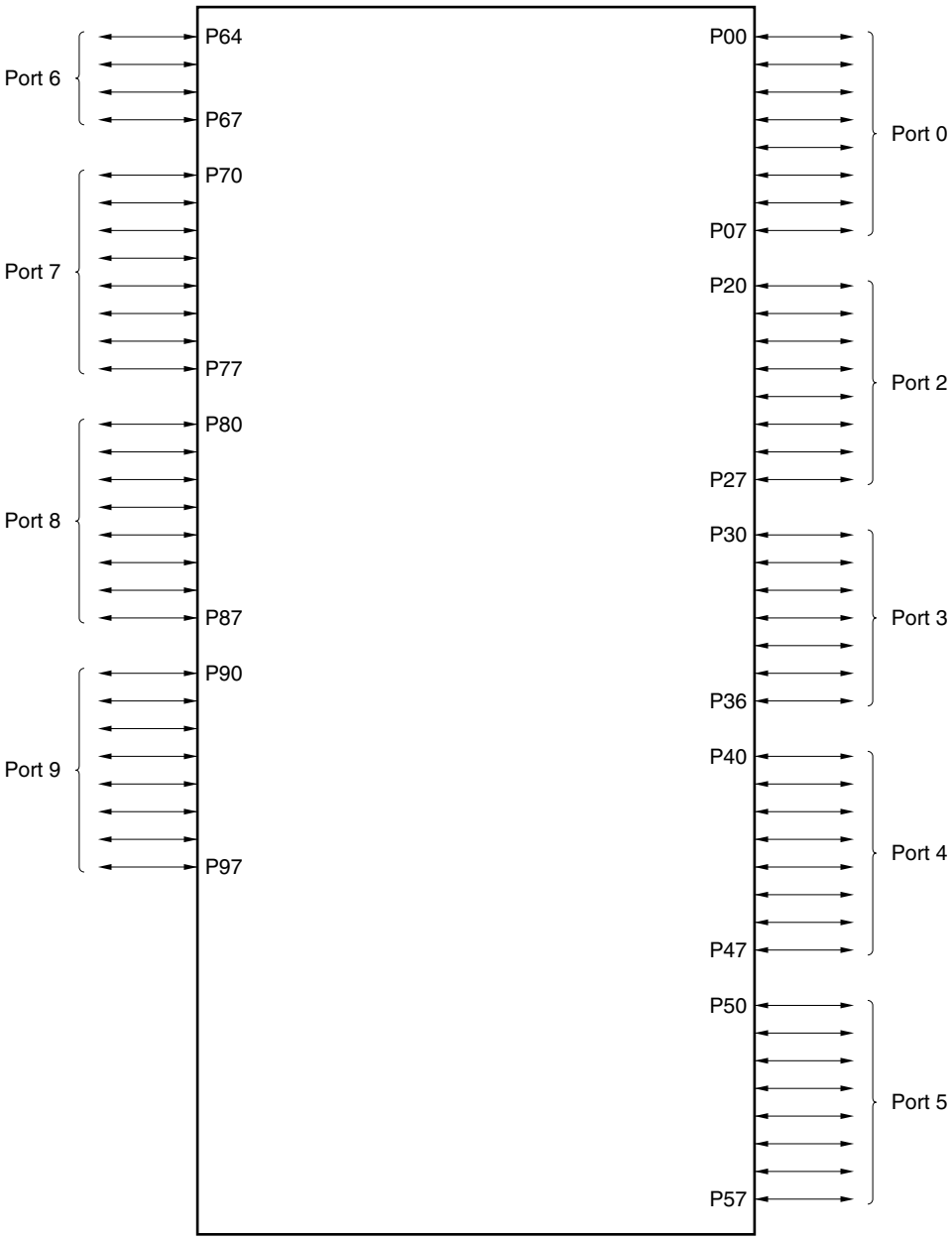


Table 4-1. Port Functions (1/2)

Pin Name	Function		Alternate Function
P00	Port 0 8-bit input/output port Input/output can be specified in 1-bit units An on-chip pull-up resistor can be specified in 1-bit units by means of software		INTP0
P01			INTP1
P02			INTP2
P03			INTP3
P04			INTP4
P05			INTP5
P06			INTP6
P07			INTP7
P20	Port 2 8-bit input/output port Input/output can be specified in 1-bit units An on-chip pull-up resistor can be specified in 1-bit units by means of software		SI31
P21			SO31
P22			$\overline{\text{SCK31}}$
P23			BUZ
P24			RxD0
P25			TxD0
P26			ASCK0
P27			PCL
P30	Port 3 7-bit input/output port Input/output can be specified in 1-bit units	An on-chip pull-up resistor can be specified in 1-bit units by means of software	SI30
P31			SO30
P32			$\overline{\text{SCK30}}$
P33		N-ch open-drain input/output port (15-V breakdown voltage). LEDs can be driven directly.	—
P34		An on-chip pull-up resistor can be specified in 1-bit units by means of software	TO00
P35			TI000
P36			TI010
P40 to P47	Port 4 8-bit input/output port Input/output can be specified in 1-bit units An on-chip pull-up resistor can be specified in 1-bit units by means of software Interrupt request flag KRIF is set to 1 by falling edge detection.		—
P50 to P57	Port 5 8-bit input/output port. TTL level input/CMOS output. Input/output can be specified in 1-bit units An on-chip pull-up resistor can be specified in 1-bit units by means of software		—
P64	Port 6 4-bit input/output port. Input/output can be specified in 1-bit units An on-chip pull-up resistor can be specified in 1-bit units by means of software		—
P65			
P66			
P67			

Table 4-1. Port Functions (2/2)

Pin Name	Function		Alternate Function
P70	Port 7. 8-bit input/output port Input/output can be specified in 1-bit units	An on-chip pull-up resistor can be specified in 1-bit units by means of software	TI52/TO52
P71		N-ch open-drain input/output port (5-V breakdown voltage)	SDA0
P72			SCL0
P73		An on-chip pull-up resistor can be specified in 1-bit units by means of software	TO01
P74			TI001
P75			TI011
P76			TI50/TO50
P77			TI51/TO51
P80 to P87	Port 8 8-bit input/output port Input/output can be specified in 1-bit units		ANI0 to ANI7
P90 to P97	Port 9 8-bit input/output port. Input/output can be specified in 1-bit units		ANI8 to ANI15

4.2 Port Configuration

A port consists of the following hardware:

Table 4-2. Configuration of Port

Item	Configuration
Control register	Port mode register (PMm: m = 0, 2 to 9) Pull-up resistor option register (PUm: m = 0, 2 to 7) Memory expansion mode register (MEM)**Note**
Port	Total: 67 ports
Pull-up resistor	Software control: 48

Note The memory expansion mode register (MEM) is a register that controls the falling edge detection function of port 4.

4.2.1 Port 0

Port 0 is an 8-bit input/output port with output latches. P00 to P07 can be specified as input mode/output mode in 1-bit units with port mode register 0 (PM0). On-chip pull-up resistors can be specified for P00 to P07 in 1-bit units with pull-up resistor option register 0 (PU0).

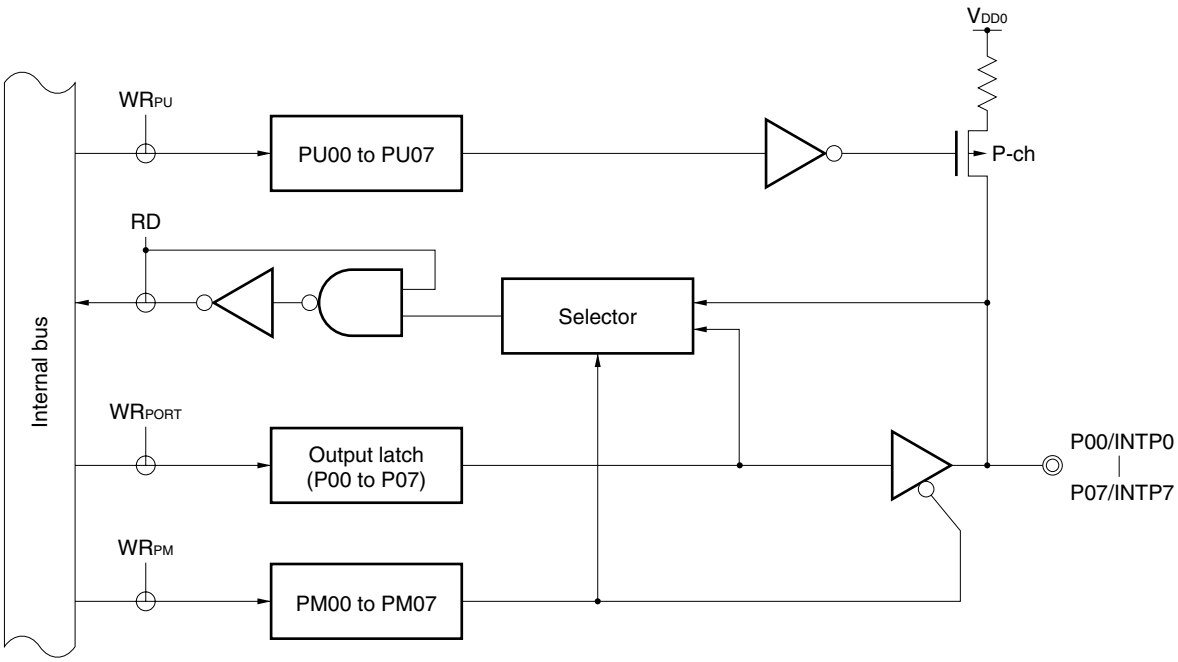
This port can also be used as an external interrupt request input.

$\overline{\text{RESET}}$ input sets port 0 to input mode.

Figure 4-2 shows a block diagram of port 0.

Caution Because port 0 also serves as an external interrupt request input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. Therefore, when the output mode is used, set the interrupt mask flag to 1.

★ **Figure 4-2. Block Diagram of P00 to P07**



PU: Pull-up resistor option register
PM: Port mode register
RD: Port 0 read signal
WR: Port 0 write signal

4.2.2 Port 2

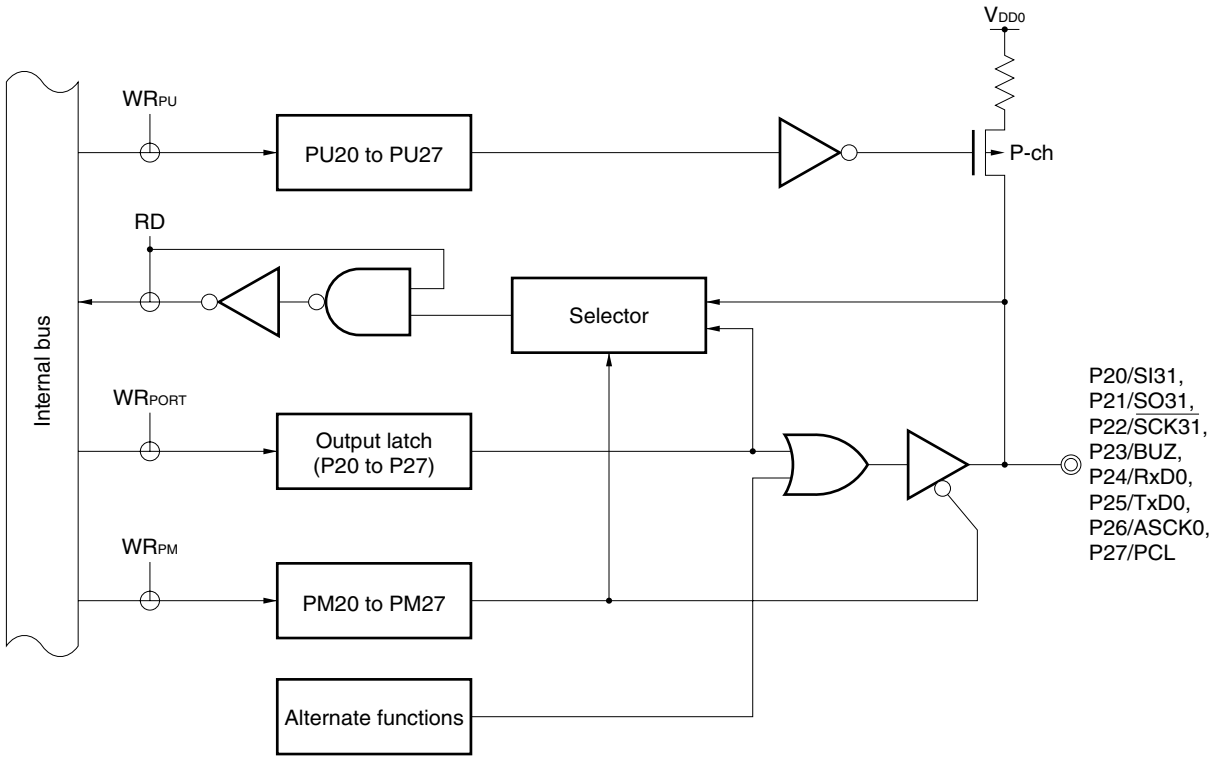
Port 2 is an 8-bit input/output port with output latches. P20 to P27 can be specified as input mode/output mode in 1-bit units with port mode register 2 (PM2). On-chip pull-up resistors can be specified for P20 to P27 in 1-bit units with pull-up resistor option register 2 (PU2).

This port can also be used as the serial data input/output and serial clock input/output of the serial interface, buzzer output, and clock output.

$\overline{\text{RESET}}$ input sets port 2 to input mode.

Figure 4-3 shows a block diagram of port 2.

Figure 4-3. Block Diagram of P20 to P27



PU: Pull-up resistor option register
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

4.2.3 Port 3

Port 3 is a 7-bit input/output port with output latches. P33 is an N-ch open-drain input/output port. P30 to P36 can be specified as input mode/output mode in 1-bit units with port mode register 3 (PM3). P33 can directly drive LEDs.

On-chip pull-up resistors can be specified for P30 to 32 and P34 to 36 in 1-bit units with pull-up resistor option register 3 (PU3).

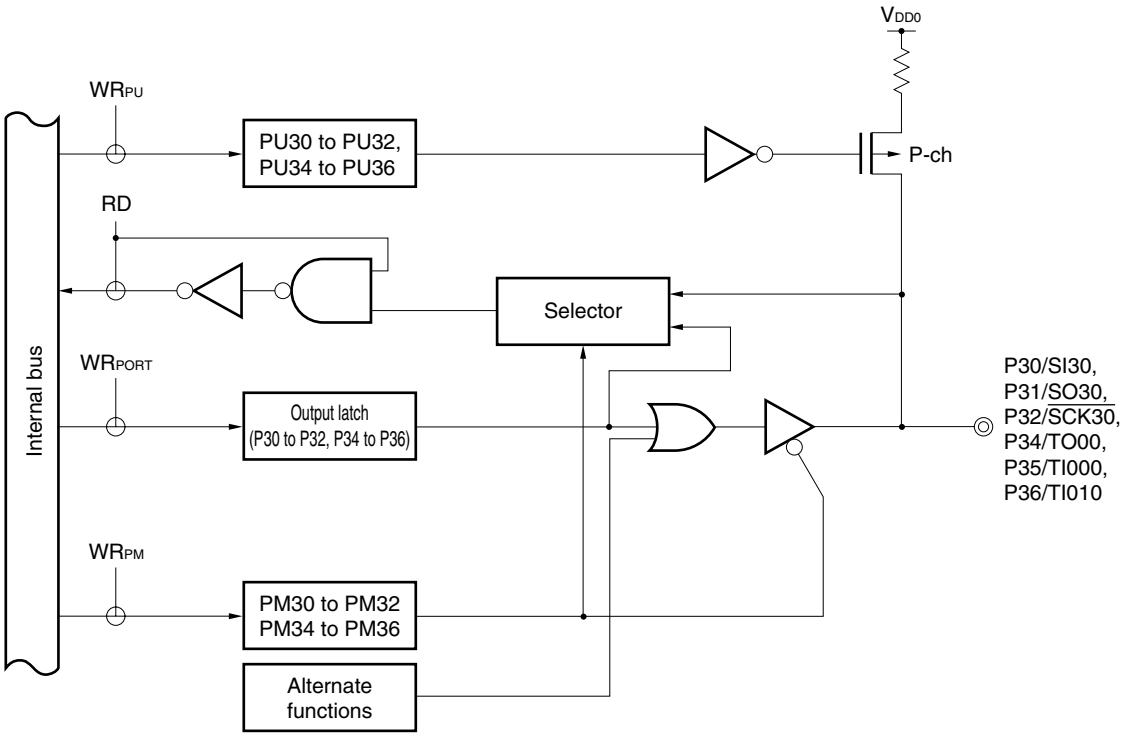
P30 to P32 and P34 to P36 can also be used as the serial data input/output and serial clock input/output of the serial interface, and timer input/output.

$\overline{\text{RESET}}$ input sets port 3 to input mode.

Figures 4-4 and 4-5 show block diagrams of port 3.

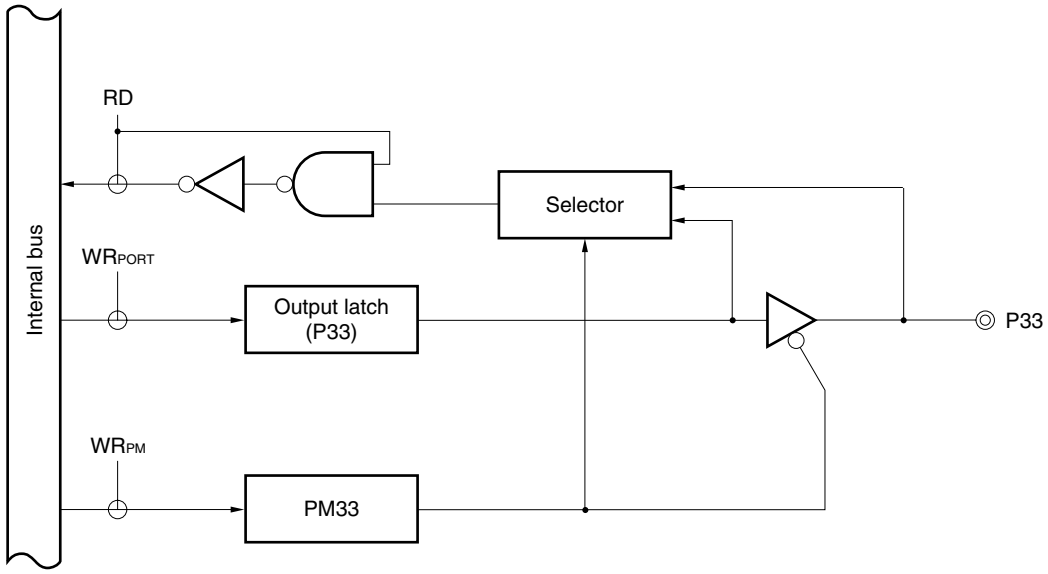
Caution P33 does not include a pull-up resistor.

Figure 4-4. Block Diagram of P30 to P32 and P34 to P36



PU: Pull-up resistor option register
PM: Port mode register
RD: Port 3 read signal
WR: Port 3 write signal

Figure 4-5. Block Diagram of P33



PM: Port mode register
RD: Port 3 read signal
WR: Port 3 write signal

4.2.4 Port 4

Port 4 is an 8-bit input/output port with output latches. P40 to P47 can be specified as input mode/output mode in 1-bit units with port mode register 4 (PM4). On-chip pull-up resistors can be specified for P40 to P47 in 1-bit units with pull-up resistor option register 4 (PU4).

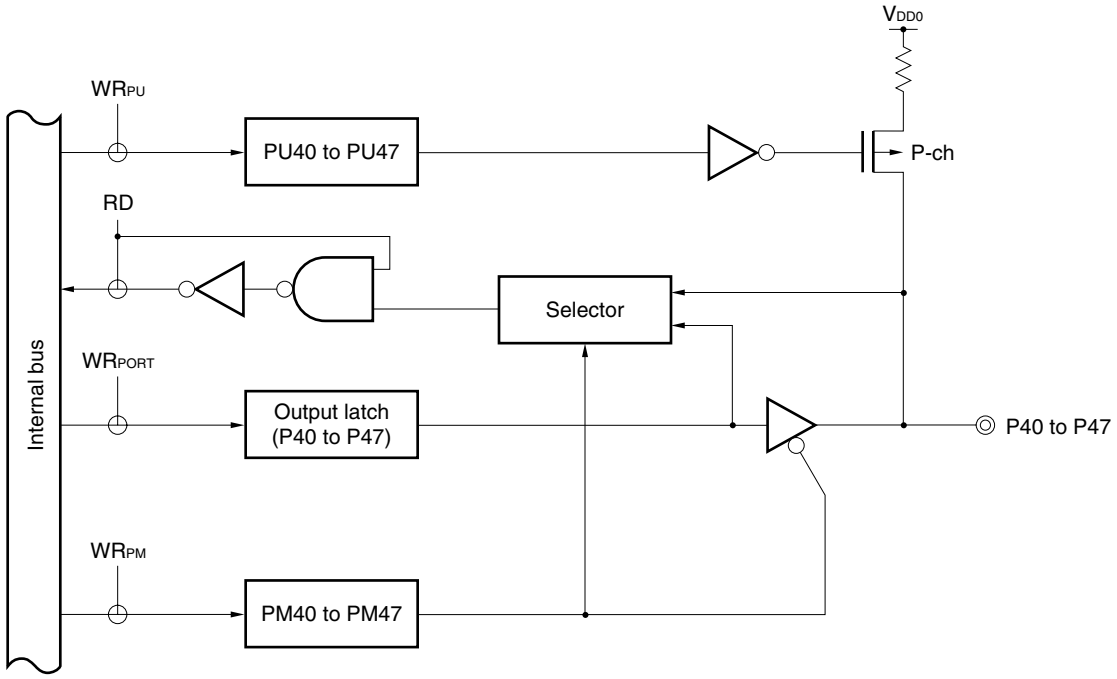
The interrupt request flag (KRIF) can be set to 1 by falling edge detection.

Falling edge detection is controlled by the memory expansion mode register (MEM).

$\overline{\text{RESET}}$ input sets port 4 to input mode.

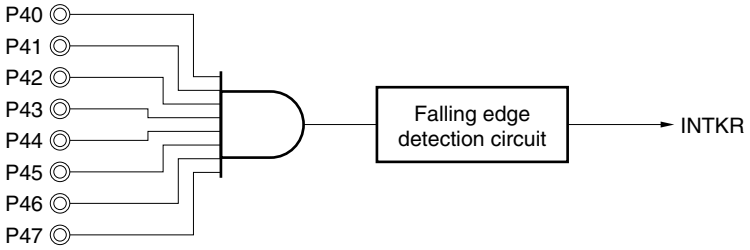
Figure 4-6 shows a block diagram of port 4.

Figure 4-6. Block Diagram of P40 to P47



PU: Pull-up resistor option register
PM: Port mode register
RD: Port 4 read signal
WR: Port 4 write signal

Figure 4-7. Block Diagram of Falling Edge Detection Circuit



4.2.5 Port 5

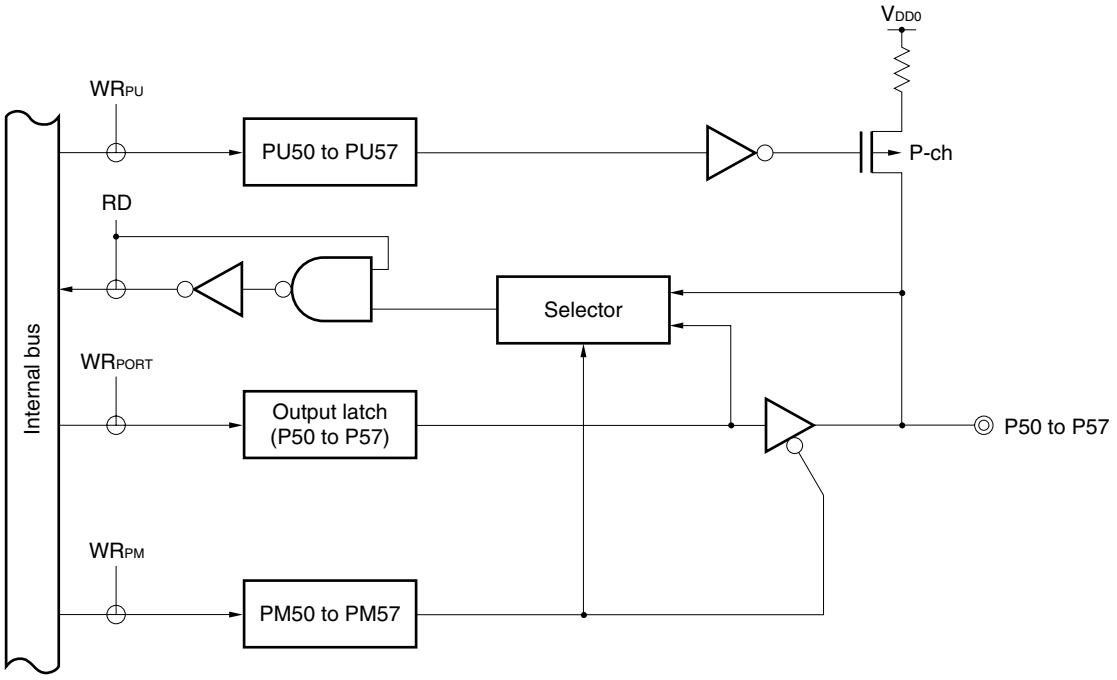
Port 5 is an 8-bit input/output port with output latches. The P50 to P57 can be specified the input mode/output mode in 1-bit units with port mode register 5 (PM5). On-chip pull-up resistors can be specified for P50 to P57 in 1-bit units with pull-up resistor option register 5 (PU5).

Port 5 is a TTL level input.

$\overline{\text{RESET}}$ input sets port 5 to input mode.

Figure 4-8 shows a block diagram of port 5.

Figure 4-8. Block Diagram of P50 to P57



PU: Pull-up resistor option register

PM: Port mode register

RD: Port 5 read signal

WR: Port 5 write signal

4.2.6 Port 6

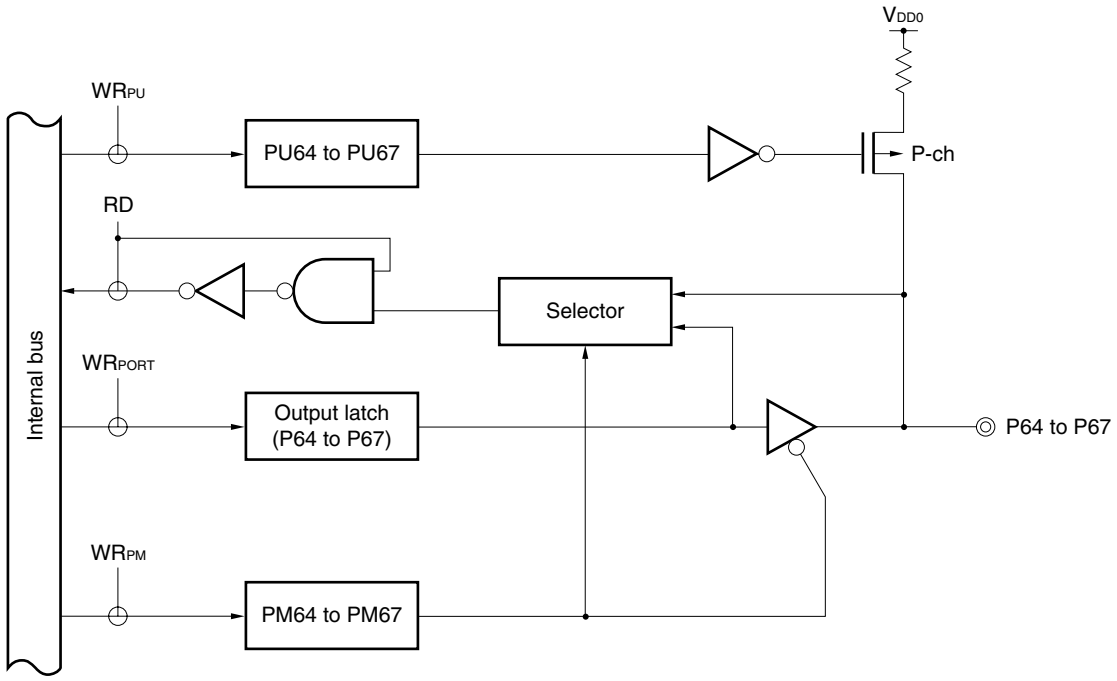
Port 6 is a 4-bit input/output port with output latches. P64 to P67 can be specified as input mode/output mode in 1-bit units with port mode register 6 (PM6).

On-chip pull-up resistors can be specified for P64 to P67 in 1-bit units with pull-up resistor option register 6 (PU6).

$\overline{\text{RESET}}$ input sets port 6 to input mode.

Figure 4-9 shows a block diagram of port 6.

Figure 4-9. Block Diagram of P64 to P67



PU: Pull-up resistor option register
PM: Port mode register
RD: Port 6 read signal
WR: Port 6 write signal

4.2.7 Port 7

This is an 8-bit input/output port with output latches. P71 and P72 are N-ch open-drain input/output ports. P70 to P77 can be specified as input mode/output mode in 1-bit units with port mode register 7 (PM7). On-chip pull-up resistors can be specified for P70, and P73 to P77 in 1-bit units with pull-up resistor option register 7 (PU7).

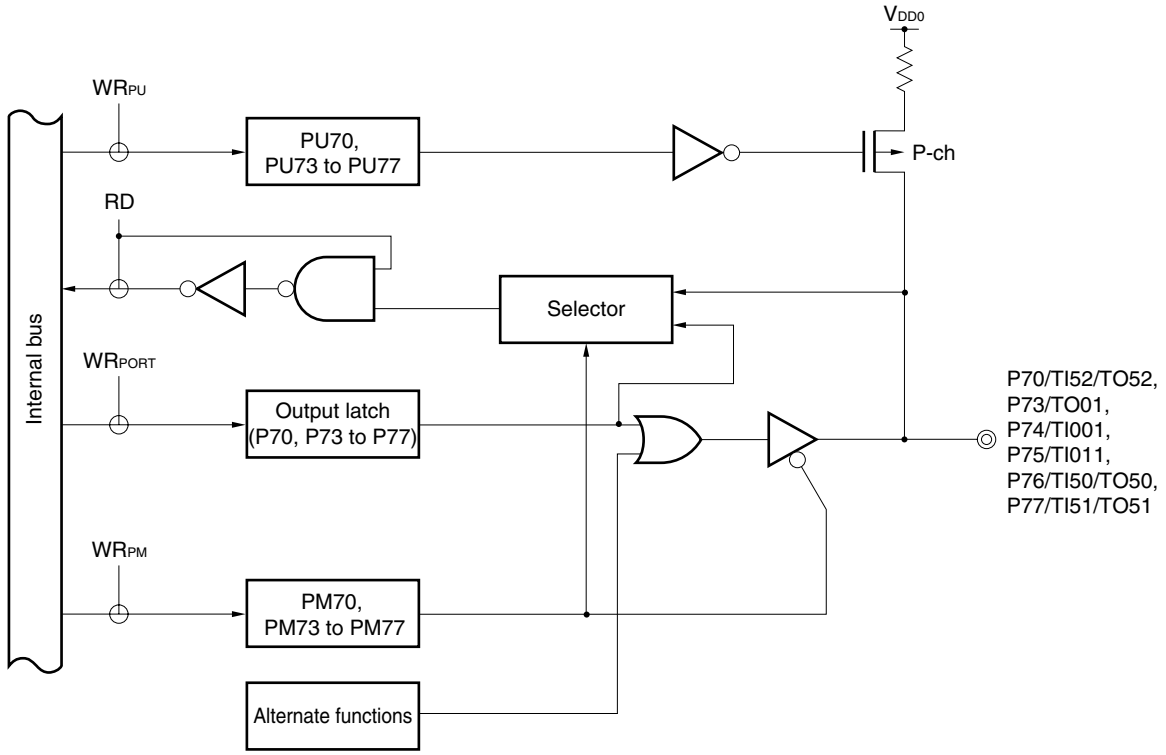
This port can also be used as the serial data input/output and serial clock input/output of the serial interface and timer input/output.

$\overline{\text{RESET}}$ input sets port 7 to input mode.

Figures 4-10 and 4-11 show block diagrams of port 7.

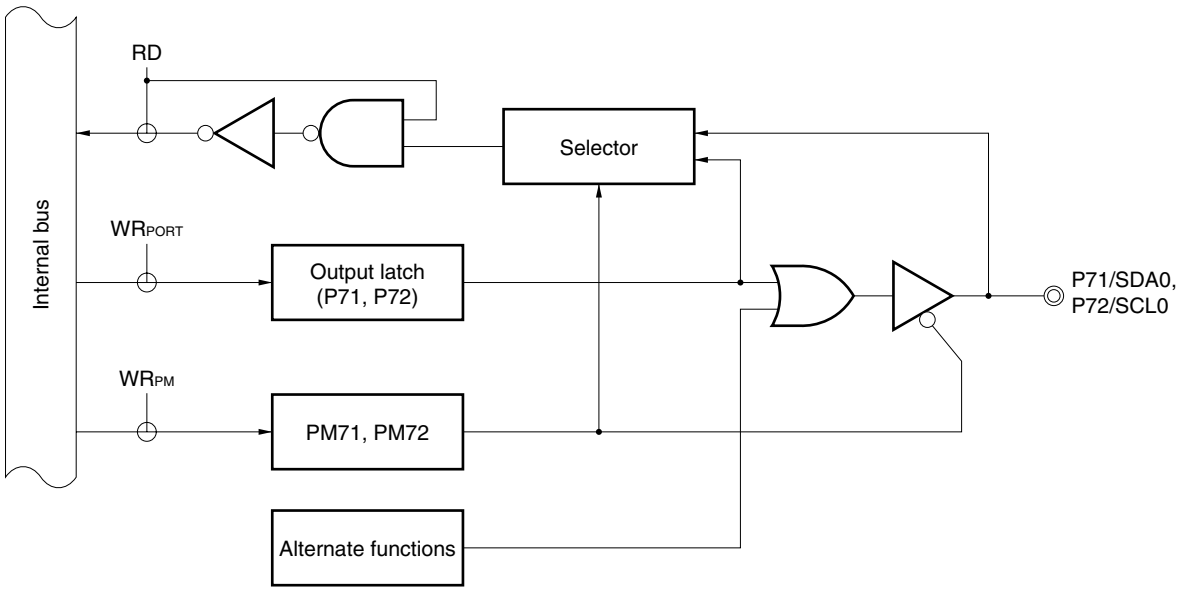
Caution P71 and P72 do not include a pull-up resistor.

Figure 4-10. Block Diagram of P70, P73 to P77



PU: Pull-up resistor option register
PM: Port mode register
RD: Port 7 read signal
WR: Port 7 write signal

Figure 4-11. Block Diagram of P71 and P72



PM: Port mode register
RD: Port 7 read signal
WR: Port 7 write signal

4.2.8 Port 8

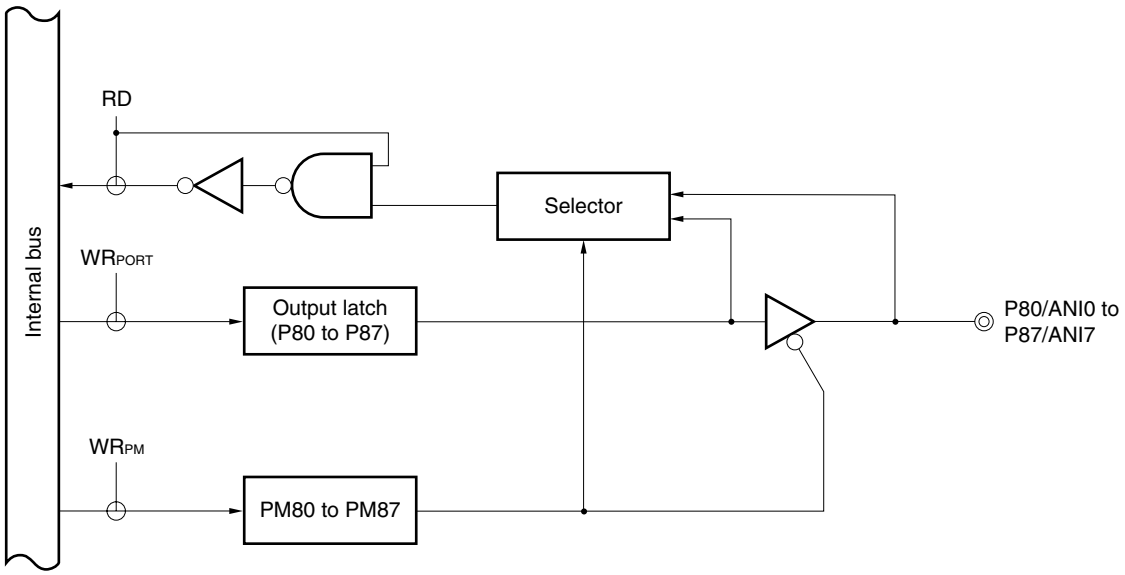
This is an 8-bit input/output port with output latches. P80 to P87 can be specified as input mode/output mode in 1-bit units with port mode register 8 (PM8).

This port can also be used as the A/D converter analog input.

$\overline{\text{RESET}}$ input sets port 8 to input mode.

Figure 4-12 shows a block diagram of port 8.

Figure 4-12. Block Diagram of P80 to P87



PM: Port mode register

RD: Port 8 read signal

WR: Port 8 write signal

4.2.9 Port 9

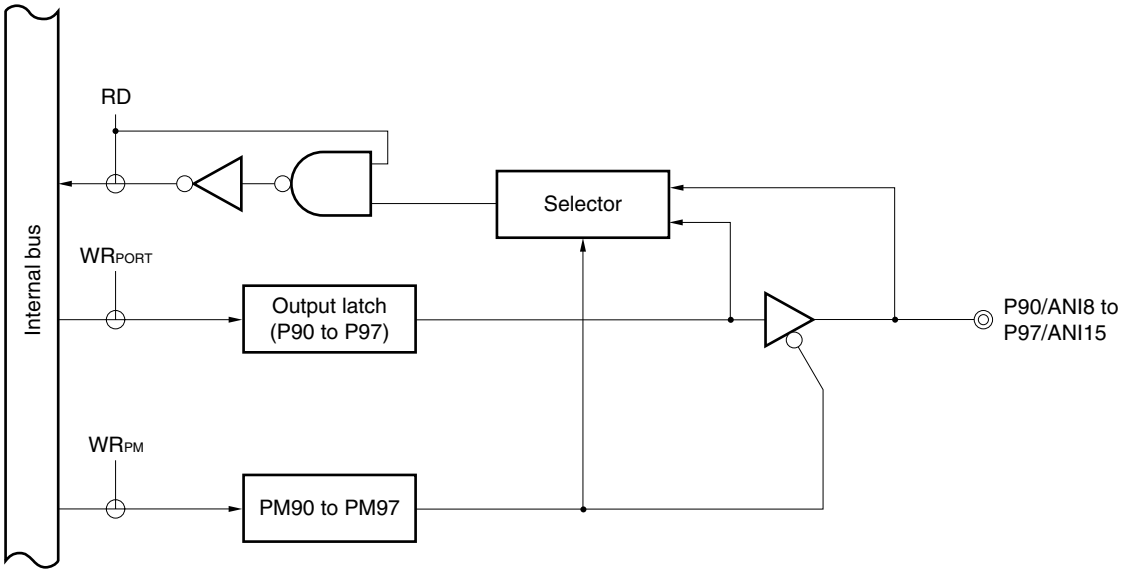
This is an 8-bit input/output port with output latches. P90 to P97 can be specified as input mode/output mode in 1-bit units with port mode register 9 (PM9).

This port is also used as the A/D converter analog input.

$\overline{\text{RESET}}$ input sets port 9 to input mode.

Figure 4-13 shows a block diagram of port 9.

Figure 4-13. Block Diagram of P90 to P97



PM: Port mode register

RD: Port 9 read signal

WR: Port 9 write signal

4.3 Port Function Control Registers

The following three types of registers control the ports.

- Port mode registers (PM0, PM2 to PM9)
- Pull-up resistor option registers (PU0, PU2 to PU7)
- Memory expansion mode register (MEM)

(1) Port mode registers (PM0, PM2 to PM9)

These registers are used to set the port to input/output in 1-bit units.

PM0 and PM2 to PM9 are independently set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the registers to FFH.

Cautions 1. When ports are used as alternate function pins, input/output should be specified by the port mode register (PMn). Therefore, when a pin is specified as an alternate function output, set the PMn and output latches to 0. When it is specified as an alternate function input, set the PMn to 1. (n = 0, 2 to 9)

2. As port 0 has an alternate function as an external interrupt input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. When the output mode is used, therefore, the interrupt mask flag should be set to 1 beforehand.

Figure 4-14. Format of Port Mode Registers (PM0, PM2 to PM9)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM3	1	PM36	PM35	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	FFH	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	FF25H	FFH	R/W
PM6	PM67	PM66	PM65	PM64	1	1	1	1	FF26H	FFH	R/W
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	FF27H	FFH	R/W
PM8	PM87	PM86	PM85	PM84	PM83	PM82	PM81	PM80	FF28H	FFH	R/W
PM9	PM97	PM96	PM95	PM94	PM93	PM92	PM91	PM90	FF29H	FFH	R/W

PMmn	Pmn pin input/output mode select (m = 0, 2 to 9; n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

(2) Pull-up resistor option registers (PU0, PU2 to PU7)

These registers set whether to use on-chip pull-up resistors at each port or not. Setting PU0 and PU2 to PU7 enables the use of on-chip pull-up resistors at the corresponding port pins.

PU0 and PU2 to PU7 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets these registers to 00H.

Cautions 1. P33, P71, P72, P80 to P87, and P90 to P97 do not incorporate pull-up resistors.

2. When using on-chip pull-up resistors, the pull-up resistors can not be disconnected even in the output mode. If used in the output mode, set the corresponding pull-up resistor option register to 0.

Figure 4-15. Format of Pull-Up Resistor Option Registers (PU0, PU2 to PU7)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PU0	PU07	PU06	PU05	PU04	PU03	PU02	PU01	PU00	FF30H	00H	R/W
PU2	PU27	PU26	PU25	PU24	PU23	PU22	PU21	PU20	FF32H	00H	R/W
PU3	0	PU36	PU35	PU34	0	PU32	PU31	PU30	FF33H	00H	R/W
PU4	PU47	PU46	PU45	PU44	PU43	PU42	PU41	PU40	FF34H	00H	R/W
PU5	PU57	PU56	PU55	PU54	PU53	PU52	PU51	PU50	FF35H	00H	R/W
PU6	PU67	PU66	PU65	PU64	0	0	0	0	FF36H	00H	R/W
PU7	PU77	PU76	PU75	PU74	PU73	0	0	PU70	FF37H	00H	R/W

PUmn	Pmn pin on-chip pull-up resistor select (m = 0, 2 to 7; n = 0 to 7)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

(3) Memory expansion mode register (MEM)

The memory expansion mode register (MEM) controls the falling edge detection at port 4.

MEM is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to 00H.

Figure 4-16. Format of Memory Expansion Mode Register (MEM)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MEM	0	0	0	0	0	0	0	MM0	FF47H	00H	R/W

MM0	Falling edge detection control
0	Operation disabled
1	Operation enabled

Cautions 1. Be sure to set bits 1 to 7 to 0.

2. MEM detects the falling edge at any of P40 to P47. In addition, when a low-level input occurs at only one of P40 to P47, the interrupt request signal (INTKR) is not generated even if the falling edge is input to the other pins.

4.4 Port Function Operations

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

4.4.1 Writing to input/output port

(1) Output mode

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

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

(2) Input mode

A value is written to the output latch by a transfer instruction, but since the output buffer is OFF, the pin status does not change.

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

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

4.4.2 Reading from input/output port

(1) Output mode

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

(2) Input mode

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

4.4.3 Operations on input/output port

(1) Output mode

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

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

(2) Input mode

The output latch contents are undefined, but since the output buffer is OFF, the pin status does not change.

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

CHAPTER 5 CLOCK GENERATOR

5.1 Clock Generator Functions

The clock generator generates the clock to be supplied to the CPU and peripheral hardware. The system clock oscillator oscillates at a frequency of 1.0 to 6.29 MHz. Oscillation can be stopped by executing the STOP instruction.

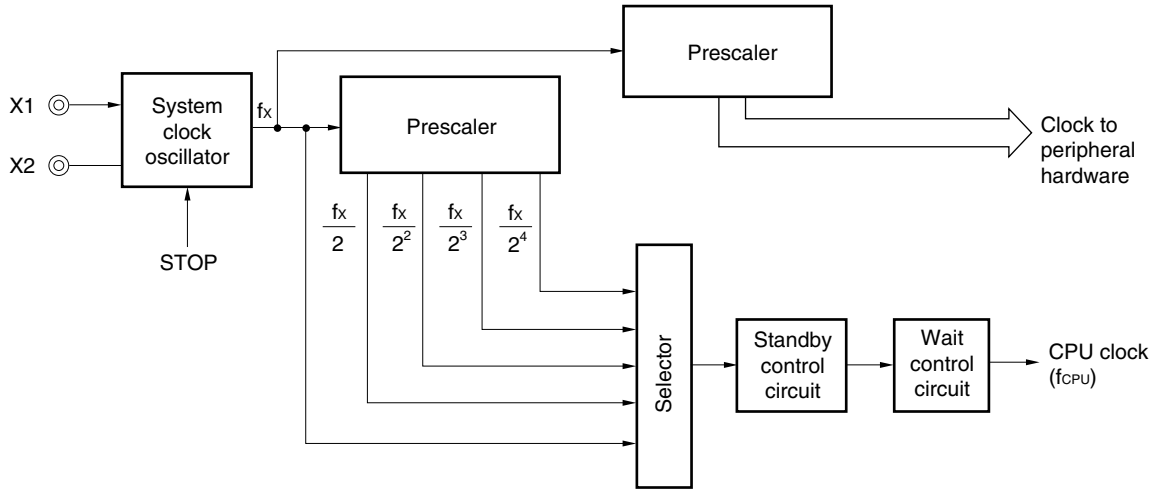
5.2 Clock Generator Configuration

The clock generator consists of the following hardware.

Table 5-1. Configuration of Clock Generator

Item	Configuration
Control register	Processor clock control register (PCC)
Oscillator	System clock oscillator

Figure 5-1. Block Diagram of Clock Generator



5.3 Clock Generator Control Register

The clock generator is controlled by the processor clock control register (PCC).

This register is used to select the CPU clock.

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

$\overline{\text{RESET}}$ input sets the PCC to 04H.

Figure 5-2. Format of Processor Clock Control Register (PCC)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PCC	0	0	0	0	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W

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

Caution Bits 3 to 7 must be set to 0.

Remark fx: System clock oscillation frequency

The fastest instructions of the μPD780701Y Subseries are carried out in 2 CPU clocks. The relationship between the CPU clock (f_{CPU}) and minimum instruction execution time is shown in Table 5-2.

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

CPU Clock (f_{CPU})	Minimum Instruction Execution Time: $2/f_{CPU}$
f_X	0.32 μs
$f_X/2$	0.64 μs
$f_X/2^2$	1.27 μs
$f_X/2^3$	2.54 μs
$f_X/2^4$	5.09 μs

fx = 6.29 MHz

fx: System clock oscillation frequency

5.4 System Clock Oscillator

5.4.1 System clock oscillator

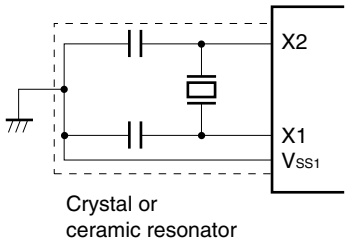
The system clock oscillator oscillates with a crystal or ceramic resonator (standard: 6.29 MHz) connected to the X1 and X2 pins.

External clocks can also be used. In this case, input a clock signal to the X1 pin and an inverted-phase clock signal to the X2 pin.

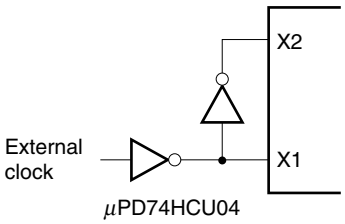
Figure 5-3 shows the external circuit of the system clock oscillator.

Figure 5-3. External Circuit of System Clock Oscillator

(a) Crystal and ceramic oscillation



(b) External clock



Cautions 1. Do not execute the STOP instruction during an input of the external clock. This is because the system clock operation is stopped, and the X2 pin is pulled up to V_{DD1}.

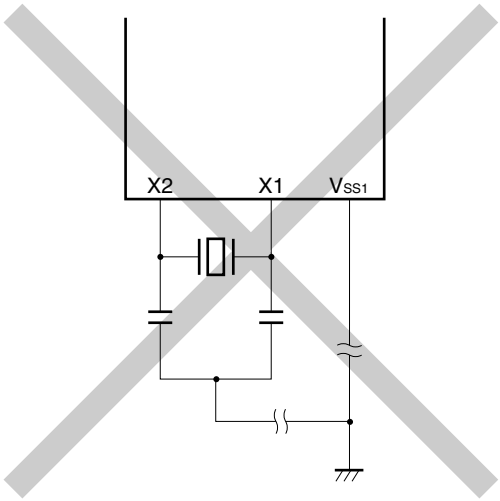
2. When using the system clock oscillator, wire the area enclosed by the broken lines in the above figures as follows to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with 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_{SS1}. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

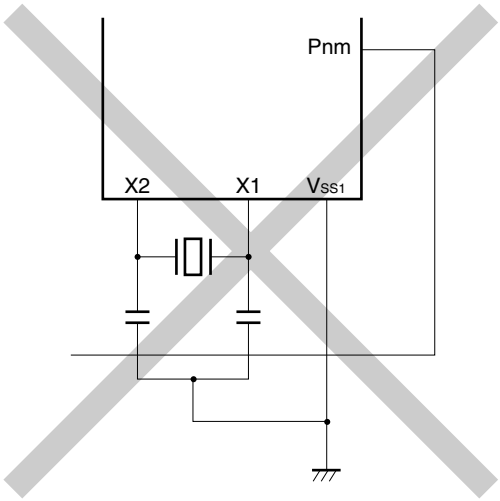
Figure 5-4 shows examples of incorrect oscillator connection.

Figure 5-4. Examples of Incorrect Oscillator Connection (1/2)

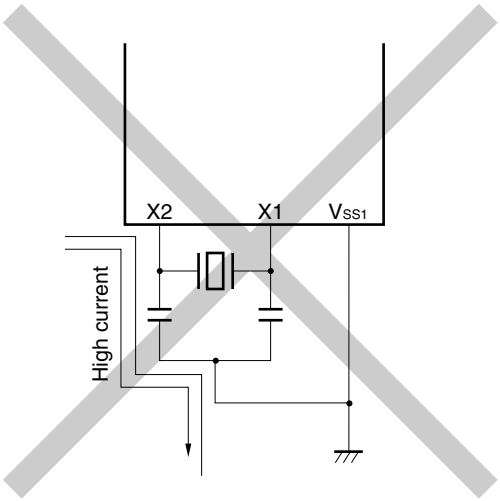
(a) Too long wiring



(b) Crossed signal line



(c) Wiring near high alternating current



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

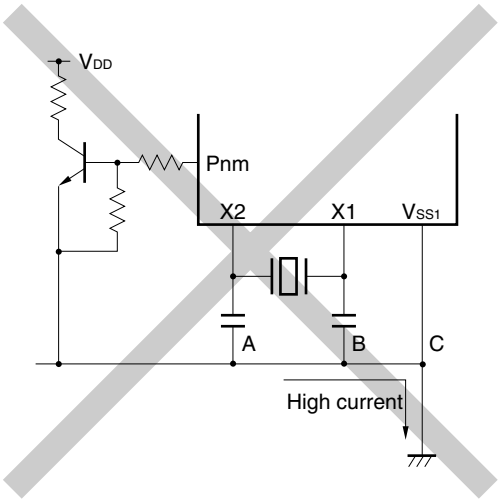
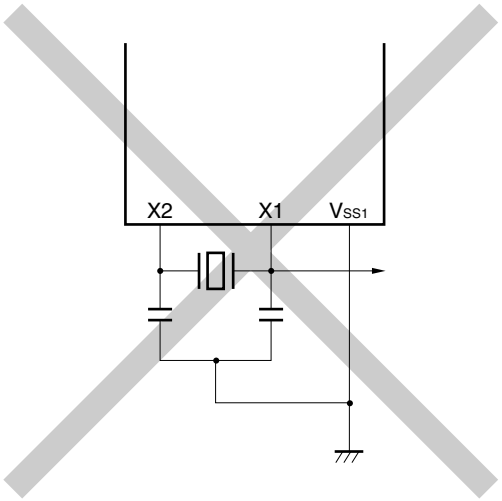


Figure 5-4. Examples of Incorrect Oscillator Connection (2/2)

(e) Signals are fetched



5.4.2 Frequency divider

The frequency divider divides the system clock oscillator output (fx) and generates various clocks.

5.5 Clock Generator Operations

The clock generator generates the following various types of clocks and controls the CPU operating modes such as the standby mode.

- System clock f_X
- CPU clock f_{CPU}
- Clock to peripheral hardware

The following clock generator functions and operations are determined by the processor clock control register (PCC).

(a) Upon generation of the $\overline{\text{RESET}}$ signal, the lowest speed mode of the system clock (5.09 μs: @ 6.29-MHz operation) is selected (PCC = 04H). System clock oscillation stops while low level is applied to the $\overline{\text{RESET}}$ pin.

(b) While the system clock is selected, one of the five CPU clock types (0.32 μs, 0.64 μs, 1.27 μs, 2.54 μs, 5.09 μs: @ 6.29-MHz operation) can be selected by setting the PCC.

(c) Two standby modes, STOP and HALT modes can be used.

(d) The system clock is divided and supplied to the peripheral hardware. Therefore, the peripheral hardware also stops if the system clock is stopped. (Except external input clock operation)

5.6 Changing CPU Clock Setting

5.6.1 Time required for switchover between CPU clocks

The CPU clock can be switched over by setting bits 0 to 2 (PCC0 to PCC2) of the processor clock control register (PCC).

The actual switchover operation is not performed immediately after writing to the PCC, and operation continues on the pre-switchover clock for several instructions (see **Table 5-3**).

Table 5-3. Maximum Time Required for CPU Clock Switchover

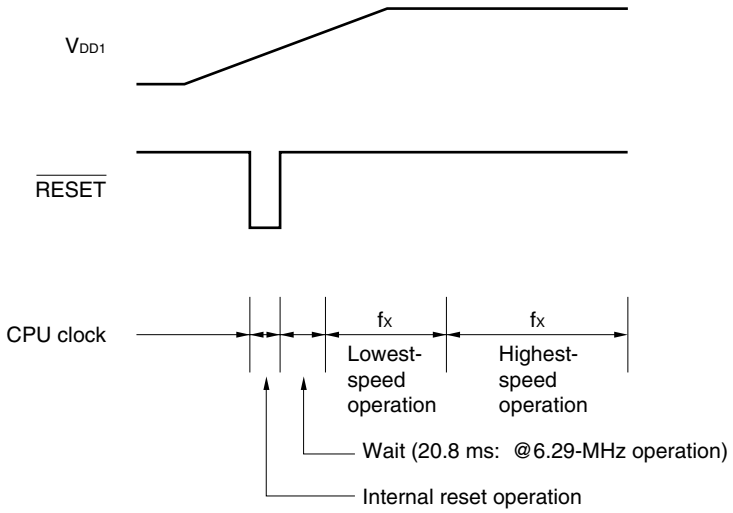
Set Value Before Switchover			Set Value After Switchover															
PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0
			0	0	0	0	0	1	0	1	0	0	1	1	1	0	0
0	0	0				16 instructions			16 instructions			16 instructions			16 instructions		
0	0	1	8 instructions						8 instructions			8 instructions			8 instructions		
0	1	0	4 instructions			4 instructions						4 instructions			4 instructions		
0	1	1	2 instructions			2 instructions			2 instructions						2 instructions		
1	0	0	1 instruction			1 instruction			1 instruction			1 instruction					

Remark One instruction is the minimum instruction execution time with the pre-switchover CPU clock.

5.6.2 CPU clock switching procedure

This section describes procedure of the switching between the CPU clocks.

Figure 5-5. System Clock and CPU Clock Switching



<1> The CPU is reset by setting the $\overline{\text{RESET}}$ signal to low level after power-on. After that, when reset is released by setting the $\overline{\text{RESET}}$ signal to high level, the system clock starts oscillation. At this time, the oscillation stabilization time ($2^{17}/f_X$) is secured automatically.

After that, the CPU starts executing the instruction at the minimum speed of the system clock (5.09 μs: @ 6.29-MHz operation).

<2> After sufficient time for the V_{DD1} voltage to increase to enable operation at maximum speed has elapsed, the PCC is rewritten and maximum-speed operations are carried out.

CHAPTER 6 16-BIT TIMER/EVENT COUNTER

6.1 Outline of Timer Integrated in μPD780701Y Subseries

This chapter explains the 16-bit timer/event counter. The internal timer of the μPD780701Y Subseries and related functions are first briefly explained below.

(1) 16-bit timer/event counters (TM00, TM01)

TM00 and TM01 can be used for various functions including as an interval timer, pulse width measurement (infrared-ray remote control receive function), external event counter, square-wave output of any frequency, PPG output, or one-shot pulse output.

(2) 8-bit timer/event counters (TM50, TM51, TM52)

TM50, TM51, and TM52 can be used as an interval timer, external event counter, square-wave output of any frequency, or PWM output. Moreover, two 8-bit timer/event counters can be used as one 16-bit timer/event counter (see **CHAPTER 7 8-BIT TIMER/EVENT COUNTER**).

(3) Watch timer (WTN0)

★ An interrupt request is generated at preset time intervals. Time intervals such as 0.5 or 1.0 second cannot be created in a 6.29-MHz system clock (see **CHAPTER 8 WATCH TIMER**).

(4) Watchdog timer (WDT)

The watchdog timer can also be used to generate a non-maskable interrupt request, maskable interrupt request, or $\overline{\text{RESET}}$ signal at preset time intervals (see **CHAPTER 9 WATCHDOG TIMER**).

(5) Clock output/buzzer output control circuit (CKU)

The clock output circuit supplies the clock made by dividing the system clock to other devices. The buzzer output circuit outputs the buzzer frequency made by dividing the system clock (see **CHAPTER 10 CLOCK OUTPUT/ BUZZER OUTPUT CONTROL CIRCUITS**).

Table 6-1. Timer/Event Counter Operations

		16-bit Timer/ Event Counter	8-bit Timer/ Event Counter	Watch Timer	Watchdog Timer
Operation Mode	Interval timer	2 channels	3 channels	1 channel[Note 1]	1 channel[Note 2]
	External event counter	√	√	—	—
Function	Timer output	√	√	—	—
	PPG output	√	—	—	—
	PWM output	—	√	—	—
	Pulse width measurement	√	—	—	—
	Square-wave output	√	√	—	—
	One-shot pulse output	√	—	—	—
	Interrupt request	√	√	√	√

Notes 1. The watch timer can perform both watch timer and interval timer functions at the same time.

2. The watchdog timer can perform either the watchdog timer function or the interval timer function. Therefore, select either one of the functions for use.

6.2 16-Bit Timer/Event Counter Functions

The 16-bit timer/event counters have the following functions.

- Interval timer
- PPG output
- Pulse width measurement
- External event counter
- Square-wave output
- One-shot pulse output

(1) Interval timer

TM0n generates an interrupt request at a preset time interval.

(2) PPG output

TM0n can output a square wave whose frequency and output pulse can be set freely.

(3) Pulse width measurement

TM0n can measure the pulse width of an externally input signal.

(4) External event counter

TM0n can measure the number of pulses of an externally input signal.

(5) Square-wave output

TM0n can output a square wave of any selected frequency.

(6) One-shot pulse output

TM0n is able to output a one-shot pulse that can set any width of output pulse.

6.3 16-Bit Timer/Event Counter Configuration

A 16-bit timer/event counter consists of the following hardware.

Table 6-2. Configuration of 16-Bit Timer/Event Counter

Item	Configuration
Timer register	16-bit timer/counter 0n (TM0n)
Register	16-bit capture/compare register 00n, 01n (CR00n, CR01n)
Timer output	TO0n
Control register	16-bit timer mode control register 0n (TMC0n) Capture/compare control register 0n (CRC0n) 16-bit timer output control register 0n (TOC0n) Prescaler mode register 0n (PRM0n) Port mode register 3, 7 (PM3, PM7)**Note**

Note See **Figure 4-4 Block Diagram of P30 to P32 and P34 to P37** and **Figure 4-10 Block Diagram of P70, P73 to P77.**

Remark n = 0, 1

Figure 6-1. Block Diagram of 16-Bit Timer/Event Counter 00 (TM00)

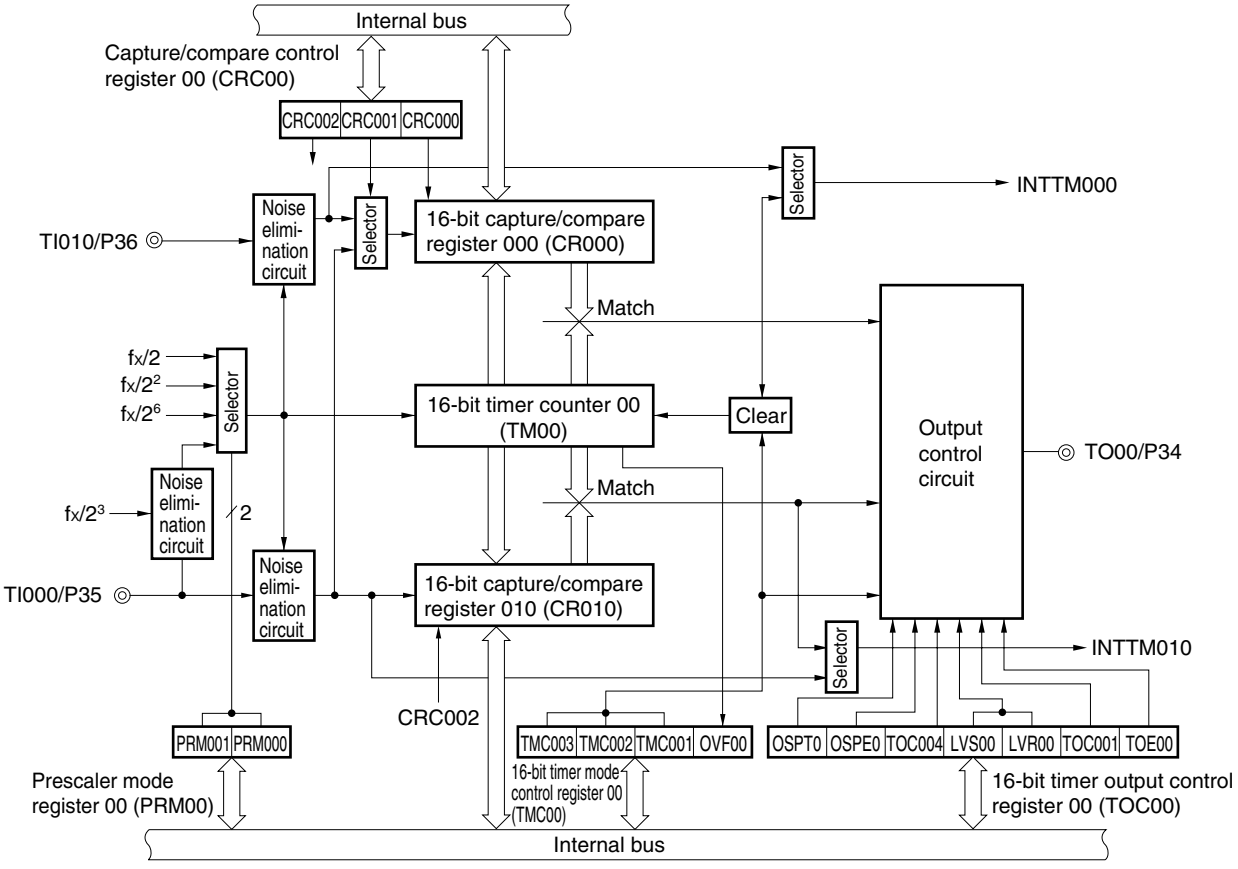
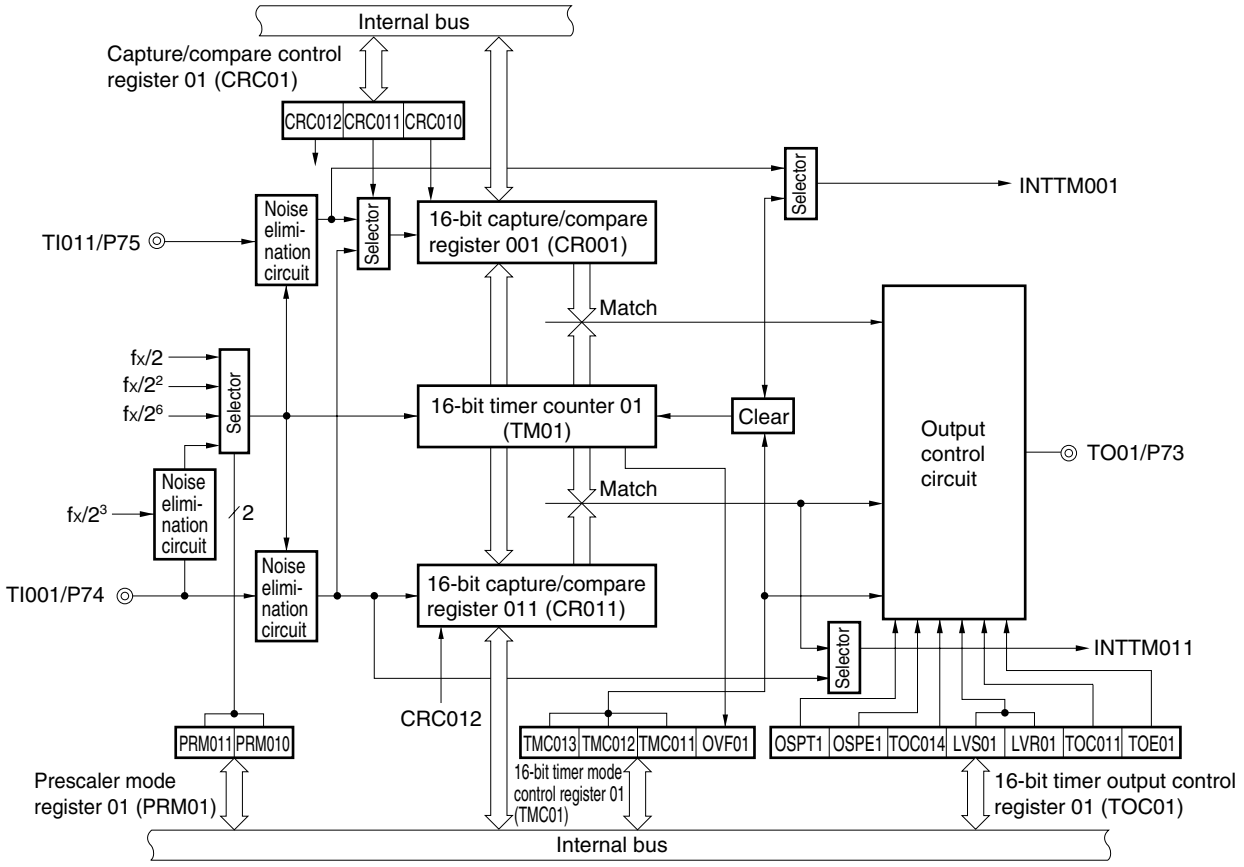


Figure 6-2. Block Diagram of 16-Bit Timer/Event Counter 01 (TM01)



(1) 16-bit timer/counter 00, 01 (TM00, TM01)

TM00 and TM01 are 16-bit read-only registers that count the count pulses.

The counter is incremented in synchronization with the rising edge of an input clock. If the count value is read during operation, input of the count clock is temporarily stopped, and the count value at that point is read. The count value is reset to 0000H in the following cases:

<1> At $\overline{\text{RESET}}$ input
<2> If TMC0n3 and TMC0n2 are cleared
<3> If the valid edge of TI00n is input in the clear & start mode by inputting the valid edge of TI00n
<4> If TM0n and CR00n match each other in the clear & start mode on match between TM0n and CR00n
<5> If OSPTn is set or if the valid edge of TI00n is input in the one-shot pulse output mode

Remark n = 0, 1

(2) 16-bit capture/compare register 000, 001 (CR000, CR001)

CR000 and CR001 are 16-bit registers with the functions of both a capture register and a compare register. Whether they are used as capture registers or as compare registers is set by bit 0 (CRC0n0) of capture/compare control register 0n (CRC0n).

- **When CR00n is used as a compare register**

  The value set in CR00n is constantly compared with the 16-bit timer/counter 0n (TM0n) count value, and an interrupt request (INTTM00n) is generated if they match. It can also be used as the register which holds the interval time when TM0n is set to interval timer operation.

- **When CR00n is used as a capture register**

  It is possible to select the valid edge of the TI00n pin or the TI01n pin as the capture trigger. Setting of the TI00n or TI01n valid edge is performed by means of prescaler mode register 0n (PRM0n).

  If CR00n is specified as a capture register and the capture trigger is specified to be the valid edge of the TI00n pin, the situation is as shown in Table 6-3. On the other hand, when the capture trigger is specified to be the valid edge of the TI01n pin, the situation is as shown in Table 6-4.

Table 6-3. TI00n Pin Valid Edge and Capture/Compare Register Capture Trigger

ES0n1	ES0n0	TI00n Pin Valid Edge	CR00n Capture Trigger	CR01n Capture Trigger
0	0	Falling edge	Rising edge	Falling edge
0	1	Rising edge	Falling edge	Rising edge
1	0	Setting prohibited	Setting prohibited	Setting prohibited
1	1	Both rising and falling edges	No capture operation	Both rising and falling edges

n = 0, 1

Table 6-4. TI01n Pin Valid Edge and Capture/Compare Register Capture Trigger

ES1n1	ES1n0	TI01n Pin Valid Edge	CR00n Capture Trigger
0	0	Falling edge	Falling edge
0	1	Rising edge	Rising edge
1	0	Setting prohibited	Setting prohibited
1	1	Both rising and falling edges	Both rising and falling edges

n = 0, 1

CR00n is set with a 16-bit memory manipulation instruction.

★ $\overline{\text{RESET}}$ input sets CR00n to 0000H.

Cautions 1. Set CR00n (n = 0, 1) to a value other than 0000H. When used as an event counter, it is not possible to count a single pulse. However, in the free-running mode and the clear mode of the valid edge of TI00n (n = 0, 1), if 0000H is set for CR00n (n = 0, 1), an interrupt request (INTTM00n: n = 0, 1) is generated after the overflow (FFFFH).

2. If the value of CR00n (n = 0, 1) after changing is smaller than the value of 16-bit timer/counter 0n (TM0n: n = 0, 1), TM0n continues counting and overflows, then starts counting again from 0. Also, if the value of CR00n after changing is less than the value before changing, it is necessary to restart the timer after CR00n changes.

3. When P35 (P74) is used as the valid edge of TI000 (TI001), it cannot be used as a timer output (TO00 (TO01)). Also, if it is used as TO00 (TO01), it cannot be used as the valid edge of TI000 (TI001).

Remark n = 0, 1

(3) 16-bit capture/compare register 010, 011 (CR010, CR011)

CR010 and CR011 are 16-bit registers with the functions of both a capture register and a compare register. Whether they are used as capture registers or compare registers is set by bit 2 (CRC0n2) of capture/compare control register 0n (CRC0n).

- **When CR01n is used as a compare register**
  The value set in CR01n is constantly compared with the 16-bit timer/counter 0n (TM0n) count value, and an interrupt request (INTTM01n) is generated if they match.

- **When CR01n is used as a capture register**
  It is possible to select the valid edge of the TI00n pin as the capture trigger. The TI00n valid edge is set by means of prescaler mode register 0n (PRM0n).

CR01n is set with a 16-bit memory manipulation instruction.

★ $\overline{\text{RESET}}$ input sets CR01n to 0000H.

Caution Set CR01n (n = 0, 1) to a value other than 0000H. When used as an event counter, it is not possible to count a single pulse. However, in the free-running mode and the clear mode of the valid edge of TI01n (n = 0, 1), if 0000H is set for CR01n (n = 0, 1), an interrupt request (INTTM01n: n = 0, 1) is generated after the overflow (FFFFH).

Remark n = 0, 1

6.4 16-Bit Timer/Event Counter Control Registers

The following five types of registers are used to control 16-bit timer/event counter 00, 01.

- 16-bit timer mode control register 00, 01 (TMC00, TMC01)
- Capture/compare control register 00, 01 (CRC00, CRC01)
- 16-bit timer output control register 00, 01 (TOC00, TOC01)
- Prescaler mode register 00, 01 (PRM00, PRM01)
- Port mode register 3, 7 (PM3, PM7)

(1) 16-bit timer mode control register 00, 01 (TMC00, TMC01)

These registers set the 16-bit timer operating mode, the 16-bit timer/counter 00, 01 (TM00, TM01) clear mode and output timing, and detects an overflow.

TMC00 and TMC01 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC00 and TMC01 values to 00H.

Caution The 16-bit timer/counter 0n (TM0n) starts operating the instant a value other than 0 (operation stop mode) is set for TMC0n2 and TMC0n3 (n = 0, 1). To stop operation, set TMC0n2 and TMC0n3 to 0.

Figure 6-3. Format of 16-Bit Timer Mode Control Register 00 (TMC00)

Address: FF60H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
TMC00	0	0	0	0	TMC003	TMC002	TMC001	OVF00

TMC003	TMC002	TMC001	Operating mode and clear mode selection	TO00 output timing selection	Interrupt request generation
0	0	0	Operation stop (TM00 cleared to 0)	No change	Not generated
0	0	1			
0	1	0	Free-running mode	Match between TM00 and CR000 or match between TM00 and CR010	Generated on match between TM00 and CR000, or match between TM00 and CR010
0	1	1		Match between TM00 and CR000, match between TM00 and CR010, or TI000 valid edge	
1	0	0	Clear & start on TI000 valid edge	Match between TM00 and CR000 or match between TM00 and CR010	
1	0	1		Match between TM00 and CR000, match between TM00 and CR010, or TI000 valid edge	
1	1	0	Clear & start on match between TM00 and CR000	Match between TM00 and CR000 or match between TM00 and CR010	
1	1	1		Match between TM00 and CR000, match between TM00 and CR010, or TI000 valid edge	

OVF00	16-bit timer/counter 00 (TM00) overflow detection
0	Overflow not detected
1	Overflow detected

Cautions 1. Write to a bit other than the OVF00 flag after timer operation stops.

2. Set the valid edge of the TI000/P35 pin with prescaler mode register 00 (PRM00).

3. If clear & start mode on a match between TM00 and CR000 is selected, when the set value of CR000 is FFFFH and the TM00 value changes from FFFFH to 0000H, the OVF00 flag is set to 1.

Remarks TO00: 16-bit timer/event counter 00 output pin
TI000: 16-bit timer/event counter 00 input pin
TM00: 16-bit timer/counter 00
CR000: Compare register 000
CR010: Compare register 010

Figure 6-4. Format of 16-Bit Timer Mode Control Register 01 (TMC01)

Address: FF68H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
TMC01	0	0	0	0	TMC013	TMC012	TMC011	OVF01

TMC013	TMC012	TMC011	Operating mode and clear mode selection	TO01 output timing selection	Interrupt request generation
0	0	0	Operation stop (TM01 cleared to 0)	No change	Not generated
0	0	1			
0	1	0	Free-running mode	Match between TM01 and CR001 or match between TM01 and CR011	Generated on match between TM01 and CR001, or match between TM01 and CR011
0	1	1		Match between TM01 and CR001, match between TM01 and CR011, or TI001 valid edge	
1	0	0	Clear & start on TI001 valid edge	Match between TM01 and CR001 or match between TM01 and CR011	
1	0	1		Match between TM01 and CR001, match between TM01 and CR011, or TI001 valid edge	
1	1	0	Clear & start on match between TM01 and CR001	Match between TM01 and CR001 or match between TM01 and CR011	
1	1	1		Match between TM01 and CR001, match between TM01 and CR011, or TI001 valid edge	

OVF01	16-bit timer/counter 01 (TM01) overflow detection
0	Overflow not detected
1	Overflow detected

Cautions 1. Write to a bit other than the OVF01 flag after timer operation stops.

2. Set the valid edge of the TI001/P74 pin with prescaler mode register 01 (PRM01).

3. If clear & start mode on a match between TM01 and CR001 is selected, when the set value of CR001 is FFFFH and the TM01 value changes from FFFFH to 0000H, the OVF01 flag is set to 1.

Remarks TO01: 16-bit timer/event counter 01 output pin
TI001: 16-bit timer/event counter 01 input pin
TM01: 16-bit timer/counter 01
CR001: Compare register 001
CR011: Compare register 011

(2) Capture/compare control register 00, 01 (CRC00, CRC01)

These registers control the operation of the 16-bit capture/compare registers (CR000, CR010, CR001, CR011).

CRC00 and CRC01 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CRC00 and CRC01 values to 00H.

Figure 6-5. Format of Capture/Compare Control Register 00 (CRC00)

Address: FF62H After reset: 00H R/W

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

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

CRC001	CR000 capture trigger selection
0	Captures on valid edge of TI010
1	Captures on inverted phase of valid edge of TI000

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

Cautions 1. Timer operation must be stopped before setting CRC00.

2. When clear & start mode on a match between TM00 and CR000 is selected with 16-bit timer mode control register 00 (TMC00), CR000 should not be specified as a capture register.

3. If both the rising and falling edges are selected as valid edge of TI000, capture is not performed.

Figure 6-6. Format of Capture/Compare Control Register 01 (CRC01)

Address: FF6AH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CRC01	0	0	0	0	0	CRC012	CRC011	CRC010

CRC012	CR011 operating mode selection
0	Operates as compare register
1	Operates as capture register

CRC011	CR001 capture trigger selection
0	Captures on valid edge of TI011
1	Captures on inverted phase of valid edge of TI001

CRC010	CR001 operating mode selection
0	Operates as compare register
1	Operates as capture register

Cautions 1. Timer operation must be stopped before setting CRC01.

2. When clear & start mode on a match between TM01 and CR001 is selected with 16-bit timer mode control register 01 (TMC01), CR001 should not be specified as a capture register.

3. If both the rising and falling edges are selected as valid edge of TI001, capture is not performed.

(3) 16-bit timer output control register 00, 01 (TOC00, TOC01)

These registers control the operation of the 16-bit timer/event counter 00, 01 output control circuit. It sets R-S type flip-flop (LV0) setting/resetting, output inversion enabling/disabling, 16-bit timer/event counter 00, 01 timer output enabling/disabling, one-shot pulse output operation enabling/disabling, and the output trigger for a one-shot pulse by means of software.

TOC00 and TOC01 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TOC00 and TOC01 values to 00H.

Figure 6-7. Format of 16-Bit Timer Output Control Register 00 (TOC00)

Address: FF63H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
TOC00	0	OSPT0	OSPE0	TOC004	LVS00	LVR00	TOC001	TOE00

OSPT0	Control of one-shot pulse output trigger by software
0	One-shot pulse trigger not used
1	One-shot pulse trigger used

OSPE0	One-shot pulse output control
0	Continuous pulse output
1	One-shot pulse output[Note]

TOC004	Timer output F/F control by match of CR010 and TM00
0	Inversion operation disabled
1	Inversion operation enabled

LVS00	LVR00	16-bit timer/event counter 00 timer output F/F status setting
0	0	No change
0	1	Timer output F/F reset (0)
1	0	Timer output F/F set (1)
1	1	Setting prohibited

TOC001	Timer output F/F control by match of CR000 and TM00
0	Inversion operation disabled
1	Inversion operation enabled

TOE00	16-bit timer/event counter 00 timer output control
0	Output disabled (Output set to level 0)
1	Output enabled

Note The one-shot pulse output operates normally only in the free-running mode and in the clear and start mode at the valid edge of TI000.

Caution Timer operation must be stopped before setting TOC00.

Remarks 1. If LVS00 and LVR00 are read after data is set, they will be 0.

2. OSPT0 is cleared automatically after the data are set, so when read it is 0.

Figure 6-8. Format of 16-Bit Timer Output Control Register 01 (TOC01)

Address: FF6BH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
TOC01	0	OSPT1	OSPE1	TOC014	LVS01	LVR01	TOC011	TOE01

OSPT1	Control of one-shot pulse output trigger by software
0	One-shot pulse trigger not used
1	One-shot pulse trigger used

OSPE1	One-shot pulse output control
0	Continuous pulse output
1	One-shot pulse output[Note]

TOC014	Timer output F/F control by match of CR011 and TM01
0	Inversion operation disabled
1	Inversion operation enabled

LVS01	LVR01	16-bit timer/event counter 01 timer output F/F status setting
0	0	No change
0	1	Timer output F/F reset (0)
1	0	Timer output F/F set (1)
1	1	Setting prohibited

TOC011	Timer output F/F control by match of CR001 and TM01
0	Inversion operation disabled
1	Inversion operation enabled

TOE01	16-bit timer/event counter 01 timer output control
0	Output disabled (Output set to level 0)
1	Output enabled

Note The one-shot pulse output operates normally only in the free-running mode and in the clear and start mode at the valid edge of TI000.

Caution Timer operation must be stopped before setting TOC01.

Remarks 1. If LVS01 and LVR01 are read after data is set, they will be 0.

2. OSPT1 is cleared automatically after the data are set, so when read it is 0.

(4) Prescaler mode register 00, 01 (PRM00, PRM01)

These registers are used to set the 16-bit timer/counter 00, 01 (TM00, TM01) count clock and TI000, TI001 input valid edges.

PRM00 and PRM01 are set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PRM00 and PRM01 values to 00H.

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

Address: FF61H After reset: 00H R/W

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

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

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

PRM001	PRM000	Count clock selection
0	0	fx/2 (3.15 MHz)
0	1	$fx/2^2$ (1.57 MHz)
1	0	$fx/2^6$ (98.3 kHz)
1	1	TI000 valid edge

Cautions 1. If the valid edge of TI000 is set for the count clock, do not set it for the clear and start mode or the capture trigger.

2. Timer operation must be stopped before setting PRM00.

3. For a capture trigger to capture with certainty, a pulse with a length of more than 2 selected count clocks is required. Similarly, the external clock requires a pulse with a length of more than 2 internal clocks. In addition, in cases where TI000 or TI010 is at the high level immediately after system reset, the rising edge is detected immediately after TM00 operation is permitted. Be sure to exercise caution when pulling-up, etc.

Remarks 1. fx: System clock oscillation frequency

2. TI000, TI010: Input pins of 16-bit timer/event counter 00

3. Figures in parentheses apply to operation with fx = 6.29 MHz.

Figure 6-10. Format of Prescaler Mode Register 01 (PRM01)

Address: FF69H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PRM01	ES111	ES101	ES011	ES001	0	0	PRM011	PRM010

ES111	ES101	TI011 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

ES011	ES001	TI001 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

PRM011	PRM010	Count clock selection
0	0	fx/2 (3.15 MHz)
0	1	$fx/2^2$ (1.57 MHz)
1	0	$fx/2^6$ (98.3 kHz)
1	1	TI001 valid edge

Cautions 1. If the valid edge of TI001 is set for the count clock, do not set it for the clear and start mode or the capture trigger.

2. Timer operation must be stopped before setting PRM01.

3. For a capture trigger to capture with certainty, a pulse with a length of more than 2 selected count clocks is required. The external clock also requires a pulse with a length of more than 2 internal clocks ($fx/2^3$). In addition, in cases where TI001 or TI011 is at the high level immediately after system reset, the rising edge is detected immediately after TM01 operation is permitted. Be sure to exercise caution when pulling-up, etc.

Remarks 1. fx: System clock oscillation frequency

2. TI001, TI011: Input pins of 16-bit timer/event counter 01

3. Figures in parentheses apply to operation with fx = 6.29 MHz.

(5) Port mode register 3, 7 (PM3, PM7)

These registers set ports 3 and 7 to input/output in 1-bit units.

When using the P34/TO00 pin, P73/TO01 pin for timer output, set PM34, PM73 and the output latch of P34, P73 to 0.

PM3 and PM7 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM3, PM7 value to FFH.

Figure 6-11. Format of Port Mode Register 3 (PM3)

Address: FF23H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM3	1	PM36	PM35	PM34	PM33	PM32	PM31	PM30

PM3n	P3n pin input/output mode selection (n = 0 to 6)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

Figure 6-12. Format of Port Mode Register 7 (PM7)

Address: FF27H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70

PM7n	P7n pin input/output mode selection (n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

6.5 16-Bit Timer/Event Counter Operations

6.5.1 Interval timer operation

Setting 16-bit timer mode control register 0n (TMC0n) and capture/compare control register 0n (CRC0n) as shown in Figure 6-13 allows operation as an interval timer. Interrupt requests are generated repeatedly using the count value set in 16-bit capture/compare register 00n (CR00n) beforehand as the interval.

When the count value of 16-bit timer/counter 0n (TM0n) matches the value set to CR00n, counting continues with the TM0n value cleared to 0 and the interrupt request signal (INTTM00n) is generated.

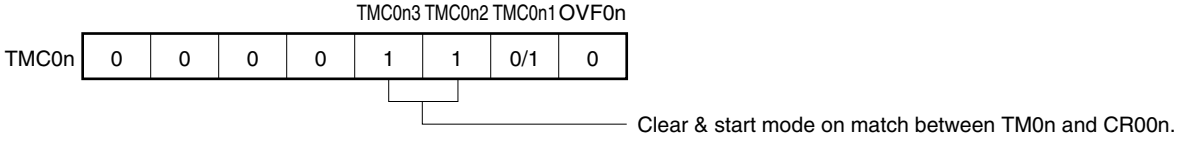
The count clock of TM0n can be selected with bits 0 and 1 (PRM0n0, PRM0n1) of prescaler mode register 0n (PRM0n).

See **6.6 Notes on 16-Bit Timer/Event Counter (3) Operation after compare register change during timer count operation** about the operation when the compare register value is changed during timer count operation.

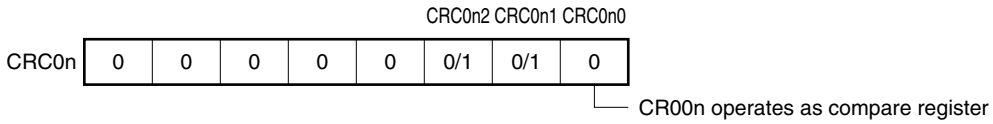
Remark n = 0, 1

Figure 6-13. Control Register Settings for Interval Timer Operation

(a) 16-bit timer mode control register 0n (TMC0n)



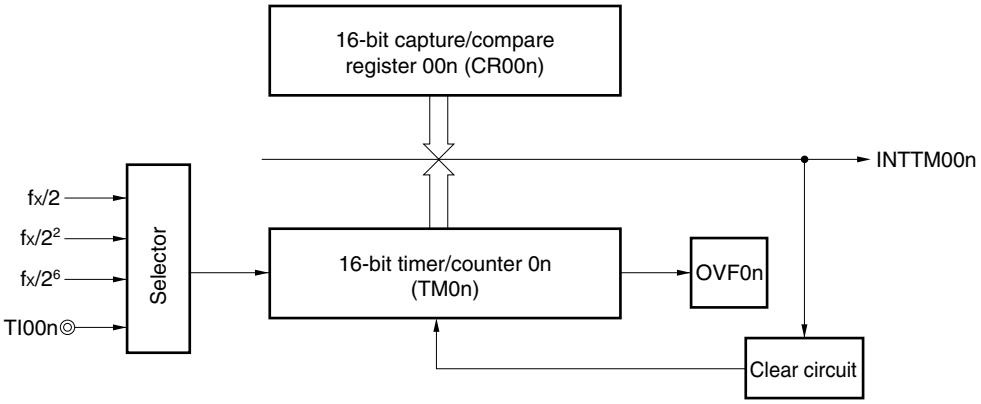
(b) Capture/compare control register 0n (CRC0n)



Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with the interval timer. See Figures 6-3 to 6-6.

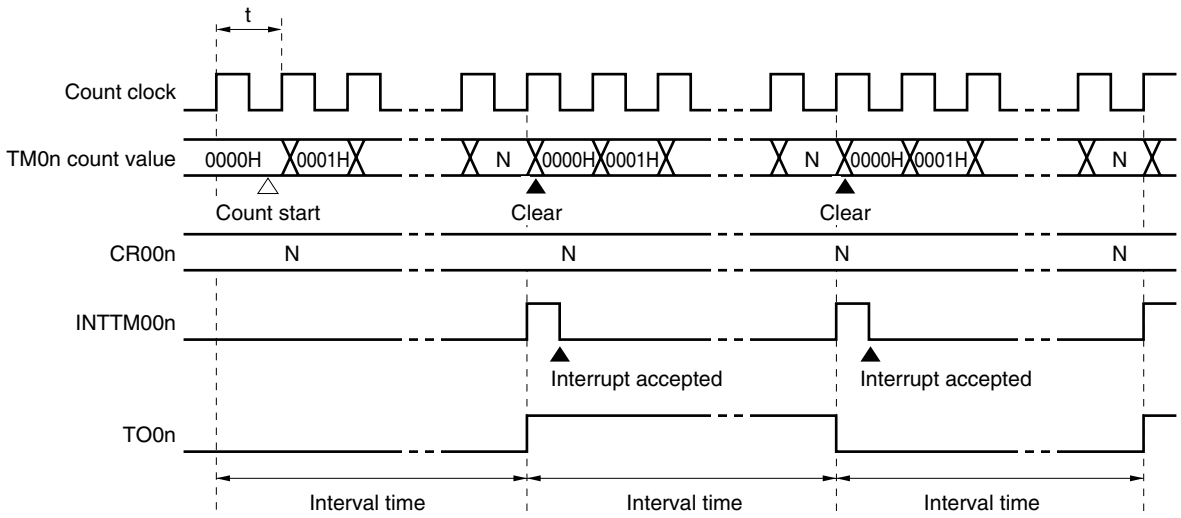
2. n = 0, 1

Figure 6-14. Configuration Diagram of Interval Timer



Remark n = 0, 1

Figure 6-15. Timing of Interval Timer Operation



Remarks 1. Interval time = (N + 1) × t: N = 0001H to FFFFH

2. n = 0, 1

6.5.2 PPG output operation

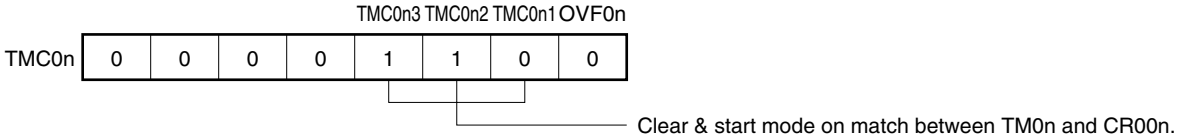
Setting 16-bit timer mode control register 0n (TMC0n) and capture/compare control register 0n (CRC0n) as shown in Figure 6-16 allows operation as a PPG (Programmable Pulse Generator) output.

In the PPG output operation, square waves are output from the TO0n pin with the pulse width and the cycle that correspond to the count values set beforehand in 16-bit capture/compare register 01n (CR01n) and in 16-bit capture/compare register 00n (CR00n), respectively.

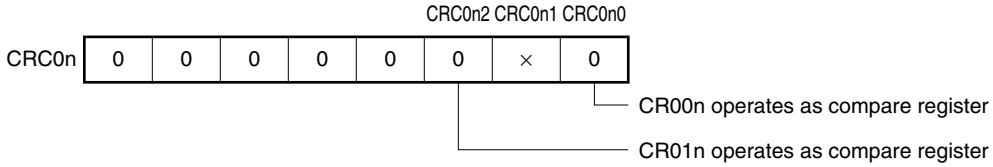
Remark n = 0, 1

Figure 6-16. Control Register Settings for PPG Output Operation

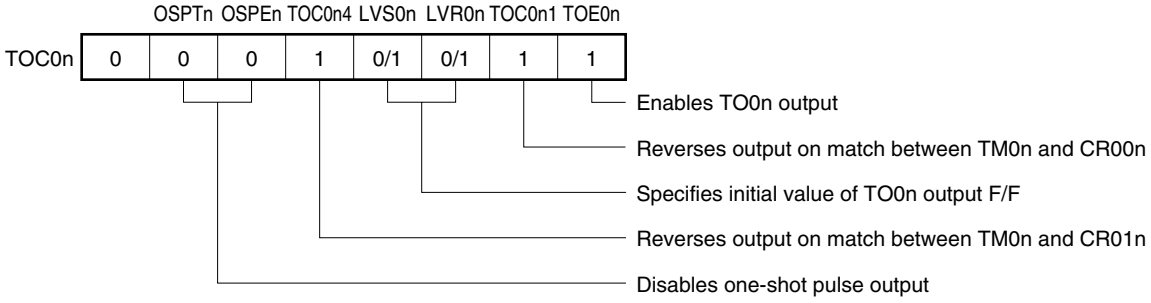
(a) 16-bit timer mode control register 0n (TMC0n)



(b) Capture/compare control register 0n (CRC0n)



(c) Timer output control register 0n (TOC0n)



Cautions 1. Values in the following range should be set in CR00n and CR01n:

0000H < CR01n < CR00n ≤ FFFFH

2. The cycle of the pulse generated by PPG output becomes (CR00n setting + 1), and the duty becomes (CR01n setting + 1)/(CR00n setting + 1).

Remark × : don't care

n = 0, 1

6.5.3 Pulse width measurement operation

It is possible to measure the pulse width of the signals input to the TI00n pin and TI01n pin using 16-bit timer/counter 0n (TM0n).

There are two measurement methods: measuring with TM0n used in free-running mode, and measuring by restarting the timer in synchronization with the edge of the signal input to the TI00n pin.

(1) Pulse width measurement with free-running counter and one capture register

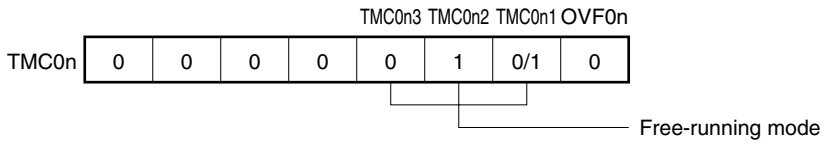
When 16-bit timer/counter 0n (TM0n) is operated in free-running mode (see register settings in **Figure 6-17**), if the edge specified by prescaler mode register 0n (PRM0n) is input to the TI00n pin, the value of TM0n is taken into 16-bit capture/compare register 01n (CR01n) and an external interrupt request signal (INTTM01n) is set. The valid edge of TI00n pin is specified by bits 4 and 5 (ES0n0, ES0n1) of PRM0n, and the rising, falling, or both rising and falling edges can be selected.

With TI00n pin valid edge detection, sampling is performed at the count clock interval selected using PRM0, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short-pulse width.

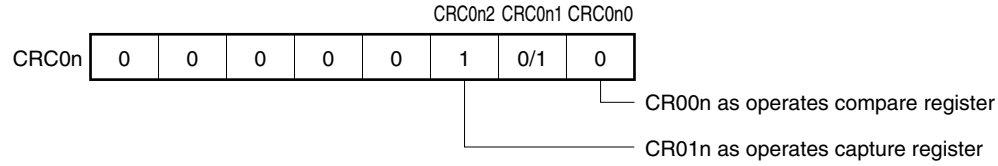
Remark n = 0, 1

Figure 6-17. Control Register Settings for Pulse Width Measurement with Free-Running Counter and One Capture Register

(a) 16-bit timer mode control register 0n (TMC0n)



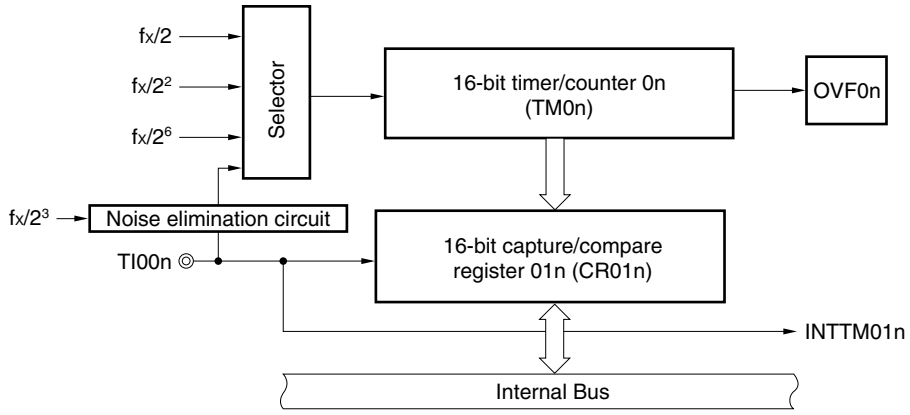
(b) Capture/compare control register 0n (CRC0n)



Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with pulse width measurement. See Figures 6-3 to 6-6.

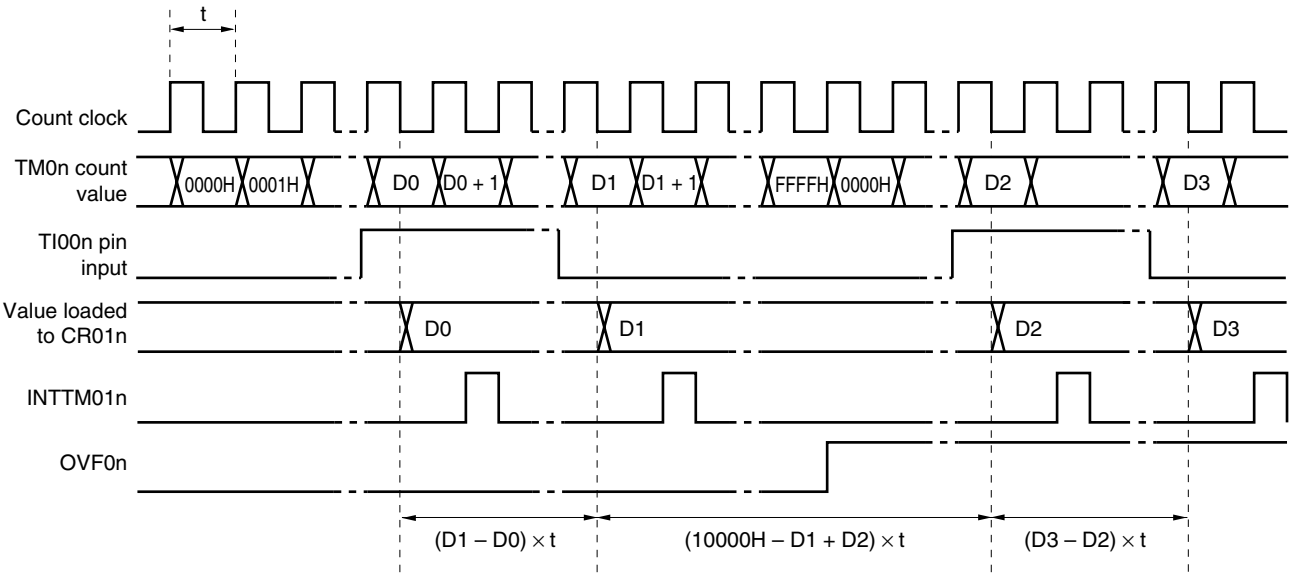
2. n = 0, 1

Figure 6-18. Configuration Diagram of Pulse Width Measurement by Free-Running Counter



Remark n = 0, 1

Figure 6-19. Timing of Pulse Width Measurement Operation by Free-Running Counter and One Capture Register (with Both Edges Specified)



Remark n = 0, 1

(2) Measurement of two pulse widths with free-running counter

When 16-bit timer/counter 0n (TM0n) is operated in free-running mode (see register settings in **Figure 6-20**), it is possible to simultaneously measure the pulse widths of the two signals input to the TI00n pin and the TI01n pin.

When the edge specified by bits 4 and 5 (ES0n0, ES0n1) of prescaler mode register 0n (PRM0n) is input to the TI00n pin, the value of TM0n is taken into 16-bit capture/compare register 01n (CR01n) and an external interrupt request signal (INTTM01n) is set.

Also, when the edge specified by bits 6 and 7 (ES1n0, ES1n1) of PRM0n is input to the TI01n pin, the value of TM0n is taken into 16-bit capture/compare register 00n (CR00n) and an external interrupt request signal (INTTM00n) is set.

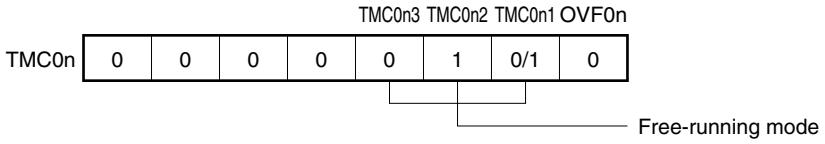
The valid edges of TI00n and TI01n pins are specified by bits 4 and 5 (ES0n0, ES0n1), and bits 6 and 7 (ES1n0, ES1n1) of PRM0n, respectively. It is possible to select the rising, falling, or both rising and falling edges as the valid edge.

With TI00n pin valid edge detection, sampling is performed at the count clock interval selected using PRM0, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short-pulse width.

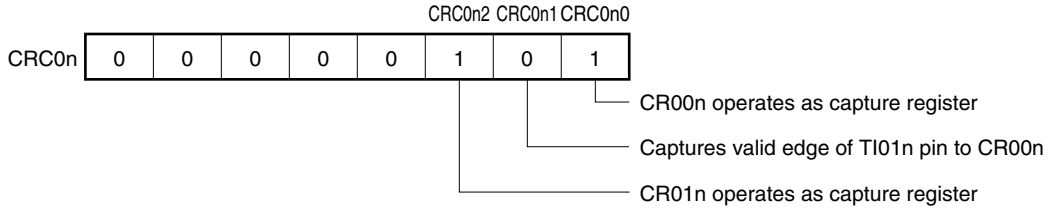
Remark n = 0, 1

Figure 6-20. Control Register Settings for Measurement of Two Pulse Widths with Free-Running Counter

(a) 16-bit timer mode control register 0n (TMC0n)



(b) Capture/compare control register 0n (CRC0n)



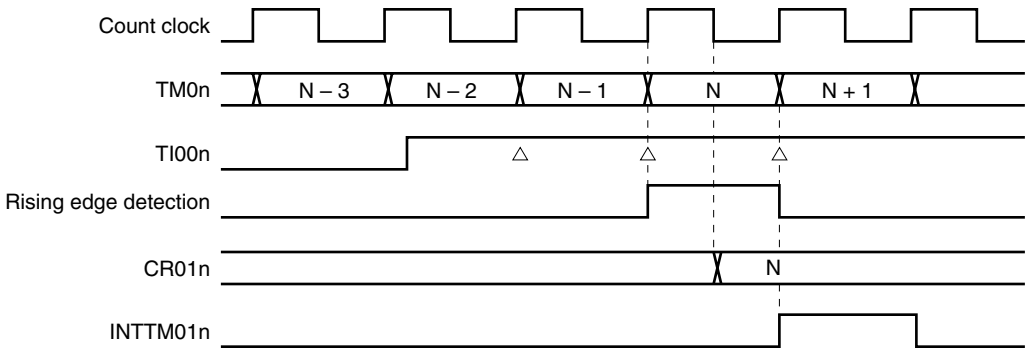
Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with pulse width measurement. See Figures 6-3 to 6-6.

2. n = 0, 1

- **Capture operation (free-running mode)**

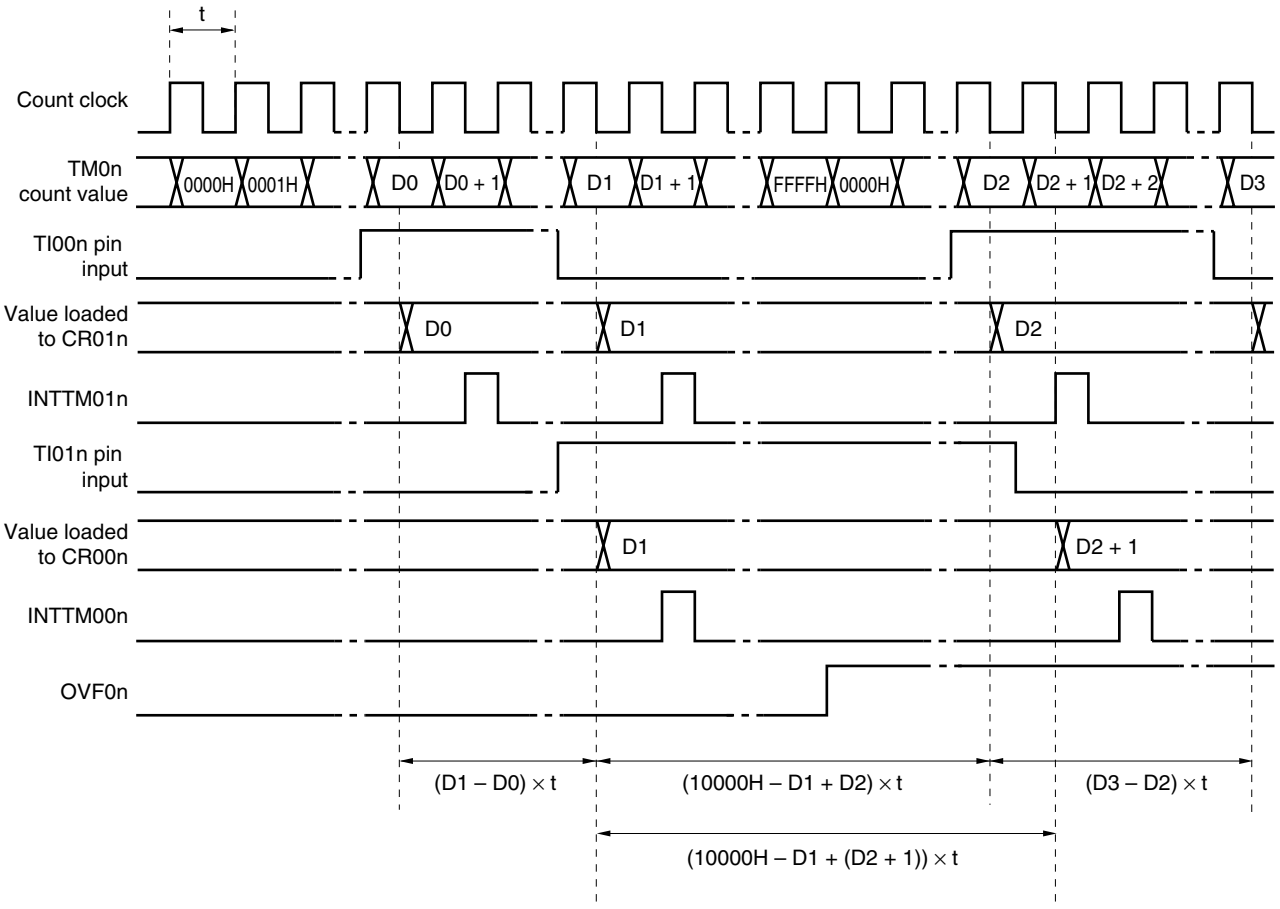
  A capture register operation when the capture trigger is input is shown.

Figure 6-21. CR01n Capture Operation with Rising Edge Specified



Remark n = 0, 1

Figure 6-22. Timing of Pulse Width Measurement Operation with Free-Running Counter (with Both Edges Specified)



Remark n = 0, 1

(3) Pulse width measurement with free-running counter and two capture registers

When 16-bit timer/counter 0n (TM0n) is operated in free-running mode (see register settings in **Figure 6-23**), it is possible to measure the pulse width of the signal input to the TI00n pin.

When the edge specified by bits 4 and 5 (ES0n0, ES0n1) of prescaler mode register 0n (PRM0n) is input to the TI00n pin, the value of TM0n is taken into 16-bit capture/compare register 01n (CR01n) and an external interrupt request signal (INTTM01n) is set.

Also, on the inverse edge input to that of the capture operation into CR01n, the value of TM0n is taken into 16-bit capture/compare register 00n (CR00n).

The valid edge of the TI00n pin is specified by bits 4 and 5 (ES0n0, ES0n1) of PRM0n, and it is possible to select the rising or falling edge.

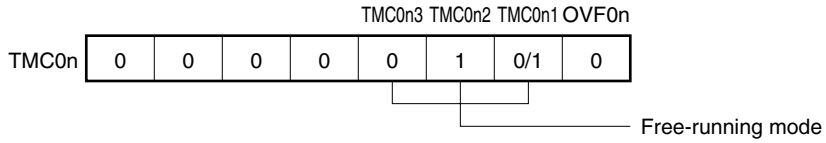
With TI00n pin valid edge detection, sampling is performed at the count clock interval selected using PRM0, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short-pulse width.

Caution If the valid edge of the TI00n pin is specified to be both rising and falling edges, capture/compare register 00n (CR00n) cannot perform the capture operation.

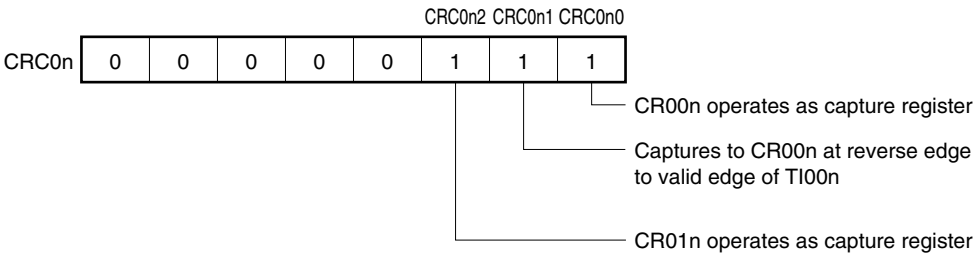
Remark n = 0, 1

Figure 6-23. Control Register Settings for Pulse Width Measurement with Free-Running Counter and Two Capture Registers

(a) 16-bit timer mode control register 0n (TMC0n)



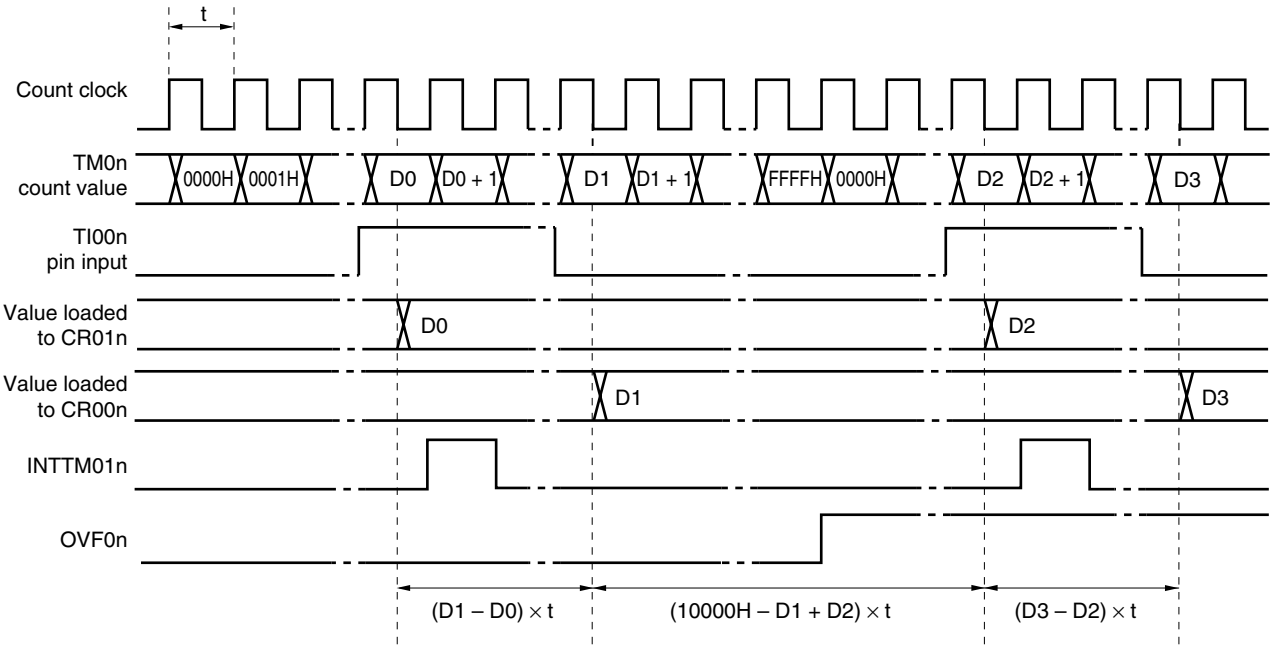
(b) Capture/compare control register 0n (CRC0n)



Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with pulse width measurement. See Figures 6-3 to 6-6.

2. n = 0, 1

Figure 6-24. Timing of Pulse Width Measurement Operation by Free-Running Counter and Two Capture Registers (with Rising Edge Specified)



Remark n = 0, 1

(4) Pulse width measurement by means of restart

When input of a valid edge to the TI00n pin is detected, the count value of 16-bit timer/counter 0n (TM0n) is taken into 16-bit capture/compare register 01n (CR01n), and the pulse width of the signal input to the TI00n pin is then measured by clearing TM0n and restarting the count (see register settings in **Figure 6-25**).

The valid edge of the TI00n pin is specified by bits 4 and 5 (ES0n0, ES0n1) of prescaler mode register 0n (PRM0n), and it is possible to select either the rising or falling edge.

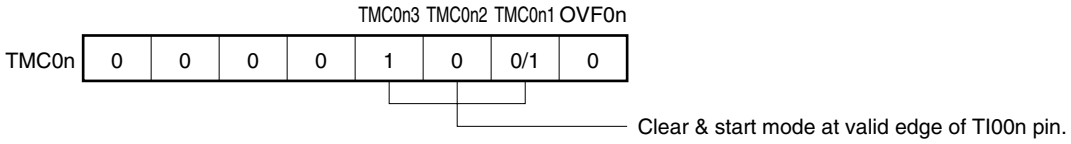
With TI00n pin valid edge detection, sampling is performed at the count clock interval selected using PRM0, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short-pulse width.

Caution If the valid edge of the TI00n pin is specified to be both rising and falling edges, capture/compare register 00n (CR00n) cannot perform the capture operation.

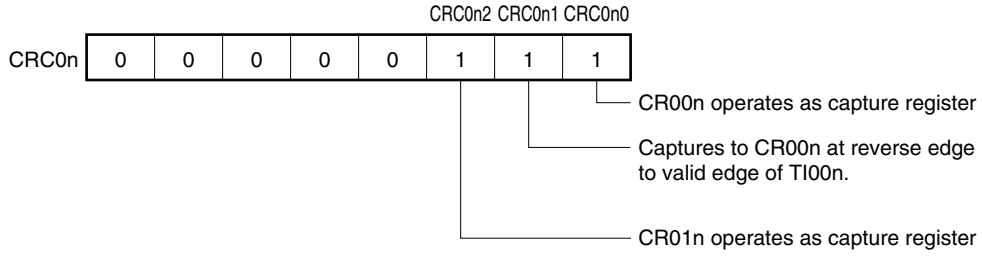
Remark n = 0, 1

Figure 6-25. Control Register Settings for Pulse Width Measurement by Means of Restart

(a) 16-bit timer mode control register 0n (TMC0n)



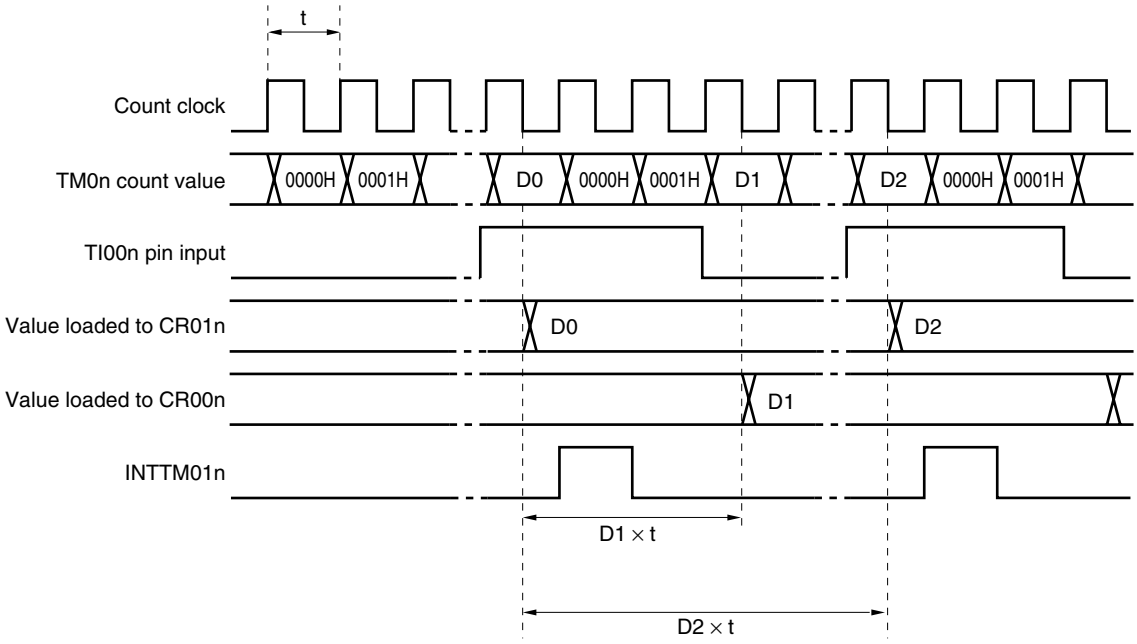
(b) Capture/compare control register 0n (CRC0n)



Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with pulse width measurement. See Figures 6-3 to 6-6.

2. n = 0, 1

Figure 6-26. Timing of Pulse Width Measurement Operation by Means of Restart (with Rising Edge Specified)



Remark n = 0, 1

6.5.4 External event counter operation

The external event counter counts the number of external clock pulses to be input to the TI00n pin with 16-bit timer/counter 0n (TM0n).

TM0n is incremented each time the valid edge specified with prescaler mode register 0n (PRM0n) is input.

When the TM0n counted value matches the 16-bit capture/compare register 00n (CR00n) value, TM0n is cleared to 0 and an interrupt request signal (INTTM00n) is generated.

A value other than 0000H should be set for CR00n (a 1 pulse count operation is not possible).

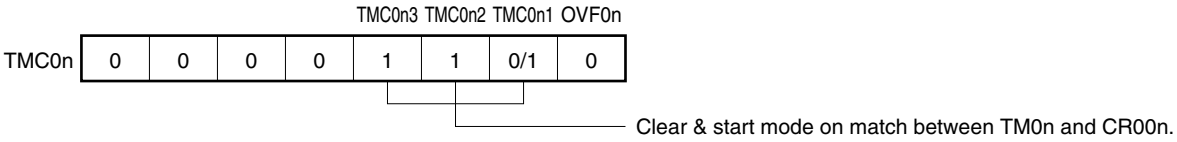
Specify the valid edge of the TI00n pin using bits 4 and 5 (ES0n0, ES0n1) of PRM0n. It is possible to select the rising, falling, or both rising and falling edges.

With TI00n pin valid edge detection, sampling is performed at the clock interval of $f_X 2^3$, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short-pulse width.

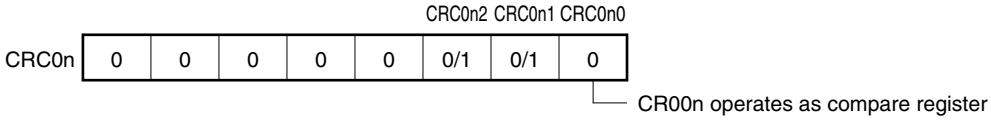
Remark n = 0, 1

Figure 6-27. Control Register Settings for External Event Counter Mode

(a) 16-bit timer mode control register 0n (TMC0n)



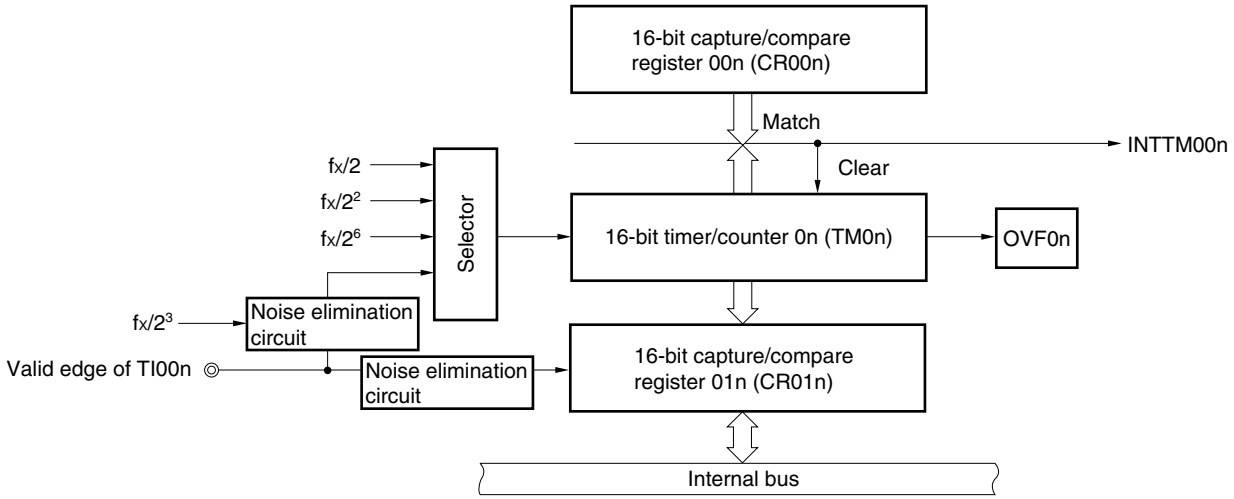
(b) Capture/compare control register 0n (CRC0n)



Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with the external event counter. See Figures 6-3 to 6-6.

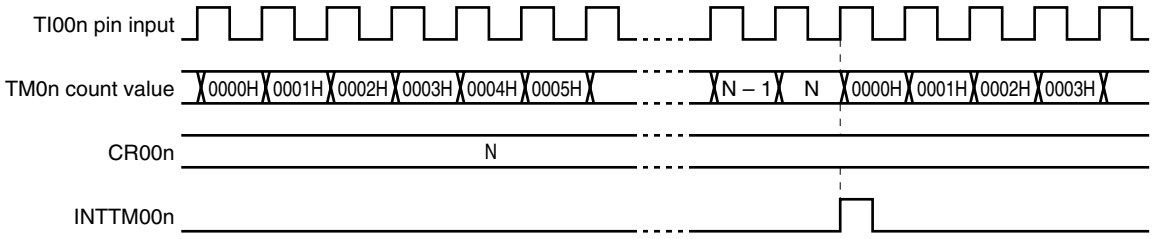
2. n = 0, 1

Figure 6-28. External Event Counter Configuration Diagram



Remark n = 0, 1

Figure 6-29. Timing of External Event Counter Operation (with Rising Edge Specified)



Caution When reading the external event counter count value, TM0n (n = 0, 1) should be read.

Remark n = 0, 1

6.5.5 Square-wave output operation

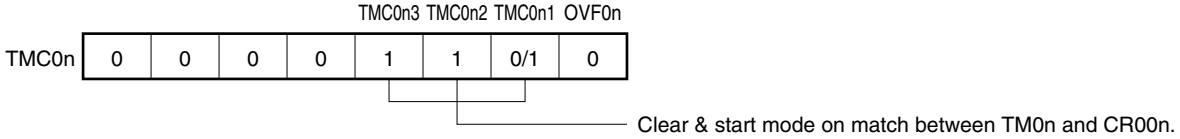
An operation whereby a square wave with any selected frequency is output at intervals taken from the count value preset to 16-bit capture/compare register 00n (CR00n).

The TO0n pin output status is reversed at intervals of the count value preset to CR00n by setting bit 0 (TOE0n) and bit 1 (TOC0n1) of 16-bit timer output control register 0n (TOC0n) to 1. This enables a square wave with any selected frequency to be output.

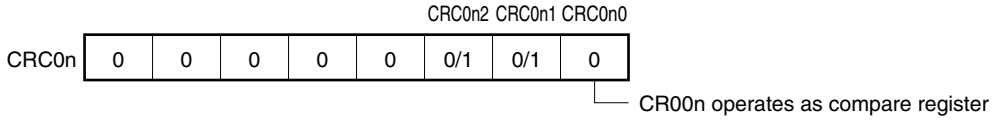
Remark n = 0, 1

Figure 6-30. Control Register Settings for Square-Wave Output Mode

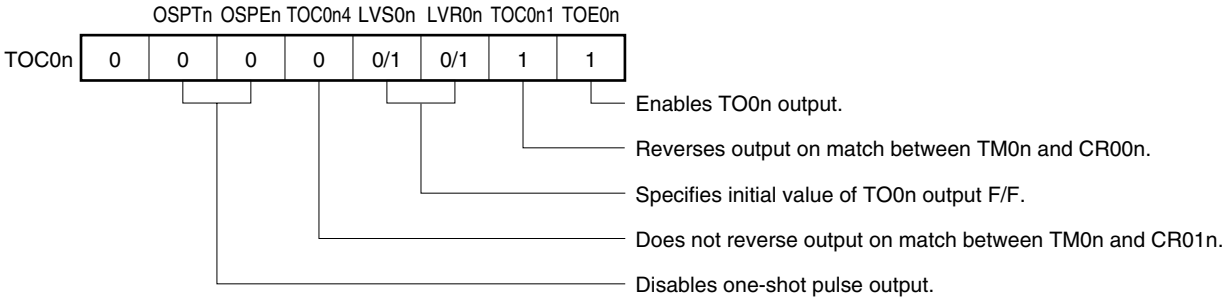
(a) 16-bit timer mode control register 0n (TMC0n)



(b) Capture/compare control register 0n (CRC0n)



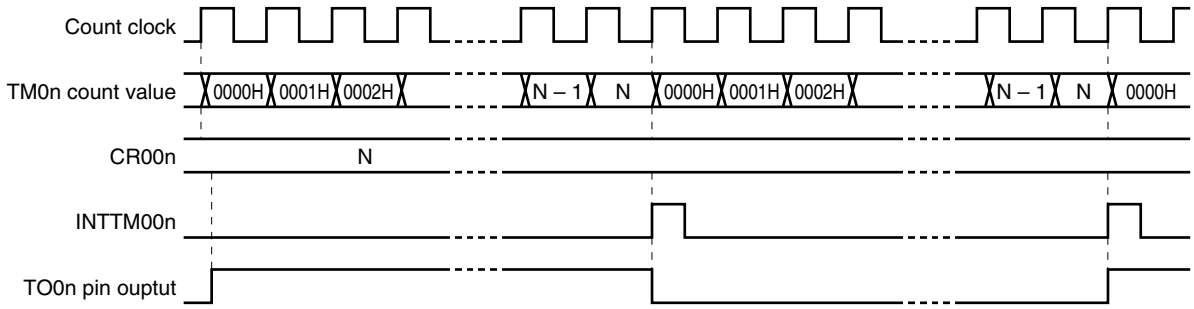
(c) 16-bit timer output control register 0n (TOC0n)



Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with square-wave output. See Figures 6-3 to 6-8.

2. n = 0, 1

Figure 6-31. Timing of Square-Wave Output Operation



Remark n = 0, 1

6.5.6 One-shot pulse output operation

A one-shot pulse synchronized with a software trigger or external trigger (pin TI00n input) can be output.

(1) One-shot pulse output by software trigger

If 16-bit timer mode control register 0n (TMC0n), capture/compare control register 0n (CRC0n), and 16-bit timer output control register 0n (TOC0n) are set as shown in Figure 6-32, and bit 6 (OSPTn) of TOC0n is set to 1 by software, a one-shot pulse is output from the TO0n pin.

By setting OSPTn to 1, the 16-bit timer/event counter 0n is cleared and started, and output becomes active at the count value set beforehand in 16-bit capture/compare register 01n (CR01n). Thereafter, output becomes inactive at the count value set beforehand in 16-bit capture/compare register 00n (CR00n).

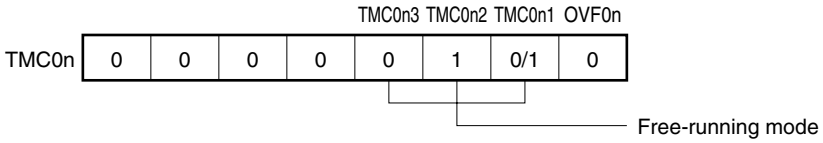
TM0n continues to operate after a one-shot pulse is output. To stop TM0n, 00H must be set to TMC0n.

Caution While a one-shot pulse is being output, do not set OSPTn (bit 6 of 16-bit timer output control register 0n (TOC0n)) to 1. To output a one-shot pulse again, perform it after an interrupt (INTTM00n) that matches CR00n is generated (n = 0, 1).

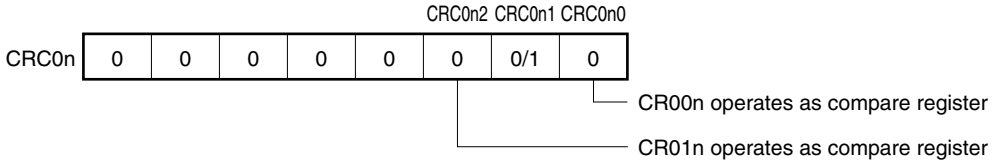
Remark n = 0, 1

Figure 6-32. Control Register Settings for One-Shot Pulse Output Operation Using Software Trigger

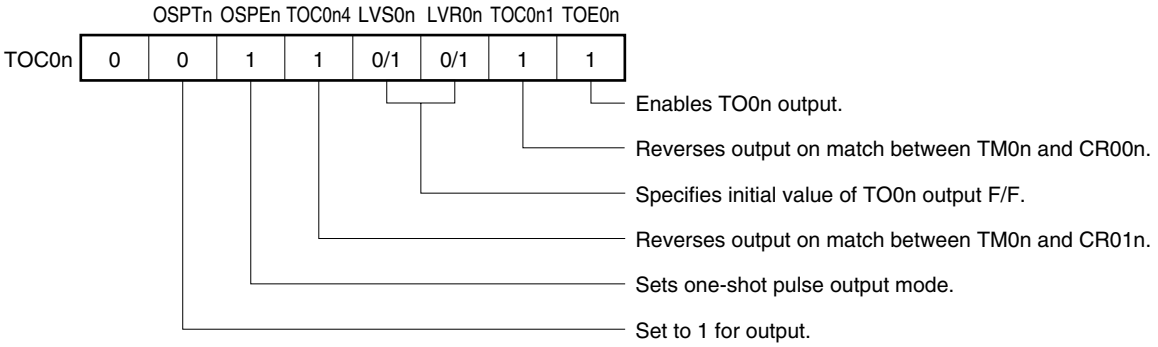
(a) 16-bit timer mode control register 0n (TMC0n)



(b) Capture/compare control register 0n (CRC0n)



(c) 16-bit timer output control register 0n (TOC0n)

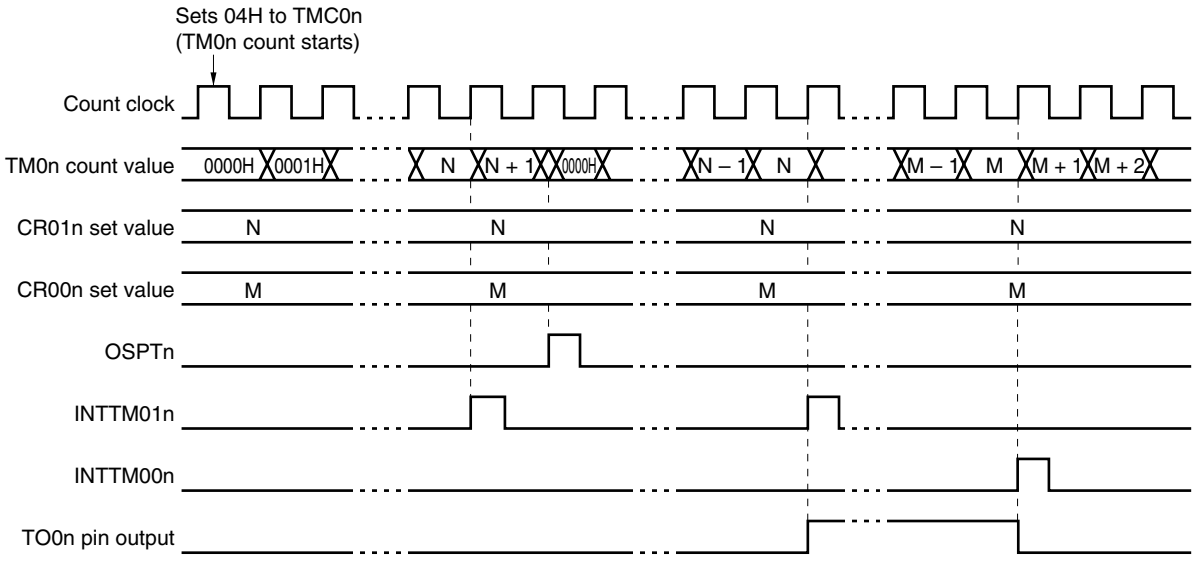


Caution Values in the following range should be set in CR00n and CR01n.
0000H < CR01n < CR00n ≤ FFFFH

Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with one-shot pulse output. See Figures 6-3 to 6-8.

2. n = 0, 1

Figure 6-33. Timing of One-Shot Pulse Output Operation Using Software Trigger



Caution 16-bit timer/counter 0n (TM0n) starts operation the moment a value other than 0, 0 (operation stop mode) is set to TMC0n2 and TMC0n3, respectively.

Remark n = 0, 1

(2) One-shot pulse output by external trigger

This sets 16-bit timer mode control register 0n (TMC0n), capture/compare control register 0n (CRC0n), and 16-bit timer output control register 0n (TOC0n) as shown in Figure 6-34 and outputs a one-shot pulse from the TO0n pin using the valid edge of the TI00n pin as an external trigger.

The TI00n pin's valid edge is specified by bits 4 and 5 (ES00n, ES01n) of prescaler mode register 0n (PRM0n) and is selected from rising, falling, and both rising and falling edges.

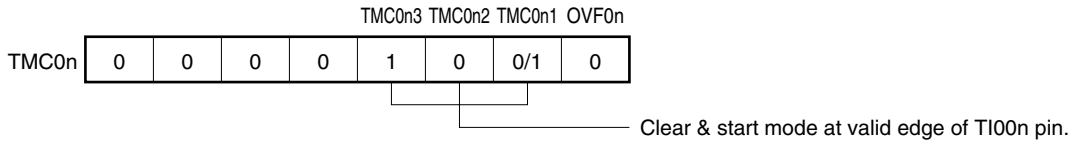
At the valid edge to the TI00n pin, the 16-bit timer/event counter clears and starts and the output becomes active at the count value set previously in 16-bit capture/compare register 01n (CR01n). After that, the output becomes inactive at the count value set previously in 16-bit capture/compare register 00n (CR00n).

Caution When a one-shot pulse is being output, even if another external trigger is generated, it is disregarded.

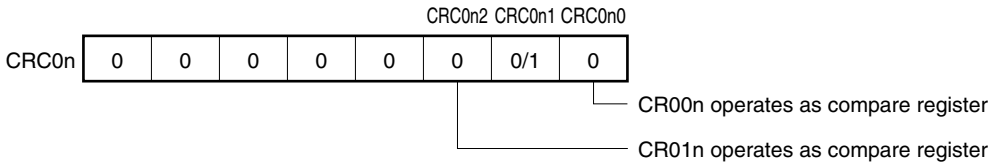
Remark n = 0, 1

Figure 6-34. Control Register Settings for One-Shot Pulse Output Operation Using External Trigger

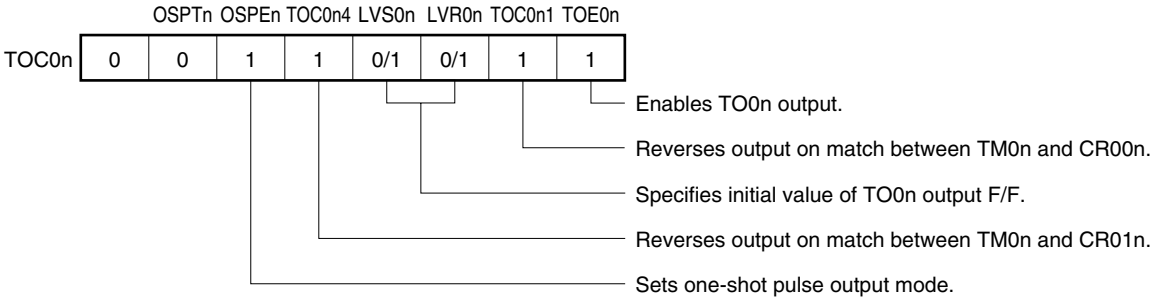
(a) 16-bit timer mode control register 0n (TMC0n)



(b) Capture/compare control register 0n (CRC0n)



(c) 16-bit timer output control register 0n (TOC0n)

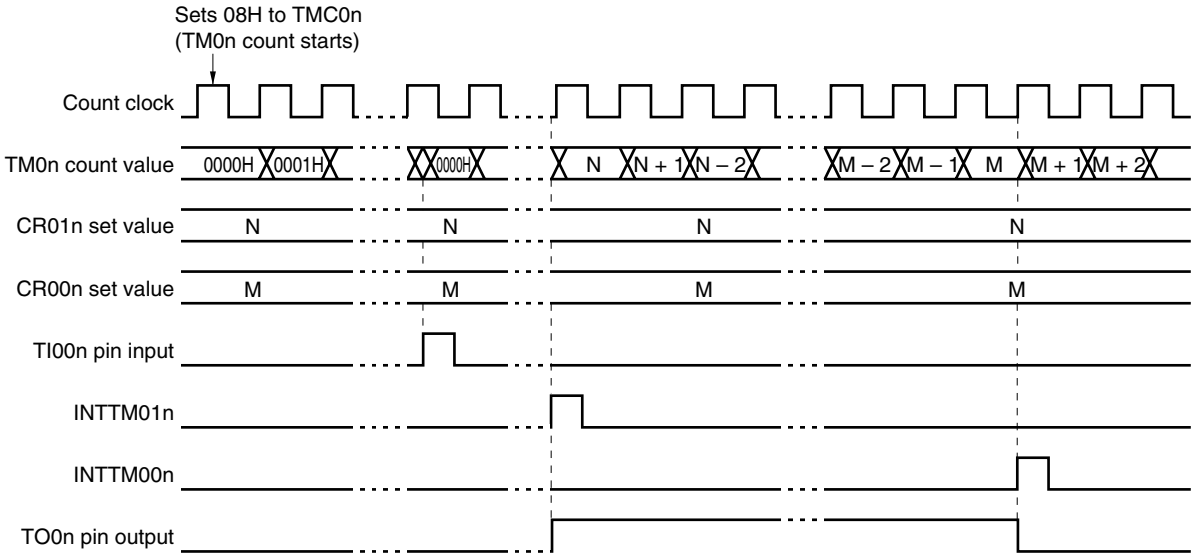


Caution Values in the following range should be set in CR00n and CR01n.
0000H < CR01n < CR00n ≤ FFFFH

Remarks 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with one-shot pulse output. See Figures 6-3 to 6-8.

2. n = 0, 1

Figure 6-35. Timing of One-Shot Pulse Output Operation Using External Trigger (Clear and start by the valid edge of TI00n, when the rising edge is specified)



Caution 16-bit timer/counter 0n (TM0n) starts operation the moment a value other than 0, 0 (operation stop mode) is set to TMC0n2 and TMC0n3, respectively.

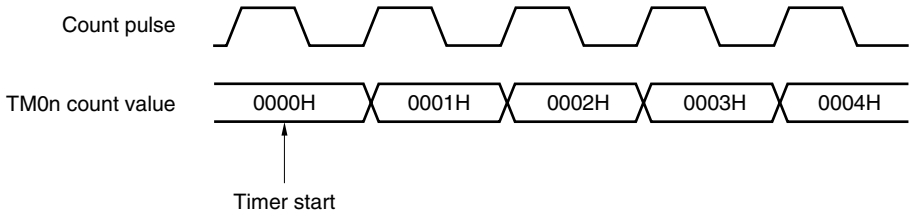
Remark n = 0, 1

6.6 Cautions for 16-Bit Timer/Event Counter

(1) Timer start errors

An error of up to one clock occurs after the timer has been started until a match signal is generated. This is because 16-bit timer/counter 0n (TM0n: n = 0, 1) is started asynchronously with the count pulse.

Figure 6-36. Timing of 16-Bit Timer/Counter Start



Remark n = 0, 1

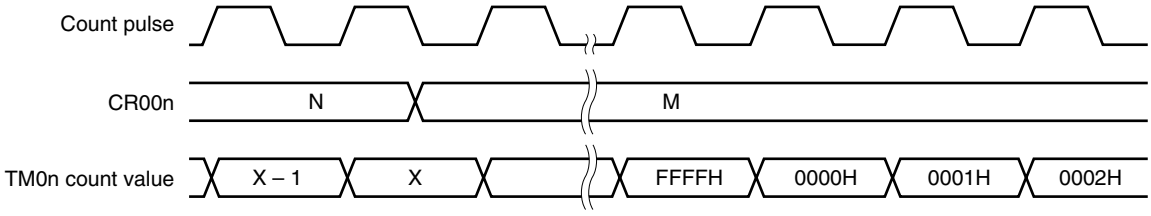
(2) 16-bit compare register setting

Set a value other than 0000H to 16-bit capture/compare registers 00n and 01n (CR00n, CR01n: n = 0, 1). This means a 1-pulse count operation cannot be performed when they are used as event counters.

(3) Operation after changing value of compare register during timer count operation

If the value after 16-bit capture/compare register 00n (CR00n: n = 0, 1) is changed is smaller than that of the 16-bit timer/counter 0n (TM0n: n = 0, 1), TM0n continues counting, overflows, and then restarts counting from 0. Therefore, if the value (M) after CR00n is changed is smaller than that (N) before the change, it is necessary to restart the timer after changing CR00n.

Figure 6-37. Timing after Changing Values of Compare Register during Timer Count Operation



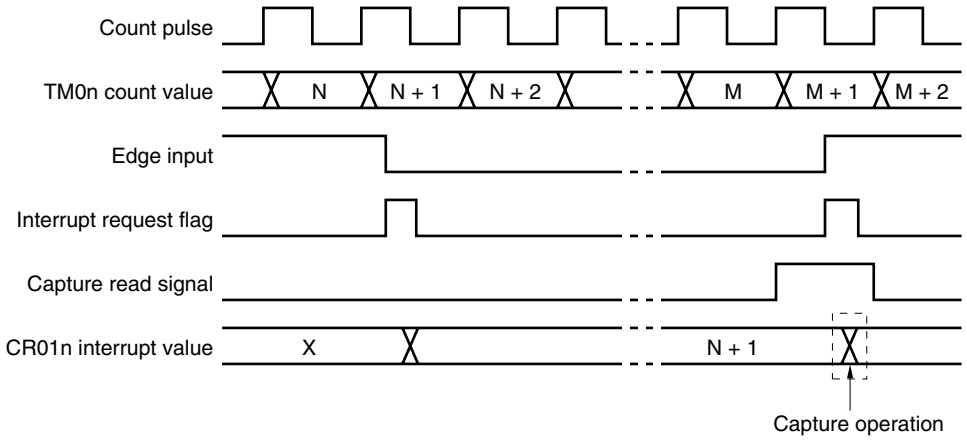
Remark N > X > M
n = 0, 1

(4) Capture register data retention timing

If the valid edge of the TI00n pin is input during 16-bit capture/compare register 01n (CR01n) read, CR01n performs a capture operation, but the capture value at this time is not guaranteed. However, the interrupt request flag (TMIF01n) upon detection of the valid edge is set.

Remark n = 0, 1

Figure 6-38. Timing of Capture Register Data Retention



Remark n = 0, 1

(5) Valid edge setting

Set the valid edge of the TI00n pin after setting bits 2 and 3 (TMC0n2 and TMC0n3) of 16-bit timer mode control register 0n (TMC0n) to 0, 0, respectively, and then stopping timer operation. The valid edge is set with bits 4 and 5 (ES0n0 and ES0n1) of prescaler mode register 0n (PRM0n).

Remark n = 0, 1

(6) Re-trigger of one-shot pulse

(a) One-shot pulse output by software

While a one-shot pulse is being output, do not set OSPTn (bit 6 of 16-bit timer output control register 0n (TOC0n)) to 1. When outputting a one-shot pulse again, output it after INTTM00n, which is the interrupt request match signal with CR00n, is generated.

(b) One-shot pulse output by external trigger

When a one-shot pulse is being output, even if another external trigger is generated, it is disregarded.

Remark n = 0, 1

(7) Operation of OVF0n flag

<1> The OVF0n flag (bit 6 of 16-bit timer mode control register 0n (TMC0n)) is set to “1” in the following case.

The clear & start mode on match between TM0n and CR00n is selected.

↓

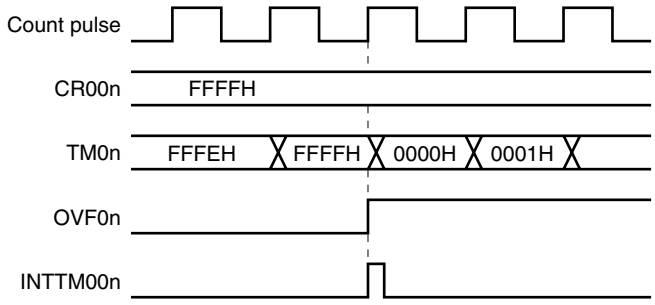
CR00n is set to FFFFH.

↓

When TM0n is counted up from FFFFH to 0000H.

Remark n = 0, 1

Figure 6-39. Operation Timing of OVF0n Flag



Remark n = 0, 1

<2> After TM0n overflows, even if the OVF0n flag is cleared before the next count clock (before TM0n becomes 0001H), it is reset and the clear instruction becomes invalid.

Remark n = 0, 1

(8) Contending operations

<1> Contention between the read period of 16-bit capture/compare registers 00n, 01n (CR00n, CR01n) and capture trigger input (CR00n and CR01n used as capture register)

Capture trigger input has priority. The data read from CR00n and CR01n are not defined.

<2> Contention between the write period of 16-bit capture/compare register 00n or 01n (CR00n, CR01n) and the match timing of CR00n or CR01n and 16-bit timer/counter 0n (TM0n) (CR00n or CR01n used as compare register)

The match discriminant is not performed normally. Do not write any data to CR00n and CR01n near the match timing.

(9) Timer operation

<1> Even if 16-bit timer/counter 0n (TM0n) is read, the value is not captured in 16-bit capture/compare register 01n (CR01n).

<2> The sampling clock for count clock TI00n is different to the one for capture trigger TI00n. The former is $f_X/2^3$ and the latter is a count clock (see Figures 6-9 and 6-10.)

<3> Regardless of the operating mode of the CPU, if the timer is stopped, the noise of the external interrupt request input is not removed.

<4> The one-shot pulse output operates normally only in the free-running mode or in the clear & start mode at the TI00n valid edge. In the clear & start mode on match between TM0n and CR00n, there is no overflow, so one-shot pulse output is not possible.

Remark n = 0, 1

(10) Capture operation

<1> When the valid edge of TI00n (n = 0, 1) is specified for the count clock, the capture register that specified TI00n as the trigger cannot perform the capture operation normally.

<2> The capture operation is performed at the rise of the count clock, but the external interrupt signal (INTTM00n, INTTM01n) is set at the rise of the next count clock.

(11) Compare operation

<1> When 16-bit capture/compare registers 00n and 01n (CR00n, CR01n) are rewritten during timer operation, if the values are close to and greater than the timer values, there is a possibility that the match interrupt will not be generated, or the clear operation will not be performed properly.

<2> When CR00n and CR01n have been set in the compare mode, they cannot perform the capture operation even if the capture trigger is input.

Remark n = 0, 1

(12) Edge detection

When the TI00n pin or TI01n pin are high level immediately after system reset, and if the rising edge or both edges are specified as the valid edge of the TI00n pin or TI01n pin, then the rising edge is detected immediately after operation of 16-bit timer/counter 0n (TM0n) is enabled. Be careful when the TI00n pin or TI01n pin are pulled up. When operation is enabled again after once being stopped, the rising edge cannot be detected.

Remark n = 0, 1

CHAPTER 7 8-BIT TIMER/EVENT COUNTER

7.1 Function of 8-Bit Timer/Event Counters

The 8-bit timer/event counters (TM50, TM51, and TM52) have the following two modes:

- Mode in which one 8-bit timer/event counter is used alone (single mode)
- Mode in which two 8-bit timer/event counters (TM50 and TM51) are connected in cascade (16-bit resolution: cascade mode)

These two modes are explained below.

(1) Mode in which one 8-bit timer/event counter is used alone (single mode)

In this mode, the counter can be used as the following.

- Interval timer
- External event counter
- Square-wave output
- PWM output

(2) Mode in which TM50 and TM51 are connected in cascade (16-bit resolution: cascade mode)

By connecting two 8-bit timer/event counters in cascade, they can be used as a 16-bit timer/event counter. In the cascade mode, the counters can be used as the following.

- 16-bit resolution interval timer
- 16-bit resolution external event counter
- 16-bit resolution square-wave output

Caution When connecting TM50 and TM51 in cascade, use TM50 as a lower timer and TM51 as a higher timer. TM52 cannot be connected in cascade.

7.2 Configuration of 8-Bit Timer/Event Counter

An 8-bit timer/event counter consists of the following hardware:

Table 7-1. Configuration of 8-Bit Timer/Event Counter

Item	Configuration
Timer register	8-bit timer/counter 5n (TM5n)
Register	8-bit timer compare register 5n (CR5n)
Timer output	TO5n
Control register	8-bit timer mode control register 5n (TMC5n) Timer clock select register 5n (TCL5n) Port mode register 7 (PM7)[Note]

Note See **Figure 4-10 Block Diagram of P70, P73 to P77**.

Remark n = 0 to 2

Figure 7-1. Block Diagram of 8-Bit Timer/Event Counter 50 (TM50)

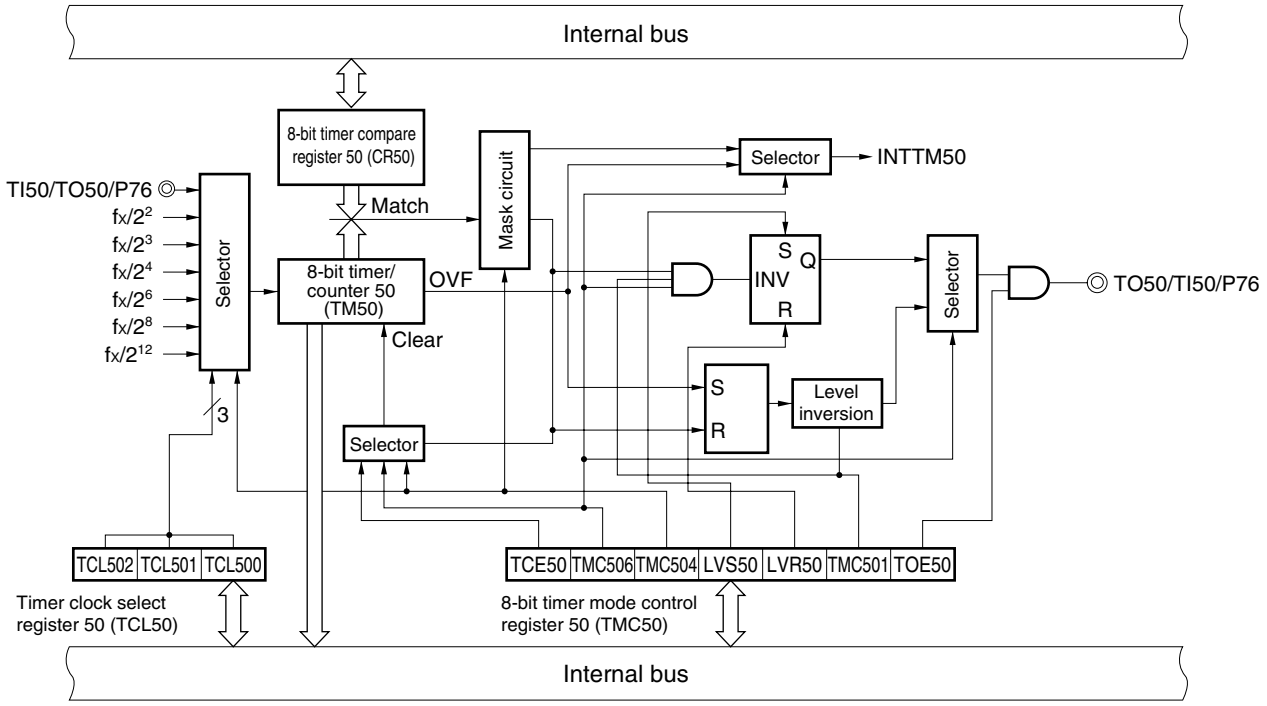


Figure 7-2. Block Diagram of 8-Bit Timer/Event Counter 51 (TM51)

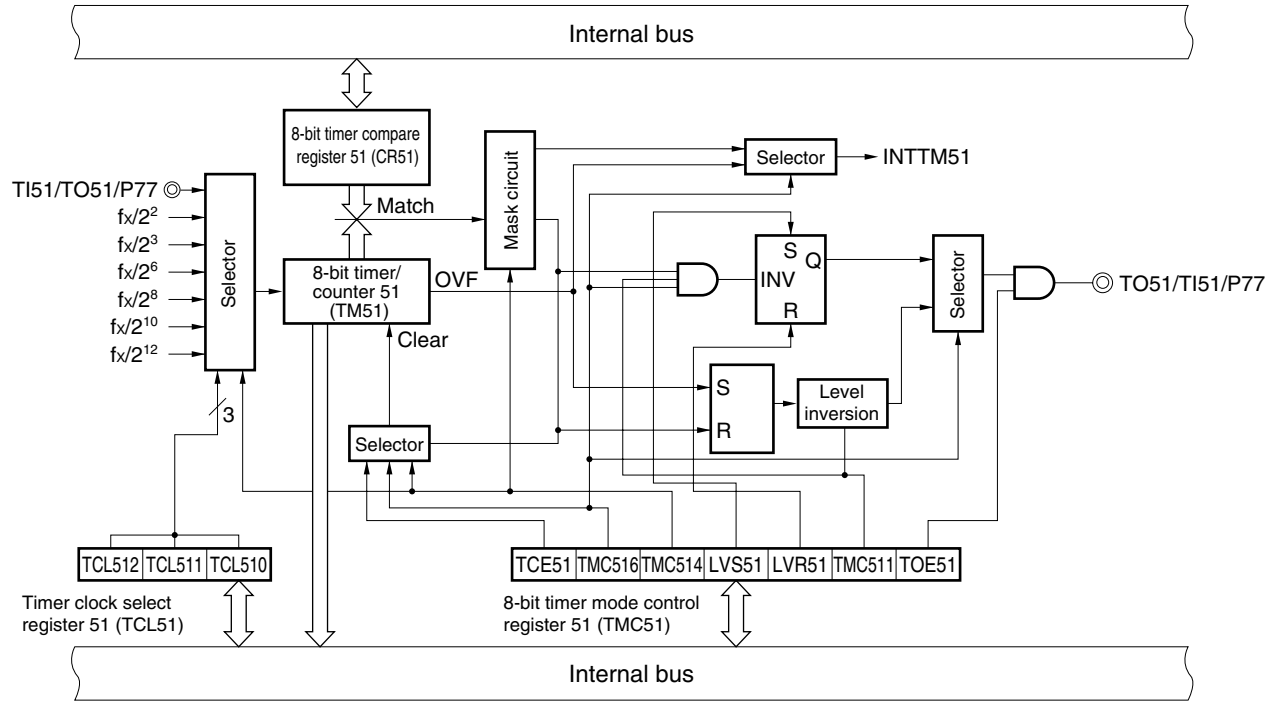
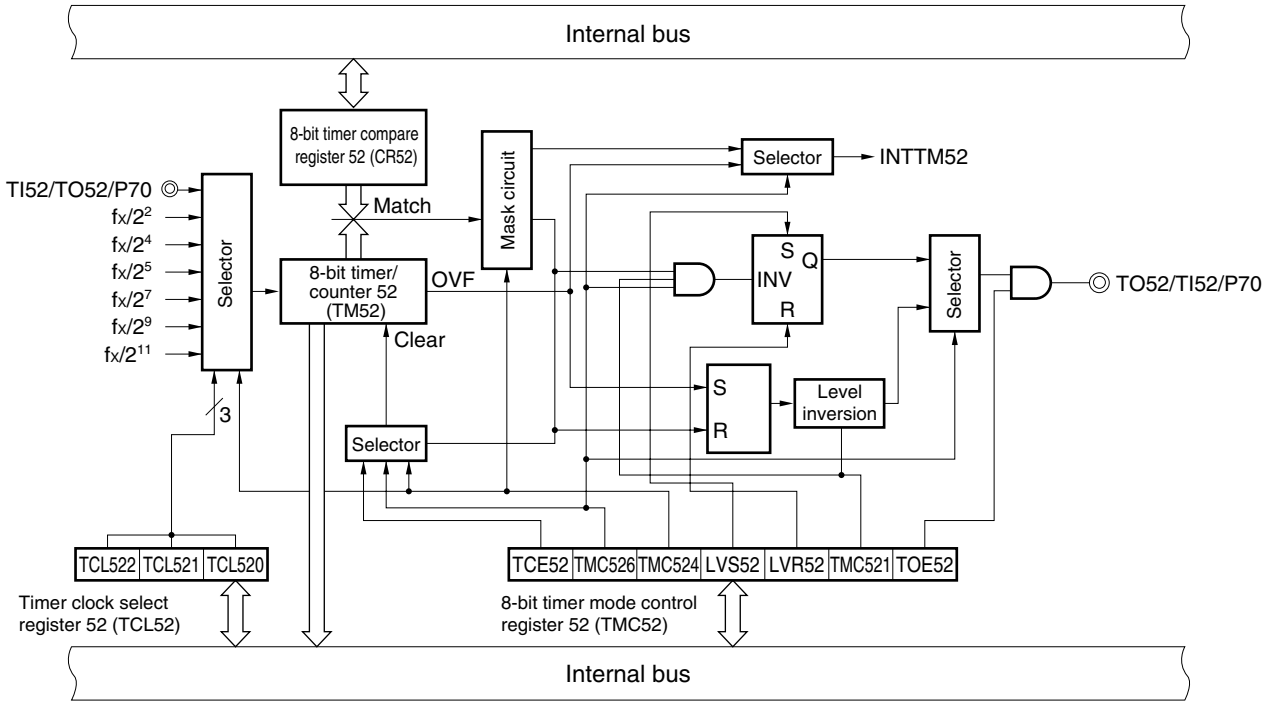


Figure 7-3. Block Diagram of 8-Bit Timer/Event Counter 52 (TM52)



(1) 8-bit timer/counters 50, 51, and 52 (TM50, TM51, and TM52)

TM50, TM51, and TM52 are 8-bit read-only registers that count count pulses.

These counters are incremented in synchronization with the rising edge of the count clock.

TM50 and TM51 can be connected in cascade and can be used as a 16-bit timer.

When TM50 and TM51 are connected in cascade and used as a 16-bit timer, the values of these timers/counters can be read by using a 16-bit manipulation instruction. TM50 and TM51 are connected with an internal 8-bit bus, which means that TM50 and TM51 are read one at a time. Therefore because the value of one may change while the other is being read, for accuracy be sure to read TM50 and TM51 twice to compare their first and second values.

If a count value is read during operation, input of the count clock is temporarily stopped, and the count value at that point is read. The count value is cleared to 00H in the following cases:

<1> When $\overline{\text{RESET}}$ is input

<2> When TCE5n is cleared

<3> When there is match between TM5n and CR5n in clear & start mode on match between TM5n and CR5n.

Caution In the cascade mode, the 16-bit timer is cleared to 00H even if bit TCE50 of TM50 is cleared.

Remark n = 0 to 2

(2) 8-bit compare registers 50, 51, and 52 (CR50, CR51, and CR52)

The value set to CR5n is always compared with the count value of 8-bit timer/counter 5n (TM5n). When the value of the compare register matches with the value of the timer/counter, an interrupt request (INTTM5n) is generated (in a mode other than the PWM mode).

The value of CR5n can be set in a range of 00H to FFH and can be rewritten during counting.

If TM50 and TM51 are connected in cascade and used as a 16-bit timer, CR50 and CR51 operate as a 16-bit compare register. Therefore, the count value and register value are compared in 16-bit units, and if the two values match each other, an interrupt request (INTTM50) is generated. At this time, interrupt request INTTM51 is also generated. When connecting TM50 and TM51 in cascade, therefore, mask the INTTM51 interrupt request.

Caution When setting data to the timers connected in cascade, be sure to stop the timer operation.

Remark n = 0 to 2

7.3 Registers Controlling 8-Bit Timer/Event Counters

The following three registers control 8-bit timer/event counters 50, 51, and 52.

- 8-bit timer mode control registers 50, 51, and 52 (TMC50, TMC51, and TMC52)
- Timer clock select registers 50, 51, and 52 (TCL50, TCL51, and TCL52)
- Port mode register 7 (PM7)

(1) 8-bit timer mode control registers 50, 51, and 52 (TMC50, TMC51, and TMC52)

TMC50, TMC51, and TMC52 perform the following six setting operations:

<1> Control the count operation of 8-bit timer/counters 50, 51, and 52 (TM50, TM51, and TM52)
<2> Select the operation mode of 8-bit timer/counters 50, 51, and 52 (TM50, TM51, and TM52)
<3> Select single mode or cascade mode
<4> Set the status of timer output F/F (flip-flop)
<5> Control the timer F/F or select the active level in PWM (free running) mode
<6> Control the timer output

TMC50, TMC51, and TMC52 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets these registers to 04H.

Figures 7-4 to 7-6 show the formats of TMC50, TMC51, and TMC52.

Caution The initial value of TMC5n (n = 0 to 2) is 04H. When TMC5n is read, the value becomes 00H. This is because bits 2 and 3 (LVR5n and LVS5n) of TMC5n are write-only bits (read = 0).

Figure 7-4. Format of 8-Bit Timer Mode Control Register 50 (TMC50)

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

Address	After reset	R/W
FF79H	04H	R/W

TCE50	TM50 count operation control
0	Disables count operation after clearing counter to 0 (disables prescaler).
1	Starts counting.

TMC506	TM50 operating mode selection
0	Clear & start mode on match between TM50 and CR50.
1	PWM (free running) mode

LVS50	LVR50	Timer output F/F status setting of 8-bit timer/event counter 50
0	0	Not affected
0	1	Resets timer output F/F (to 0).
1	0	Sets timer output F/F (to 1).
1	1	Setting prohibited

TMC501	Other than PWM mode (TMC506 = 0)	PWM mode (TMC506 = 1)
	Timer F/F control	Active level selection
0	Disables inverted operation.	High active
1	Enables inverted operation.	Low active

TOE50	Timer output control of 8-bit timer/event counter 50
0	Disables output (port mode).
1	Enables output.

Remarks 1. The PWM output becomes inactive when TCE50 is set to 0 in the PWM mode.

2. If LVS50 and LVR50 are read immediately after data has been set, these bits are 0.

Figure 7-5. Format of 8-Bit Timer Mode Control Register 51 (TMC51)

Symbol	7	6	5	4	3	2	1	0
TMC51	TCE51	TMC516	0	TMC514	LVS51	LVR51	TMC511	TOE51

Address	After reset	R/W
FF7BH	04H	R/W

TCE51	TM51 count operation control
0	Disables count operation after clearing counter to 0 (disables prescaler).
1	Starts counting.

TMC516	TM51 operating mode selection
0	Clear & start mode on match between TM51 and CR51.
1	PWM (free running) mode

TMC514	Single mode/cascade mode selection
0	Single mode
1	Cascade mode (connected to TM50)

LVS51	LVR51	Timer output F/F status setting of 8-bit timer/event counter 51
0	0	Not affected
0	1	Resets timer output F/F (to 0).
1	0	Sets timer output F/F (to 1).
1	1	Setting prohibited

TMC511	Other than PWM mode (TMC516 = 0)	PWM mode (TMC516 = 1)
	Timer F/F control	Active level selection
0	Disables inverted operation.	High active
1	Enables inverted operation.	Low active

TOE51	Timer output control of 8-bit timer/event counter 51
0	Disables output (port mode).
1	Enables output.

Remarks 1. The PWM output becomes inactive when TCE51 is set to 0 in the PWM mode.

2. If LVS51 and LVR51 are read immediately after data has been set, these bits are 0.

Figure 7-6. Format of 8-Bit Timer Mode Control Register 52 (TMC52)

Symbol	7	6	5	4	3	2	1	0
TMC52	TCE52	TMC526	0	0	LVS52	LVR52	TMC521	TOE52

Address	After reset	R/W
FF7DH	04H	R/W

TCE52	TM52 count operation control
0	Disables count operation after clearing counter to 0 (disables prescaler).
1	Starts counting.

TMC526	TM52 operating mode selection
0	Clear & start mode on match between TM52 and CR52.
1	PWM (free running) mode

LVS52	LVR52	Timer output F/F status setting of 8-bit timer/event counter 52
0	0	Not affected
0	1	Resets timer output F/F (to 0).
1	0	Sets timer output F/F (to 1).
1	1	Setting prohibited

TMC521	Other than PWM mode (TMC526 = 0)	PWM mode (TMC526 = 1)
	Timer F/F control	Active level selection
0	Disables inverted operation.	High active
1	Enables inverted operation.	Low active

TOE52	Timer output control of 8-bit timer/event counter 52
0	Disables output (port mode).
1	Enables output.

Remarks 1. The PWM output becomes inactive when TCE52 is set to 0 in the PWM mode.

2. If LVS52 and LVR52 are read immediately after data has been set, these bits are 0.

(2) Timer clock select registers 50, 51, and 52 (TCL50, TCL51, and TCL52)

These registers specify the count clock of 8-bit timer/counters 50, 51, and 52 (TM50, TM51, and TM52) and the valid edges of TI50, TI51, and TI52 inputs.

TCL50, TCL51, and TCL52 are set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets these registers to 00H.

Figures 7-7 to 7-9 show the formats of TCL50, TCL51, and TCL52.

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

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCL50	0	0	0	0	0	TCL502	TCL501	TCL500	FF78H	00H	R/W

TCL502	TCL501	TCL500	Count clock selection
0	0	0	Falling edge of TI50
0	0	1	Rising edge of TI50
0	1	0	$f_X/2^2$ (1.57 MHz)
0	1	1	$f_X/2^3$ (786 kHz)
1	0	0	$f_X/2^4$ (393 kHz)
1	0	1	$f_X/2^6$ (98.3 kHz)
1	1	0	$f_X/2^8$ (24.6 kHz)
1	1	1	$f_X/2^{12}$ (1.54 kHz)

Cautions 1. Timer operation must be stopped before rewriting TCL50 to a value other than the identical data.

2. Be sure to set bits 3 to 7 of TCL50 to 0.

Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

Figure 7-8. Format of Timer Clock Select Register 51 (TCL51)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCL51	0	0	0	0	0	TCL512	TCL511	TCL510	FF7AH	00H	R/W

TCL512	TCL511	TCL510	Count clock selection
0	0	0	Falling edge of TI51
0	0	1	Rising edge of TI51
0	1	0	$f_X/2^2$ (1.57 MHz)
0	1	1	$f_X/2^3$ (786 kHz)
1	0	0	$f_X/2^6$ (98.3 kHz)
1	0	1	$f_X/2^8$ (24.6 kHz)
1	1	0	$f_X/2^{10}$ (6.14 kHz)
1	1	1	$f_X/2^{12}$ (1.54 kHz)

Cautions 1. Timer operation must be stopped before rewriting TCL51 to a value other than the identical data.

2. Be sure to set bits 3 to 7 of TCL51 to 0.

Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

3. The settings of TCL510 to TCL512 are invalid when TM50 and TM51 are connected in cascade.

Figure 7-9. Format of Timer Clock Select Register 52 (TCL52)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCL52	0	0	0	0	0	TCL522	TCL521	TCL520	FF7CH	00H	R/W

TCL522	TCL521	TCL520	Count clock selection
0	0	0	Falling edge of TI52
0	0	1	Rising edge of TI52
0	1	0	$f_X/2^2$ (1.57 MHz)
0	1	1	$f_X/2^4$ (393 kHz)
1	0	0	$f_X/2^5$ (197 kHz)
1	0	1	$f_X/2^7$ (49.2 kHz)
1	1	0	$f_X/2^9$ (12.3 kHz)
1	1	1	$f_X/2^{11}$ (3.07 kHz)

Cautions 1. Timer operation must be stopped before rewriting TCL52 to a value other than the identical data.

2. Be sure to set bits 3 to 7 of TCL52 to 0.

Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

(3) Port mode register 7 (PM7)

This register sets port 7 to the input or output mode in 1-bit units.

When pins P70/TI52/TO52, P76/TI50/TO50, and P77/TI51/TO51 are used for timer output, clear PM70, PM76, and PM77 and the output latches of P70, P76, and P77 to 0.

PM7 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to FFH.

Figure 7-10. Format of Port Mode Register 7 (PM7)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	FF27H	FFH	R/W

PM7n	P7n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

7.4 Operation of 8-Bit Timer/Event Counter

7.4.1 Interval timer (8-bit) operation

The 8-bit timer/event counters operate as interval timers that repeatedly generate an interrupt request at time intervals specified by the count values set to the corresponding 8-bit timer/compare register 5n (CR5n) in advance.

When the count values of the 8-bit timer/counter 5n (TM5n) match the values set to corresponding compare register CR5n, the value of TM5n is cleared to 0, TM5n continues counting, and at the same time, an interrupt request signal (INTTM5n) is generated.

The count clock of TM5n can be selected by bits 0 to 2 (TCL5n0 to TCL5n2) of timer clock select register 5n (TCL5n).

For operation when the value of the compare register is changed during timer count operation, refer to **7.5 Notes on 8-Bit Timer/Event Counter (2)**.

<Setting>

<1> Set each register.
- TCL5n: Selects count clock.
- CR5n: Compare value
- TMC5n: Selects clear & start mode on match between TM5n and CR5n (TMC5n = 0000×××0B × = don't care).

<2> The count operation is started when TCE5n is set to 1.

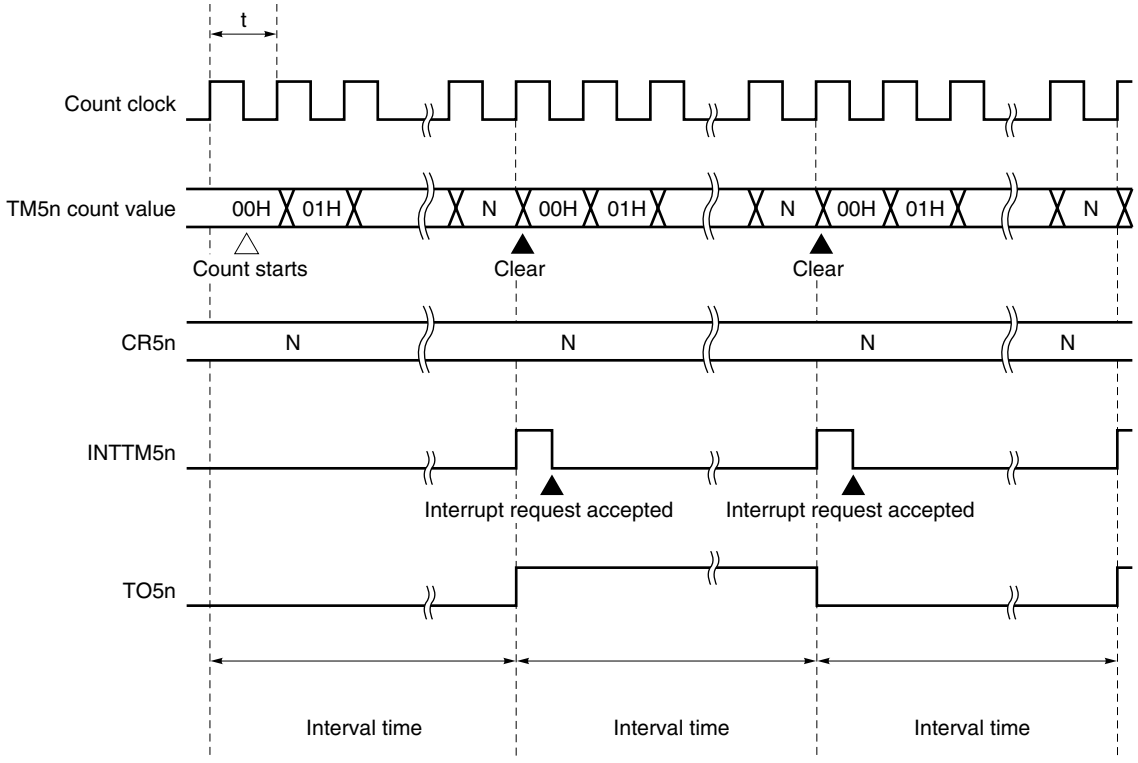
<3> INTTM5n occurs when the values of TM5n and CR5n match (TM5n is cleared to 00H).

<4> After that, INTTM5n repeatedly occurs at the same interval. To stop the count operation, clear TCE5n to 0.

Remark n = 0 to 2

Figure 7-11. Timing of Interval Timer Operation (1/3)

(a) Basic operation

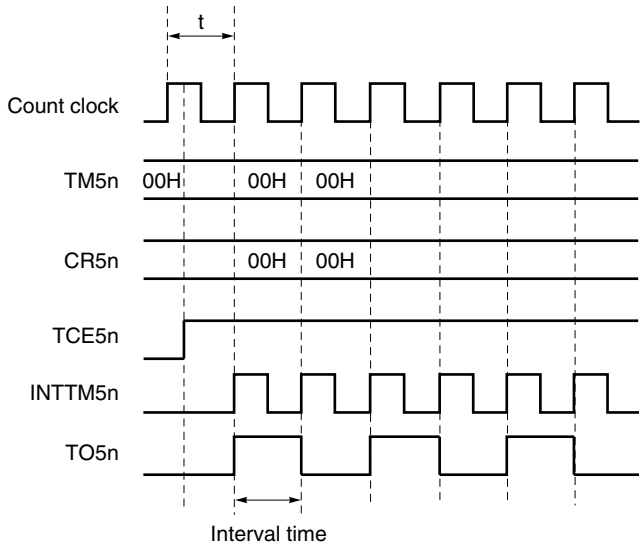


Remarks 1. Interval time = (N + 1) × t: N = 00H to FFH

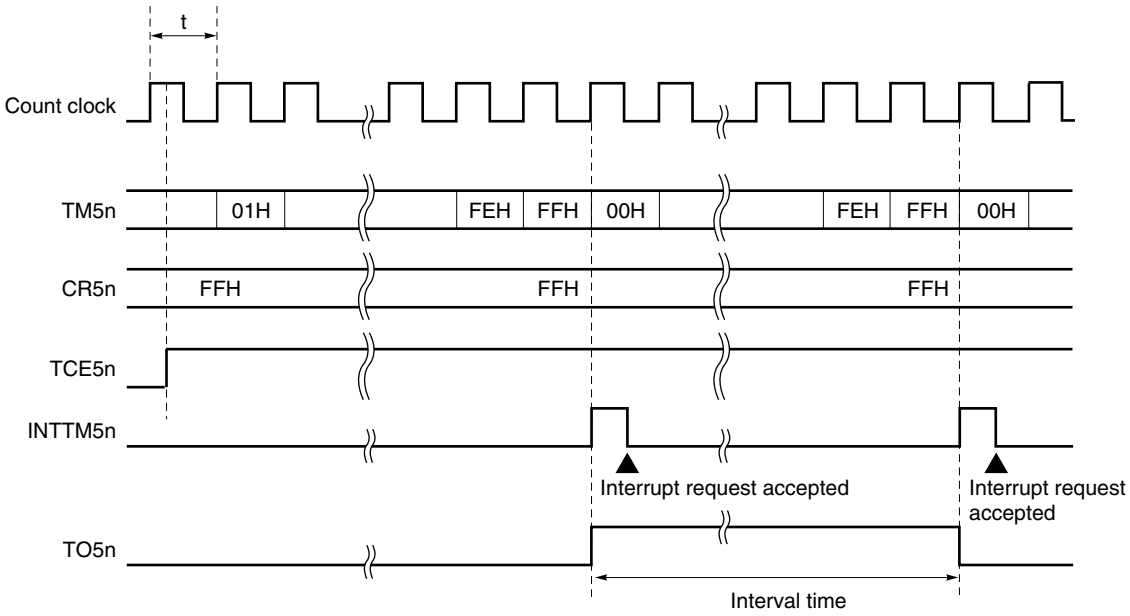
2. n = 0 to 2

Figure 7-11. Timing of Interval Timer Operation (2/3)

(b) When CR5n = 00H



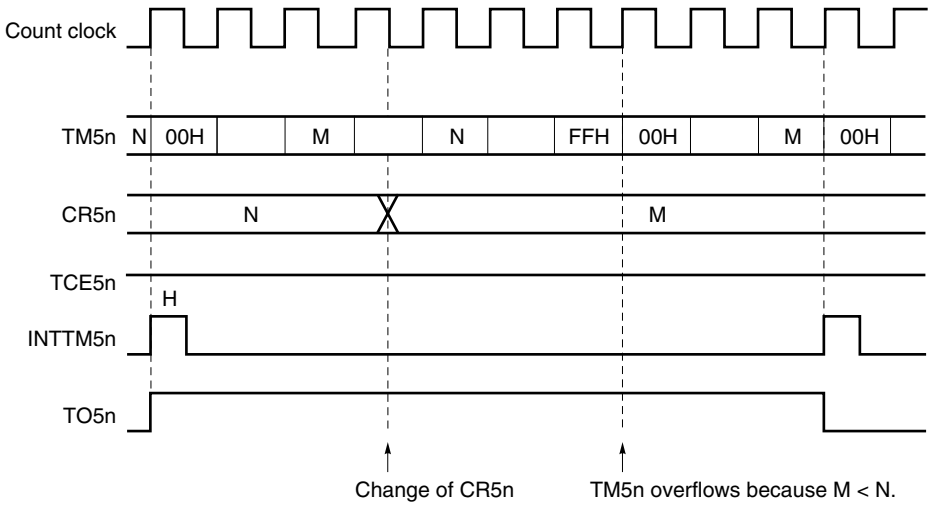
(c) When CR5n = FFH



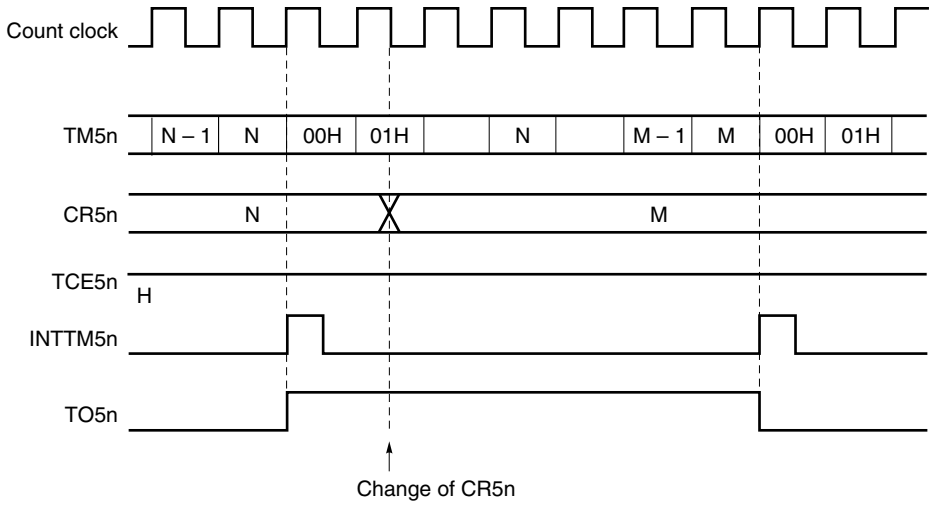
Remark n = 0 to 2

Figure 7-11. Timing of Interval Timer Operation (3/3)

(d) Operation when CR5n is changed (M < N)



(e) Operation when CR5n is changed (M > N)



Remark n = 0 to 2

7.4.2 External event counter operation

The external event counter counts the number of clock pulses externally input to pins TI50/TO50/P76, TI51/TO51/P77, and TI52/TO52/P70 by using 8-bit timer/counter 5n (TM5n).

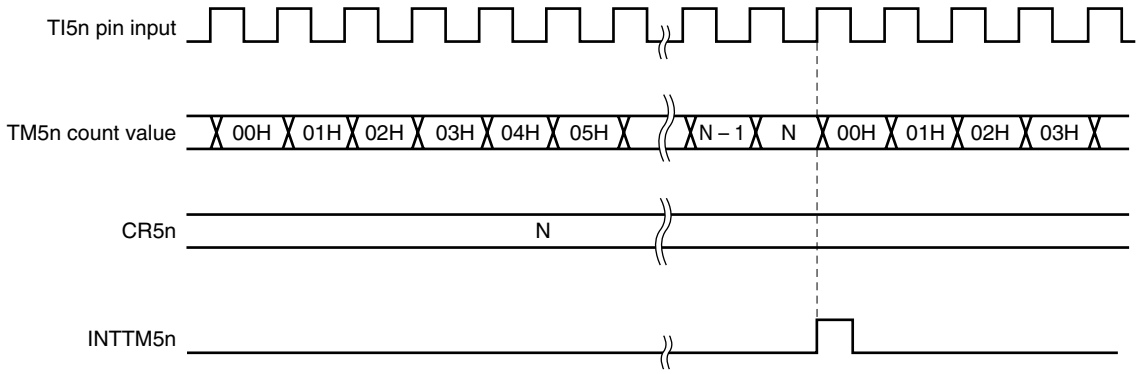
Each time the valid edge specified by timer clock select register 5n (TCL5n) is input, the value of TM5n is incremented. Either the rising or falling edge can be specified as the valid edge.

When the count value of TM5n matches the values of corresponding 8-bit timer compare register 5n (CR5n), TM5n is cleared to 0, and an interrupt request signal (INTTM5n) is generated.

Whenever the TM5n value matches the value of CR5n, INTTM5n is generated.

Remark n = 0 to 2

Figure 7-12. Timing of External Event Counter Operation (with Rising Edge Specified)



Remark N = 00H to FFH
n = 0 to 2

7.4.3 Square-wave output (8-bit resolution) operation

The 8-bit timer/event counter operates as a square wave output at an interval set in advance to 8-bit compare register 5n (CR5n).

When bit 0 (TOE5n) of 8-bit timer mode control register 5n (TMC5n) is set to 1, the output status of TO5n is inverted at the interval time specified by the count value set in advance to CR5n. In this way, a square wave of any frequency (duty factor = 50%) can be output.

<Setting>

<1> Set each register.

- Clear the port latch and port mode register 7 (PM7) to 0.
- TCL5n: Selects count clock
- CR5n: Compare value
- TMC5n: Clear & start mode on match between TM5n and CR5n

LVS5n	LVR5n	Timer output F/F status setting
1	0	High-level output
0	1	Low-level output

Enables inverting timer output F/F

Timer output enabled → TOE5n = 1

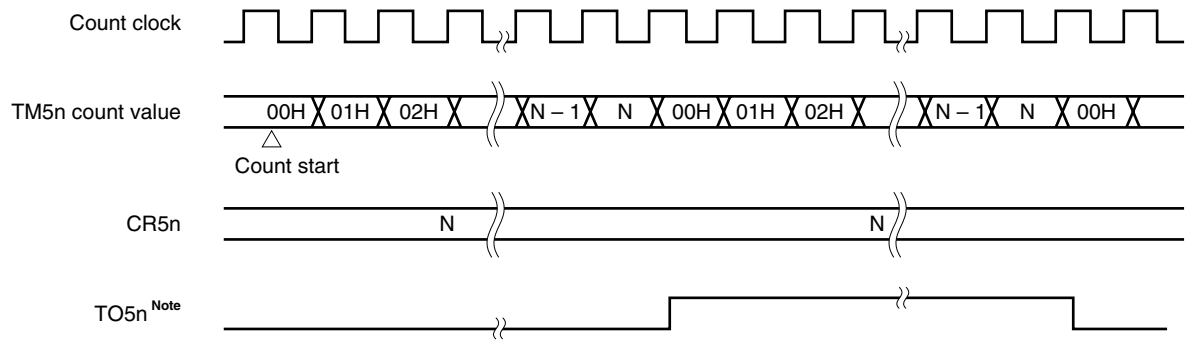
<2> The count operation is started if TCE5n is set to 1.

<3> The timer output F/F is inverted if the values of TM5n and CR5n match.
INTTM5n is generated and TM5n is cleared to 00H.

<4> After that, the timer output F/F is inverted at the same interval, and a square wave is output from TO5n.

Remark n = 0 to 2

Figure 7-13. Timing of Square-Wave Output Operation



Note The initial value of TO5n output can be set with bits 2 and 3 (LVR5n and LVS5n) of 8-bit timer mode control register 5n (TMC5n).

Remark n = 0 to 2

7.4.4 8-bit PWM output operation

The PWM output operation is performed when bit 6 (TMC5n6) of 8-bit timer mode control register 5n (TMC5n) is set to 1.

A pulse with a duty factor determined by the value set to 8-bit timer compare register 5n (CR5n) is output from TO5n.

Set the width of the active level of the PWM pulse to CR5n. The active level can be selected by bit 1 (TMC5n1) of TMC5n.

The count clock can be selected by bits 0 to 2 (TCL5n0 to TCL5n2) of timer clock select register 5n (TCL5n).

The PWM output can be enabled or disabled by bit 0 (TOE5n) of TMC5n.

Caution CR5n can only be rewritten once per cycle in PWM mode.

(1) Basic operation of PWM output

<Setting>

<1> Clear the port latch and port mode register 7 (PM7) to 0.
<2> Set an active-level width with 8-bit timer compare register 5n (CR5n).
<3> Select a count clock with timer clock select register 5n (TCL5n).
<4> Set an active level by using bit 1 (TMC5n1) of TMC5n.
<5> The count operation is started when bit 7 (TCE5n) of TMC5n is set to 1.
To stop the count operation, clear TCE5n to 0.

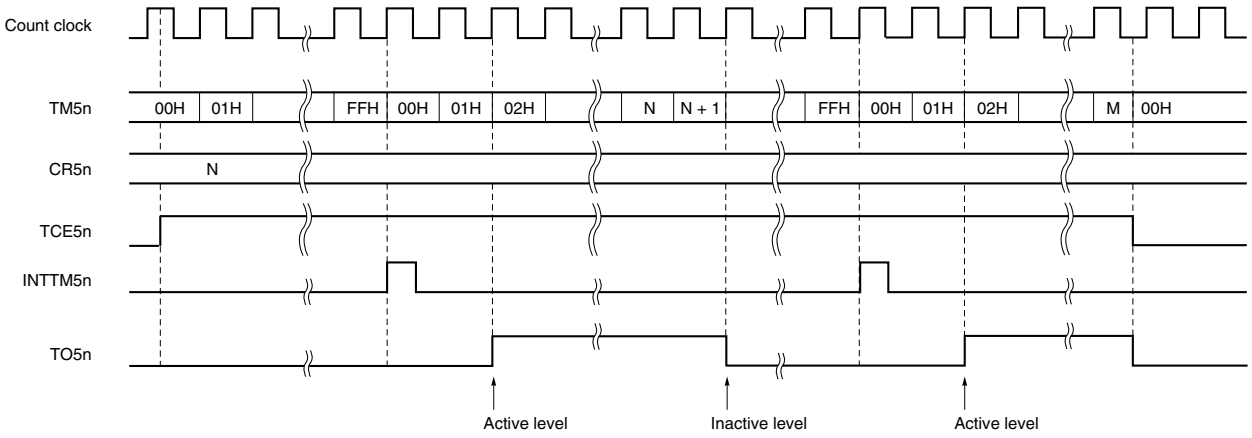
<PWM output operation>

<1> When the count operation is started, an inactive level is output as the PWM output (output from TO5n) until an overflow occurs.
<2> When an overflow occurs, the active level set in step <1> above is output. This active level is continuously output until the value of CR5n matches the count value of 8-bit timer/counter 5n (TM5n).
<3> An inactive level is output as the PWM output after the value of CR5n has matched the count value of TM5n, until an overflow occurs again.
<4> After that, <2> and <3> are repeated until the count operation is stopped.
<5> When the count operation is stopped by clearing TCE5n to 0, the PWM output becomes inactive.

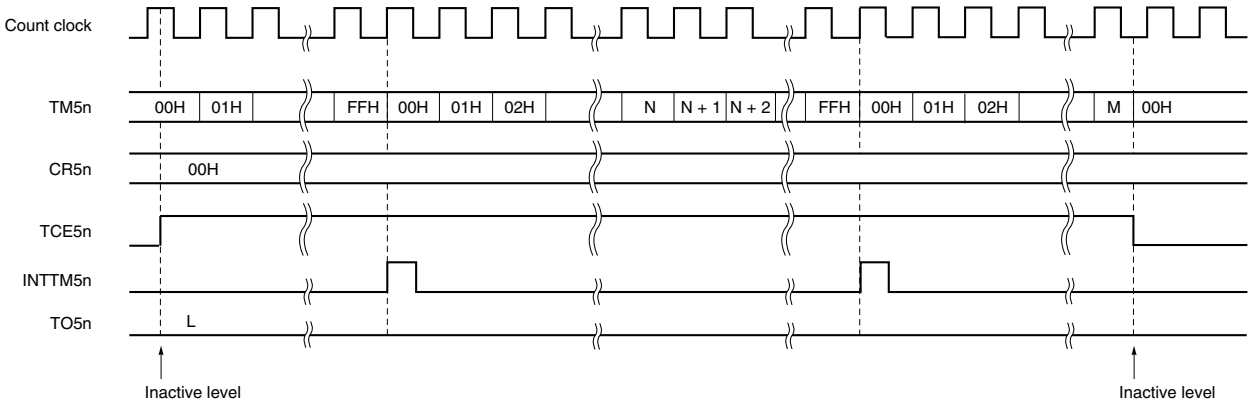
Remark n = 0 to 2

Figure 7-14. Operation Timing of PWM Output

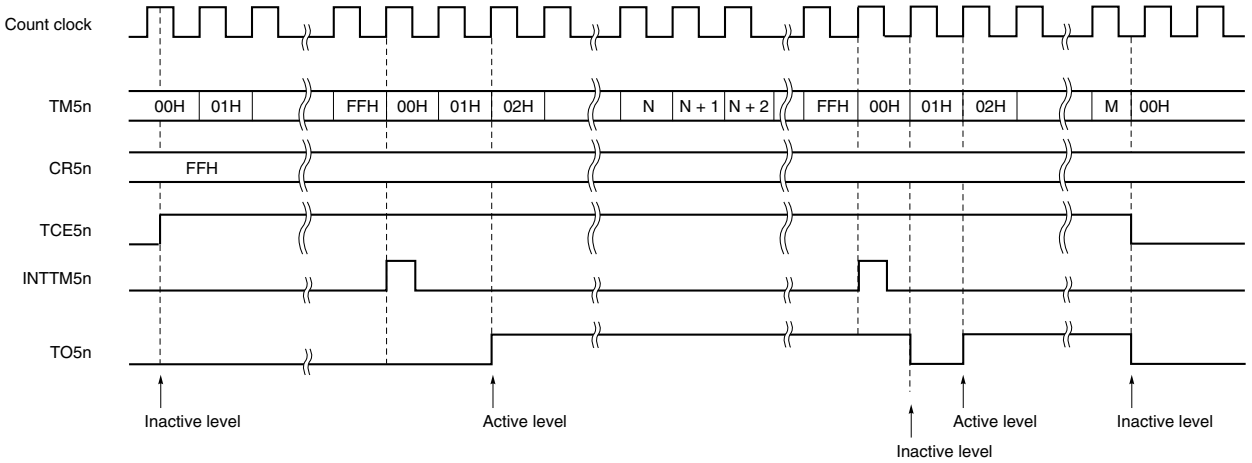
(a) Basic operation (when active level = H)



(b) When CR5n = 0



(c) When CR5n = FFH

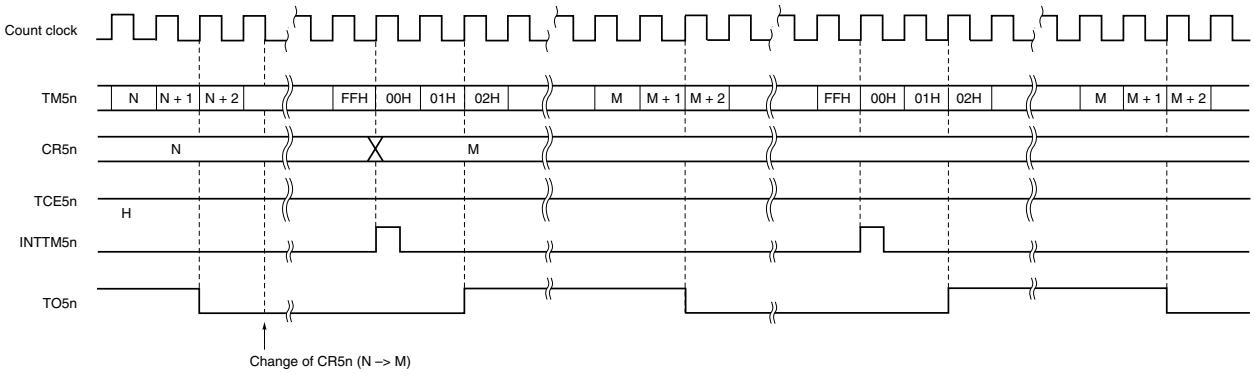


Remark n = 0 to 2

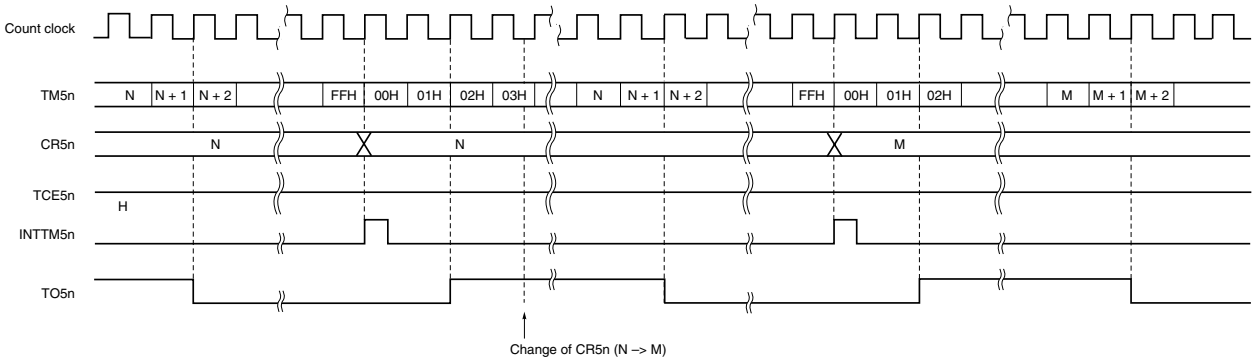
(2) Operation when CR5n is changed

Figure 7-15. Operation Timing When CR5n Is Changed

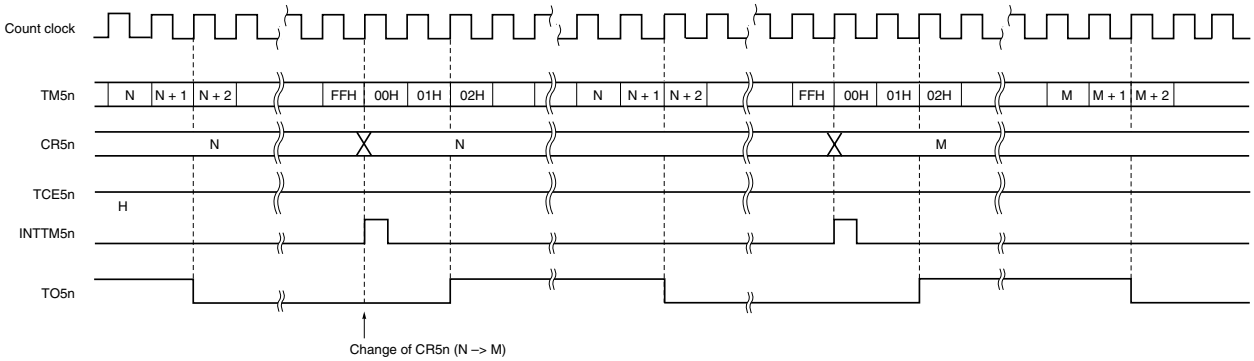
(a) If value of CR5n is changed from N to M before TM5n overflows



(b) If value of CR5n is changed from N to M after TM5n overflows



(c) If value of CR5n is changed from N to M during 2 clocks (00H, 01H) immediately after TM5n overflows



Remark n = 0 to 2

7.4.5 Interval timer (16-bit) operation

- **Cascade (16-bit timer) mode**

  The 16-bit resolution timer/counter mode is set by setting bit 4 (TMC514) of 8-bit timer mode control register 51 (TMC51) to 1.

  In this mode, TM50 and TM51 operate as a 16-bit interval timer that repeatedly generates an interrupt request at intervals specified by the count value set in advance to 8-bit timer compare registers 50 and 51 (CR50 and CR51).

<Setting>

<1> Set each register.

- TCL50: TM50 selects a count clock.
  TM51, which is connected in cascade, does not have to be set.
- CR50 and CR51: Compare values (each compare value can be set in a range of 00H to FFH).
- TMC50 and TMC51: Select the clear & start mode on match between TM50 and CR50 (TM51 and CR51).

  [TM50 → TMC50 = 0000×××0B ×: don't care
  TM51 → TMC51 = 0001×××0B ×: don't care]

<2> By setting TCE51 of TMC51 to 1 first, and then setting TCE50 of TMC50 to 1, the count operation is started.

<3> When the value of TM50 connected in cascade matches the value of CR50, TM50 generates INTTM50 (TM50 and TM51 are cleared to 00H).

<4> After that, INTTM50 is repeatedly generated at the same interval.

Cautions 1. Timer operation must be stopped before setting the compare registers (CR50 and CR51).

2. Even if the timers are connected in cascade, TM51 generates INTTM51 when the count value of TM51 matches the value of CR51. Be sure to mask TM51 to disable the generation of an interrupt.

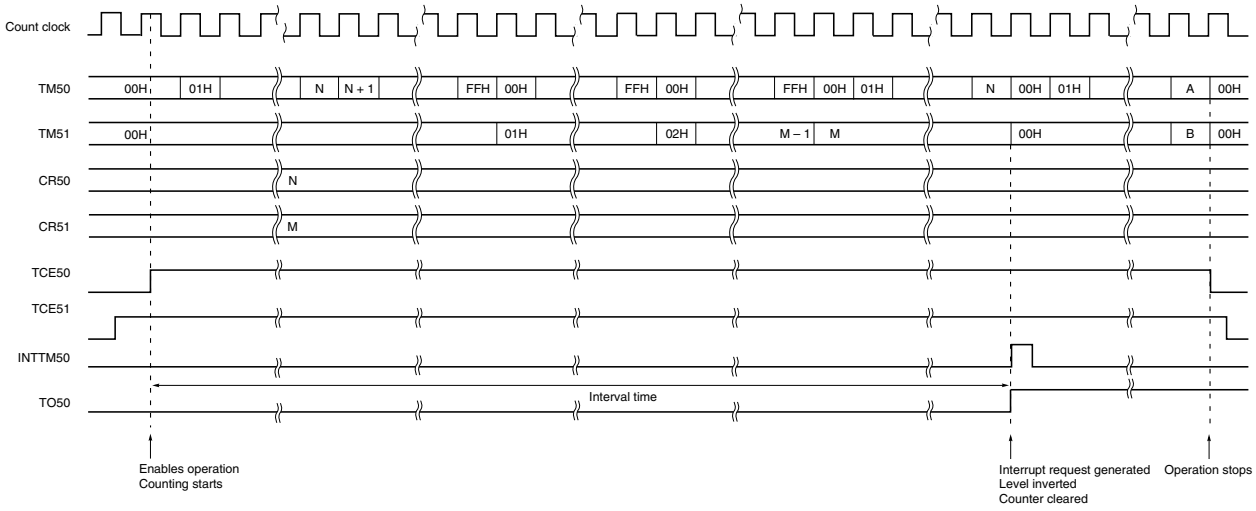
3. Set TCE50 and TCE51 in the order of TM51, then TM50.

4. Counting can be started or stopped by setting or clearing only the TCE50 bit of TM50 to 1 or 0.

5. TM52 cannot be connected in cascade.

Figure 7-16 shows an example of timing in the 16-bit resolution cascade mode.

Figure 7-16. 16-Bit Resolution Cascade Mode

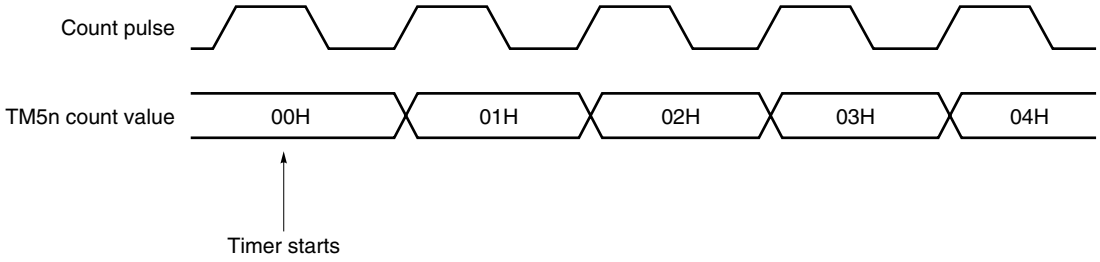


7.5 Cautions for 8-Bit Timer/Event Counter

(1) Timer start errors

An error of up to 1 clock occurs after the timer has been started until a match signal is generated. This is because 8-bit timer/counter 5n (TM5n: n = 0 to 2) is started asynchronously with the count pulse.

Figure 7-17. Start Timing of 8-Bit Timer/Counter

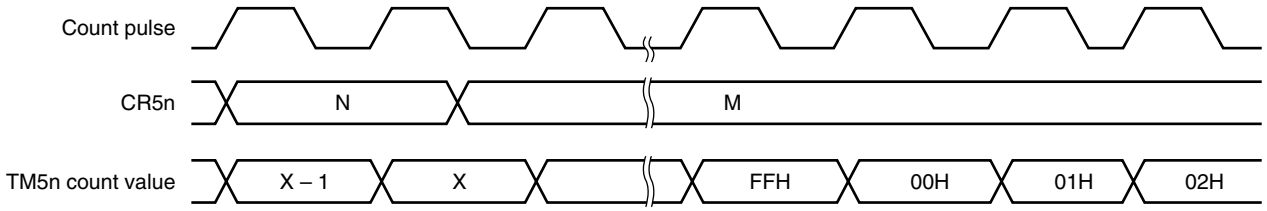


Remark n = 0 to 2

(2) Operation after changing value of compare register during timer count operation

If the value after 8-bit compare register 5n (CR5n: n = 0 to 2) is changed is smaller than that of corresponding 8-bit timer counter 5n (TM5n: n = 0 to 2), TM5n continues counting, overflows, and then restarts counting from 0. Therefore, if the value (M) after CR5n is changed is smaller than that (N) before the change, it is necessary to restart the timer after changing the value of CR5n.

Figure 7-18. Timing after Changing Values of Compare Registers during Timer Count Operation



Caution Except when TI5n input is selected, be sure to clear TCE5n to 0 before setting the STOP mode.

Remark N > X > M
n = 0 to 2

(3) Reading TM5n during timer operation

Because the count clock is stopped when TM5n is read during operation, select a count clock with a waveform whose high-/low-level is longer than two CPU clock cycles. For example, in the case of a CPU clock (f_{CPU}) equal to f_X, TM5n can be read as long as the selected count clock is $f_X/4$ or lower.

Remark n = 0 to 2
f_X: System clock oscillation frequency

CHAPTER 8 WATCH TIMER

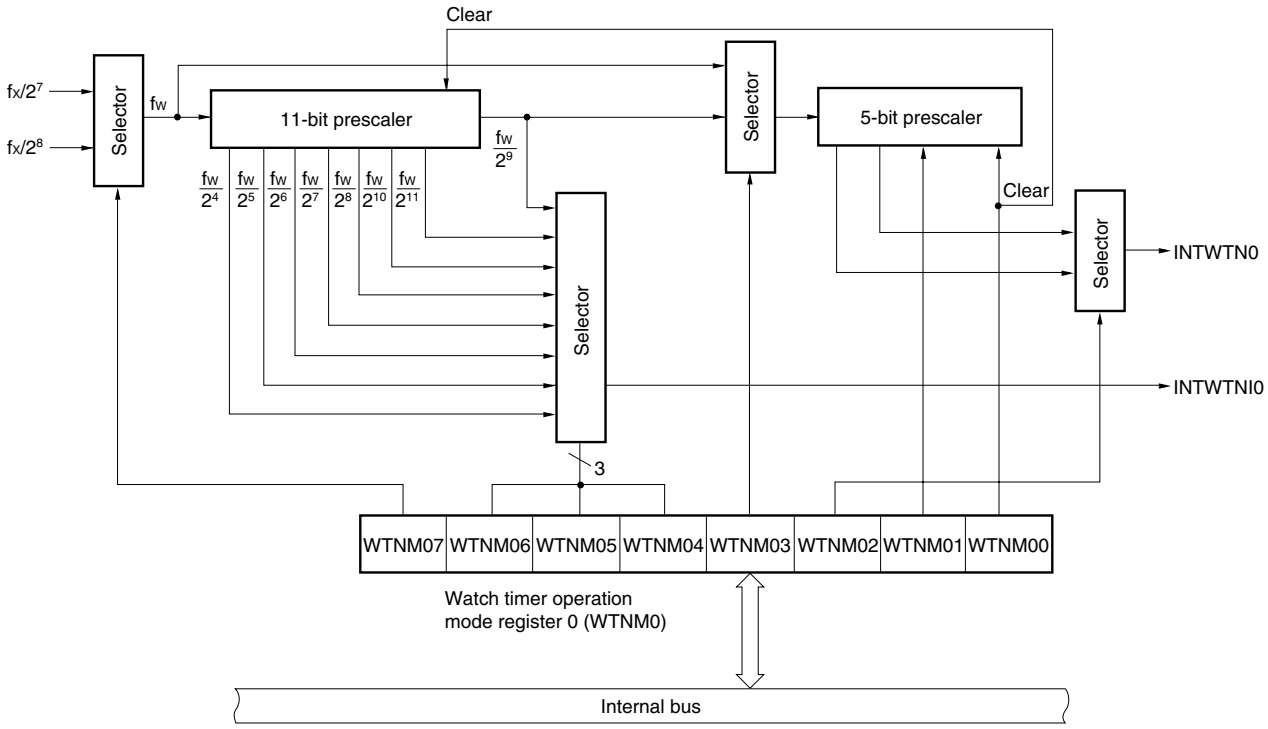
8.1 Watch Timer Functions

The watch timer has the following functions.

- Watch timer
- Interval timer

The watch timer and the interval timer can be used simultaneously.
Figure 8-1 shows the watch timer block diagram.

Figure 8-1. Block Diagram of Watch Timer



Remark fx: System clock oscillation frequency
fw: Watch timer clock frequency

(1) Watch timer

An interrupt request (INTWTN0) is generated at preset time intervals. Time intervals such as 0.5 sec. or 1.0 sec. cannot be created in a 6.29-MHz system clock, therefore set time intervals such as 0.5 sec. or 1.0 sec. using a program.

(2) Interval timer

An interrupt request (INTWTNI0) is generated at a preset time interval.

Table 8-1. Interval Time of Interval Timer

Interval Time	When Operated at f_X = 6.291456 MHz	
	$f_W = f_X/2^8$	$f_W = f_X/2^7$
$2^4 \times 1/f_W$	651 μs	326 μs
$2^5 \times 1/f_W$	1.30 ms	651 μs
$2^6 \times 1/f_W$	2.60 ms	1.30 ms
$2^7 \times 1/f_W$	5.21 ms	2.60 ms
$2^8 \times 1/f_W$	10.4 ms	5.21 ms
$2^9 \times 1/f_W$	20.8 ms	10.4 ms
$2^{10} \times 1/f_W$	41.7 ms	20.8 ms
$2^{11} \times 1/f_W$	83.3 ms	41.7 ms

Remark f_X: System clock oscillation frequency
f_W: Watch timer clock frequency ($f_X/2^7$ or $f_X/2^8$)

8.2 Watch Timer Configuration

The watch timer consists of the following hardware.

Table 8-2. Configuration of Watch Timer

Item	Configuration
Prescaler	11 bits $\times$ 1, 5 bits $\times$ 1
Control register	Watch timer operation mode register 0 (WTNM0)

8.3 Watch Timer Control Registers

Watch timer operation mode register 0 (WTNM0) is a register to control watch timer.

- **Watch timer operation mode register 0 (WTNM0)**
  This register enables/disables the watch timer operation, sets 11-bit prescaler interval time, and controls 5-bit prescaler operation.
  WTNM0 is set with a 1-bit or 8-bit memory manipulation instruction.
  $\overline{\text{RESET}}$ input sets WTNM0 to 00H.

Figure 8-2. Format of Watch Timer Operation Mode Register 0 (WTNM0)

Address: FF41H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
WTNM0	WTNM07	WTNM06	WTNM05	WTNM04	WTNM03	WTNM02	WTNM01	WTNM00

WTNM07	Watch timer count clock (fw) selection
0	$f_X/2^8$ (24.576 kHz)
1	$f_X/2^7$ (49.152 kHz)

WTNM06	WTNM05	WTNM04	11-bit prescaler interval time selection
0	0	0	$2^4/f_W$
0	0	1	$2^5/f_W$
0	1	0	$2^6/f_W$
0	1	1	$2^7/f_W$
1	0	0	$2^8/f_W$
1	0	1	$2^9/f_W$
1	1	0	$2^{10}/f_W$
1	1	1	$2^{11}/f_W$

WTNM03	WTNM02	Selection of interrupt request timing of the clock timer		
			WTNM07 = 0	WTNM07 = 1
0	0	$2^{14}/f_W$	$2^{22}/f_X$ (0.67 s)	$2^{21}/f_X$ (0.33 s)
0	1	$2^{13}/f_W$	$2^{21}/f_X$ (0.33 s)	$2^{20}/f_X$ (0.167 s)
1	0	$2^5/f_W$	$2^{13}/f_X$ (0.13 ms)	$2^{12}/f_X$ (651 μs)
1	1	$2^4/f_W$	$2^{12}/f_X$ (651 μs)	$2^{11}/f_X$ (326 μs)

WTNM01	5-bit prescaler operation control
0	Clear after operation stop
1	Start

WTNM00	Watch timer operation
0	Operation stop (clear both 11-bit prescaler and 5-bit prescaler)
1	Operation enable

Remarks 1. fw: Watch timer clock frequency ($f_X/2^7$ or $f_X/2^8$)

2. fx: System clock oscillation frequency

3. Figures in parentheses apply to operation with fx = 6.291456 MHz.

8.4 Watch Timer Operations

8.4.1 Watch timer operation

By counting 3 times, the watch timer operates as a clock timer with preset timing intervals such as 0.5 sec. or 1.0 sec.

Bits 2, 3, and 7 (WTNM02, WTNM03, WTNM07) of watch timer operation mode register 0 (WTNM0) enable the selection of timing for the watch timer.

The watch timer generates an interrupt request (INTWT) at a constant time interval.

If bit 0 (WTNM00) and bit 1 (WTNM01) of watch timer operation mode register 0 (WTNM0) are set to 1, the count operation starts. If they are set to 0, the 5-bit prescaler is cleared and the count operation stops.

For simultaneous operation of the interval timer, a zero-second start can be achieved by setting WTNM01 to 0. However, in this case, since the 11-bit prescaler is not cleared, at the first overflow (INTWTN0) after the clock timer's zero sec. start, an error of up to $2^{11} \times 1/f_W$ sec. is generated.

8.4.2 Interval timer operation

The watch timer operates as interval timer that generates an interrupt request (INTWTNI0) repeatedly at an interval of the preset count value.

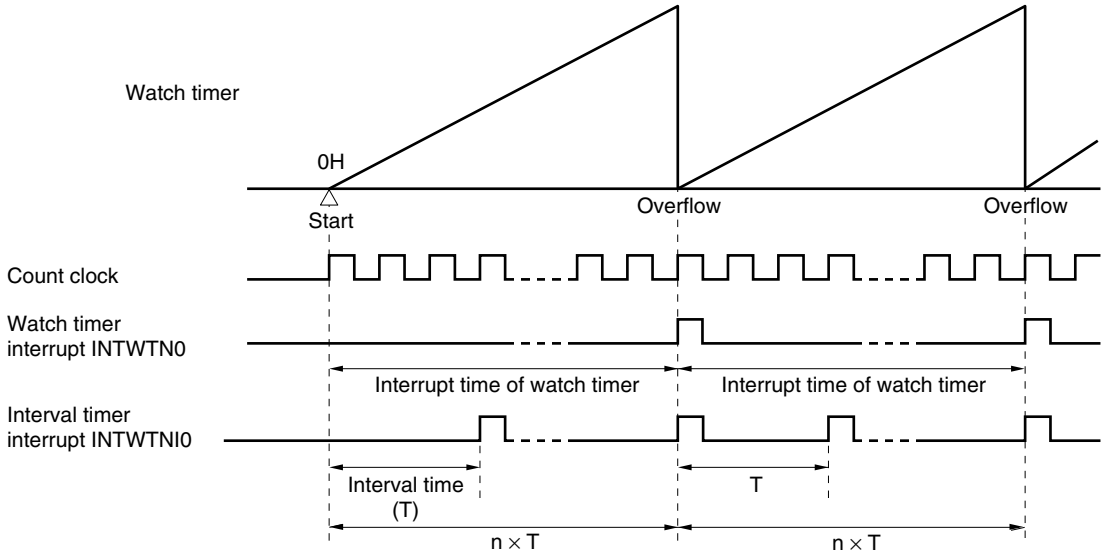
The interval time can be selected with bits 4 to 6 and 7 (WTNM04 to WTNM06 and WTNM07) of watch timer operation mode register 0 (WTNM0).

Table 8-3. Interval Time of Interval Timer

WTNM06	WTNM05	WTNM04	Interval Time	When Operated at f_X = 6.291456 MHz	
				WTNM07 = 0	WTNM07 = 1
0	0	0	$2^4 \times 1/f_W$	651 μs	326 μs
0	0	1	$2^5 \times 1/f_W$	1.30 ms	651 μs
0	1	0	$2^6 \times 1/f_W$	2.60 ms	1.30 ms
0	1	1	$2^7 \times 1/f_W$	5.21 ms	2.60 ms
1	0	0	$2^8 \times 1/f_W$	10.4 ms	5.21 ms
1	0	1	$2^9 \times 1/f_W$	20.8 ms	10.4 ms
1	1	0	$2^{10} \times 1/f_W$	41.7 ms	20.8 ms
1	1	1	$2^{11} \times 1/f_W$	83.3 ms	41.7 ms

Remark f_X: System clock oscillation frequency
f_W: Watch timer clock frequency

Figure 8-3. Operation Timing of Watch Timer/Interval Timer



Remark n: The number of times of interval timer operations

CHAPTER 9 WATCHDOG TIMER

9.1 Watchdog Timer Functions

The watchdog timer has the following functions.

- Watchdog timer
- Interval timer
- Oscillation stabilization time selection

Caution Select the watchdog timer mode or the interval timer mode with the watchdog timer mode register (WDTM). (The watchdog timer and the interval timer cannot be used simultaneously.)

(1) Watchdog timer mode

The watchdog timer is used to detect an inadvertent program loop (program runaway). When the runaway is detected, a non-maskable interrupt request or $\overline{\text{RESET}}$ can be generated.

Table 9-1. Runaway Detection Time of Watchdog Timer

Runaway Detection Time	At f_X = 6.29 MHz	Runaway Detection Time	At f_X = 6.29 MHz
$2^{12} \times 1/f_X$	651 μs	$2^{16} \times 1/f_X$	10.4 ms
$2^{13} \times 1/f_X$	1.30 ms	$2^{17} \times 1/f_X$	20.8 ms
$2^{14} \times 1/f_X$	2.60 ms	$2^{18} \times 1/f_X$	41.7 ms
$2^{15} \times 1/f_X$	5.21 ms	$2^{20} \times 1/f_X$	167 ms

f_X: System clock oscillation frequency

(2) Interval timer mode

An interrupt request is generated at preset time intervals.

Table 9-2. Interval Time

Interval Time	At f_X = 6.29 MHz	Interval Time	At f_X = 6.29 MHz
$2^{12} \times 1/f_X$	651 μs	$2^{16} \times 1/f_X$	10.4 ms
$2^{13} \times 1/f_X$	1.30 ms	$2^{17} \times 1/f_X$	20.8 ms
$2^{14} \times 1/f_X$	2.60 ms	$2^{18} \times 1/f_X$	41.7 ms
$2^{15} \times 1/f_X$	5.21 ms	$2^{20} \times 1/f_X$	167 ms

f_X: System clock oscillation frequency

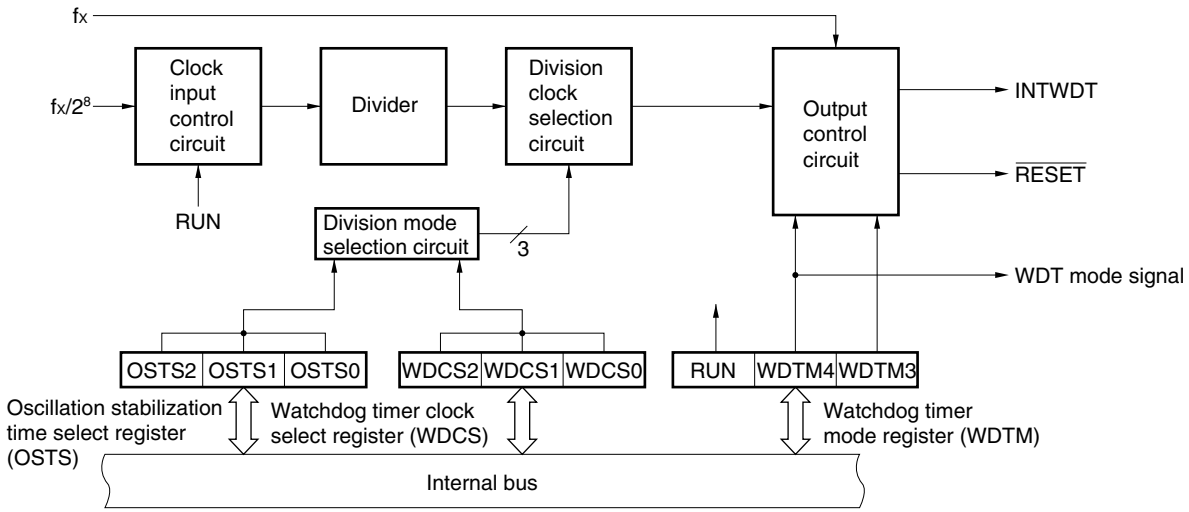
9.2 Watchdog Timer Configuration

The watchdog timer consists of the following hardware.

Table 9-3. Configuration of Watchdog Timer

Item	Configuration
Control register	Watchdog timer clock select register (WDCS) Watchdog timer mode register (WDTM) Oscillation stabilization time select register (OSTS)

Figure 9-1. Block Diagram of Watchdog Timer



9.3 Watchdog Timer Control Registers

The following three types of registers are used to control the watchdog timer.

- Watchdog timer clock select register (WDCS)
- Watchdog timer mode register (WDTM)
- Oscillation stabilization time select register (OSTS)

(1) Watchdog timer clock select register (WDCS) (refer to **Figure 9-2)**

This register sets the overflow time of the watchdog timer and the interval timer.

WDCS is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets WDCS to 00H.

Figure 9-2. Format of Watchdog Timer Clock Select Register (WDCS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0	FF42H	00H	R/W

WDCS2	WDCS1	WDCS0	Watchdog timer/interval timer overflow time selection
0	0	0	$2^{12}/f_X$ (651 μs)
0	0	1	$2^{13}/f_X$ (1.30 ms)
0	1	0	$2^{14}/f_X$ (2.60 ms)
0	1	1	$2^{15}/f_X$ (5.21 ms)
1	0	0	$2^{16}/f_X$ (10.4 ms)
1	0	1	$2^{17}/f_X$ (20.8 ms)
1	1	0	$2^{18}/f_X$ (41.7 ms)
1	1	1	$2^{20}/f_X$ (167 ms)

Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz

(2) Watchdog timer mode register (WDTM)

This register sets the watchdog timer operating mode and enables/disables counting.

WDTM is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets WDTM to 00H.

Figure 9-3. Format of Watchdog Timer Mode Register (WDTM)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	FFF9H	00H	R/W

WDTM4	WDTM3	Watchdog timer operation mode select[Note 1]
0	×	Interval timer mode[Note 2] (maskable interrupt request is generated when overflow occurs)
1	0	Watchdog timer mode 1 (non-maskable interrupt request is generated when overflow occurs)
1	1	Watchdog timer mode 2 (reset operation starts when overflow occurs)

RUN	Watchdog timer operation select[Note 3]
0	Stops counting
1	Clears counter and starts counting

Notes 1. Once WDTM3 and WDTM4 have been set to 1, they cannot be cleared to 0 by software.

2. The watchdog timer starts operating as an interval timer as soon as the RUN bit has been set to 1.

3. Once RUN has been set to 1, it cannot be cleared to 0 by software. Therefore, once counting has started, it cannot be stopped by any means other than $\overline{\text{RESET}}$ input.

Caution When RUN is set to 1 so that the watchdog timer is cleared, the actual overflow time is up to 0.5% shorter than the time set by the Watchdog Timer Clock Select Register (WDCS).

Remark × : don't care

(3) Oscillation Stabilization Time Select Register (OSTS)

This register selects the oscillation stabilization time from the reset time or STOP mode released time to the time when oscillation is stabilized.

OSTS is set with an 8-bit memory operation instruction.

$\overline{\text{RESET}}$ input sets OSTS to 04H. Therefore, when releasing the STOP mode by $\overline{\text{RESET}}$ input, the time required until STOP is released is 2^{17}/fx.

Figure 9-4. Format of Oscillation Stabilization Time Select Register (OSTS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time when STOP mode is released
0	0	0	2^{12}/fx (651 μs)
0	0	1	2^{14}/fx (2.60 ms)
0	1	0	2^{15}/fx (5.21 ms)
0	1	1	2^{16}/fx (10.4 ms)
1	0	0	2^{17}/fx (20.8 ms)
Other than above			Setting prohibited

Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz

9.4 Watchdog Timer Operations

9.4.1 Operation as watchdog timer

The watchdog timer detects an inadvertent program loop (runaway) when bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1.

The runaway detection time interval of the watchdog timer can be selected by bits 0 to 2 (WDCS0 to WDCS2) of the watchdog timer clock select register (WDCS). By setting bit 7 (RUN) of the WDTM to 1, the watchdog timer is started. Set RUN to 1 within the set runaway detection time interval after the watchdog timer has been started. By setting RUN to 1, the watchdog timer can be cleared and counting can start. If RUN is not set to 1, and the runaway detection time is exceeded, the system is reset or a non-maskable interrupt request is generated according to the value of bit 3 (WDTM3) of WDTM.

The watchdog timer continues operation in the HALT mode, but stops in the STOP mode. Therefore, set RUN to 1 before entering the STOP mode to clear the watchdog timer, and then execute the STOP instruction.

Cautions 1. The actual runaway detection time may be up to 0.5% shorter than the set time.

2. The count operation of the watchdog timer is stopped when the subsystem clock is selected as the CPU clock.

Table 9-4. Runaway Detection Time of Watchdog Timer

WDCS22	WDCS21	WDCS20	Inadvertent Loop Detection Time	At f_X = 6.29 MHz
0	0	0	$2^{12} \times 1/f_X$	651 μs
0	0	1	$2^{13} \times 1/f_X$	1.30 ms
0	1	0	$2^{14} \times 1/f_X$	2.60 ms
0	1	1	$2^{15} \times 1/f_X$	5.21 ms
1	0	0	$2^{16} \times 1/f_X$	10.4 ms
1	0	1	$2^{17} \times 1/f_X$	20.8 ms
1	1	0	$2^{18} \times 1/f_X$	41.7 ms
1	1	1	$2^{20} \times 1/f_X$	167 ms

f_X: System clock oscillation frequency

9.4.2 Operation as interval timer

The watchdog timer operates as an interval timer that generates interrupt requests repeatedly at an interval of a preset count value by setting bit 4 (WDTM4) of the watchdog timer mode register (WDTM) to 0.

The interval time of interval timer is selected with bits 0 to 2 (WDCS0 to WDCS2) of the watchdog timer clock select register (WDCS). When bit 7 (RUN) of WDTM is set to 1, the watchdog timer starts operation as an interval timer.

When the watchdog timer operates as an interval timer, the interrupt mask flag (WDTMK) and priority specify flag (WDTPR) are validated and the maskable interrupt request (INTWDT) can be generated. Among maskable interrupts, INTWDT has the highest priority at default.

The interval timer continues operating in the HALT mode but it stops in STOP mode. Therefore, set RUN to 1 before the STOP mode is set, clear the interval timer and then execute the STOP instruction.

Cautions 1. Once bit 4 (WDTM4) of WDTM is set to 1 (this selects the watchdog timer mode), the interval timer mode is not set unless $\overline{\text{RESET}}$ input is applied.

2. The interval time just after setting by WDTM may be up to 0.5% shorter than the set time.

Table 9-5. Interval Time of Interval Timer

WDCS2	WDCS1	WDCS0	Interval Time	At f_X = 6.29 MHz
0	0	0	$2^{12} \times 1/f_X$	651 μs
0	0	1	$2^{13} \times 1/f_X$	1.30 ms
0	1	0	$2^{14} \times 1/f_X$	2.60 ms
0	1	1	$2^{15} \times 1/f_X$	5.21 ms
1	0	0	$2^{16} \times 1/f_X$	10.4 ms
1	0	1	$2^{17} \times 1/f_X$	20.8 ms
1	1	0	$2^{18} \times 1/f_X$	41.7 ms
1	1	1	$2^{20} \times 1/f_X$	167 ms

f_X: System clock oscillation frequency

CHAPTER 10 CLOCK OUTPUT/BUZZER OUTPUT CONTROL CIRCUITS

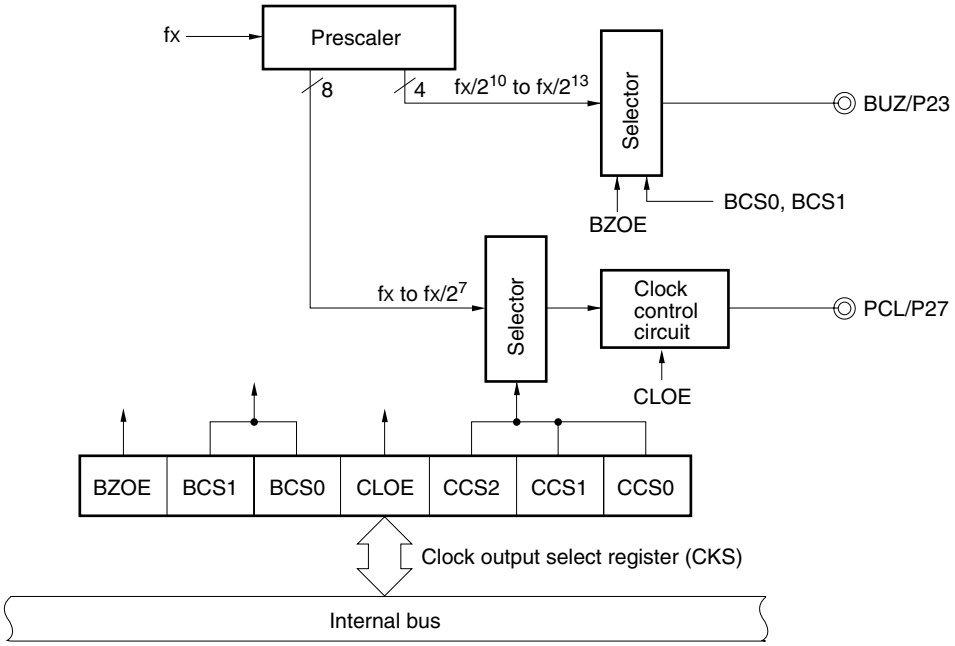
10.1 Clock Output/Buzzer Output Control Circuit Functions

The clock output control circuit is intended for carrier output during remote controlled transmission and clock output for supply to peripheral LSIs. The clock selected with the clock output selection register (CKS) is output.

The buzzer output control circuit is intended for square-wave output of buzzer frequency selected by the CKS.

Figure 10-1 shows the block diagram of clock output/buzzer output control circuit.

Figure 10-1. Block Diagram of Clock Output/Buzzer Output Control Circuit



10.2 Clock Output/Buzzer Output Control Circuit Configuration

The clock output/buzzer output control circuit consists of the following hardware.

Table 10-1. Configuration of Clock Output/Buzzer Output Control Circuit

Item	Configuration
Control register	Clock output selection register (CKS) Port mode register 2 (PM2)[Note]

Note See **Figure 4-3 Block Diagram of P20 to P27**.

10.3 Clock Output/Buzzer Output Control Circuit Control Registers

The following two types of registers are used to control the clock output/buzzer output control circuit.

- Clock output selection register (CKS)
- Port mode register 2 (PM2)

(1) Clock output selection register (CKS)

This register sets output enable/disable for clock output (PCL) and buzzer frequency output (BUZ), and sets the output clock.

CKS is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CKS to 00H.

Figure 10-2. Format of Clock Output Selection Register (CKS)

Address: FF40H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CKS	BZOE	BCS1	BCS0	CLOE	0	CCS2	CCS1	CCS0

BZOE	BUZ output enable/disable setting
0	Stops clock division circuit operation. BUZ fixed to low level
1	Enables clock division circuit operation. BUZ output enabled

BCS1	BCS0	BUZ output clock select
0	0	$f_X/2^{10}$ (6.14 kHz)
0	1	$f_X/2^{11}$ (3.07 kHz)
1	0	$f_X/2^{12}$ (1.54 kHz)
1	1	$f_X/2^{13}$ (768 Hz)

CLOE	PCL output enable/disable setting
0	Stops clock division circuit operation. PCL fixed to low level
1	Enables clock division circuit operation. PCL output enabled

CCS2	CCS1	CCS0	PCL output clock select
0	0	0	f_X (6.29 MHz)
0	0	1	$f_X/2$ (3.15 MHz)
0	1	0	$f_X/2^2$ (1.57 MHz)
0	1	1	$f_X/2^3$ (786 kHz)
1	0	0	$f_X/2^4$ (393 kHz)
1	0	1	$f_X/2^5$ (197 kHz)
1	1	0	$f_X/2^6$ (98.3 kHz)
1	1	1	$f_X/2^7$ (49.2 kHz)
Other than above			Setting prohibited

Caution Be sure to set bit 3 to 0.

Remarks 1. f_X: System clock oscillation frequency

2. Figures in parentheses apply to operation with f_X = 6.29 MHz.

(2) Port mode register 2 (PM2)

This register sets port 2 to input/output in 1-bit units.

When using the P23/BUZ pin for buzzer output and P27/PCL pin for clock output, set PM23, PM27, and the output latch of P23, P27 to 0.

PM2 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM2 to FFH.

Figure 10-3. Format of Port Mode Register 2 (PM2)

Address: FF22H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20

PM2n	P2n pin input/output mode selection (n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

10.4 Clock Output/Buzzer Output Control Circuit Operations

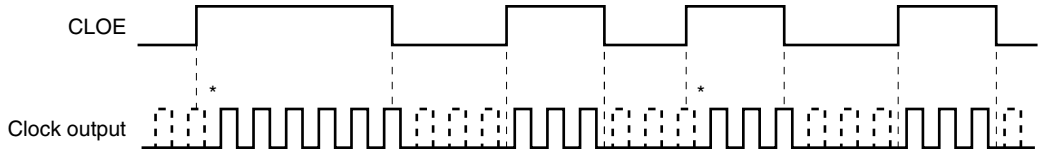
10.4.1 Operation as clock output

The clock pulse is output using the following procedure.

<1> Select the clock pulse output frequency with bits 0 to 3 (CCS0 to CCS3) of the clock output selection register (CKS) (clock pulse output in disabled status).
<2> Set bit 4 (CLOE) of CKS to 1, and enable clock output.

Remark The clock output control circuit is designed not to output pulses with a small width while enabling/disabling the clock output is switched. As shown in Figure 10-4, be sure to start output from the low period of the clock (marked with * in the figure). When stopping output, do so after securing the high level of the clock.

Figure 10-4. Output Application Example of Remote Control



10.4.2 Operation as buzzer output

The buzzer frequency is output using the following procedure.

<1> Select the buzzer output frequency with bits 5 and 6 (BCS0 and BCS1) of the clock output selection register (CKS) (buzzer output in disabled status).
<2> Set bit 7 (BZOE) of CKS to 1, and enable buzzer output.

CHAPTER 11 A/D CONVERTER

11.1 Function of A/D Converter

The A/D converter converts an analog input signal into a digital value, and consists of up to sixteen channels (ANI0 to ANI15) with a resolution of 8 bits.

The A/D converter has the following two functions.

(1) 8-bit resolution A/D conversion

8-bit resolution A/D conversion is carried out repeatedly for one channel selected among analog inputs ANI0 to ANI15. Each time an A/D conversion operation ends, an interrupt request (INTAD) is generated.

(2) Power-fail detection function

This function is to detect a voltage drop of a battery set in an automobile. The values of the A/D conversion result (ADCR3 register value) and power-fail comparison threshold register 3 (PFT3) are compared. The INTAD is generated only when a comparative condition has been matched.

Figure 11-1. Block Diagram of A/D Converter

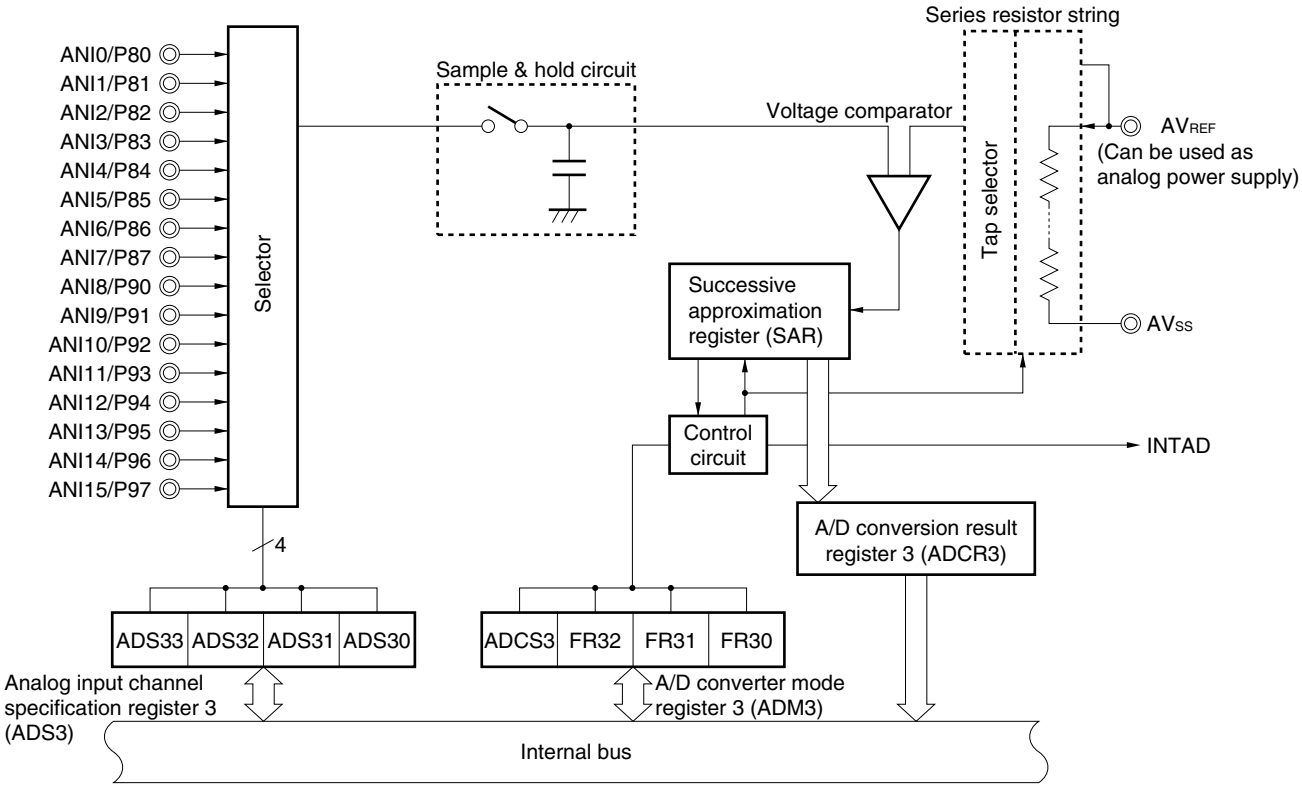
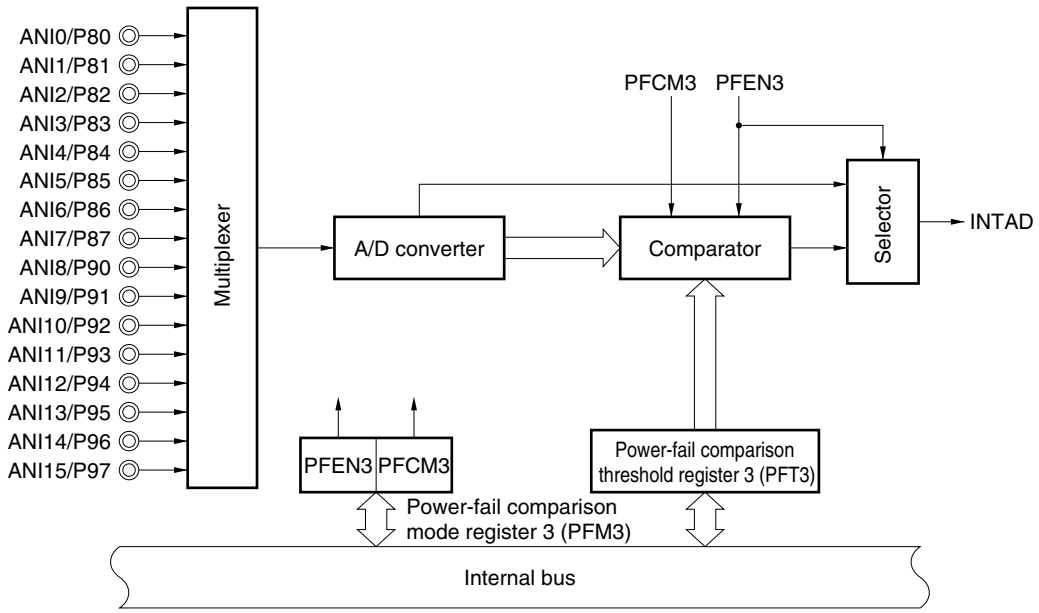


Figure 11-2. Block Diagram of Power-Fail Detection Function



11.2 A/D Converter Configuration

The A/D converter consists of the following hardware.

Table 11-1. Configuration of A/D Converter

Item	Configuration
Analog input	16 channels (ANI0 to ANI15)
Registers	Successive approximation register (SAR) A/D conversion result register 3 (ADCR3)
Control register	A/D converter mode register 3 (ADM3) Analog input channel specification register 3 (ADS3) Power-fail comparison mode register 3 (PFM3) Power-fail comparison threshold register 3 (PFT3)

(1) Successive approximation register (SAR)

This register compares the analog input voltage value with the voltage tap (compare value) value applied from the series resistor string, and holds the result starting from the most significant bit (MSB).

When the result up to the least significant bit (LSB) is held (end of A/D conversion), the SAR contents are transferred to the A/D conversion result register.

(2) A/D conversion result register 3 (ADCR3)

The ADCR3 stores the A/D conversion result. Each time A/D conversion ends, the conversion result is loaded from the successive approximation register.

ADCR3 is read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ADCR3 to undefined.

Caution When writing to A/D converter mode register 3 (ADM3) and analog input channel specification register 3 (ADS3), the contents of ADCR3 may become undefined. Read the conversion result following conversion completion before writing to ADM3, ADS3. Using timing other than the above may cause an incorrect conversion result to be read.

(3) Sample & hold circuit

The sample & hold circuit samples each analog input signal sequentially applied from the input circuit, and sends it to the voltage comparator. This circuit holds the sampled analog input voltage value during A/D conversion.

(4) Voltage comparator

The voltage comparator compares the analog input with the series resistor string output voltage.

(5) Series resistor string

The series resistor string is connected between AV_{REF} and AV_{SS}, and generates a voltage to be compared with the analog input.

(6) Pins ANI0 to ANI15

These sixteen analog input pins input analog signals to undergo A/D conversion to the A/D converter. ANI0 to ANI15 are alternate-function pins that can also be used for digital input.

Cautions 1. Observe the rated range of the ANI0 to ANI15 input voltage. If a voltage of AV_{REF} or higher or a voltage of AV_{SS} or lower (even if within the range of absolute maximum ratings) is input to an analog input channel, the converted value of that channel becomes undefined. In addition, the converted values of the other channels may also be affected.

2. The analog input pins (ANI0 to ANI15) are also used as input port pins (P80 to P87 and P90 to P97). When A/D conversion is performed with any of ANI0 to ANI15 selected, do not execute the input instruction to ports 8 and 9 while conversion is in progress; otherwise the conversion resolution may be degraded. If a digital pulse is applied to the pins adjacent to the pins currently used for A/D conversion, the expected value of the A/D conversion may not be obtained due to coupling noise. Therefore, do not apply a pulse to the pins adjacent to the pin undergoing A/D conversion.

(7) AV_{REF} pin

The AV_{REF} pin inputs the A/D converter reference voltage.

It converts signals input to ANI0 to ANI15 into digital signals based on a voltage between AV_{REF} and AV_{SS}.

In a standby mode, the current flowing into series resistor strings can be reduced by changing the input voltage of the AV_{REF} pin to AV_{SS} level.

It can also be used as the analog power supply. When the A/D converter is used, be sure to use the AV_{REF} pin for the power supply.

Caution A series resistor string of several tens of kΩ is connected between the AV_{REF} and AV_{SS} pins. Therefore, if the output impedance of the reference voltage source is high, this will result in parallel connection to the series resistor string between the AV_{REF} and AV_{SS} pins, resulting in a large reference voltage error.

(8) AV_{SS} pin

The AV_{SS} pin is the GND potential pin for the A/D converter. Always use the AV_{SS} pin at the same potential as the V_{SS0} pin, even when the A/D converter is not used.

11.3 A/D Converter Control Registers

The following four types of registers are used to control the A/D converter.

- A/D converter mode register 3 (ADM3)
- Analog input channel specification register 3 (ADS3)
- Power-fail comparison mode register 3 (PFM3)
- Power-fail comparison threshold register 3 (PFT3)

(1) A/D converter mode register 3 (ADM3)

This register sets the conversion time for analog input to be A/D converted, and starts/stops conversion.

ADM3 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ADM3 to 00H.

Figure 11-3. Format of A/D Converter Mode Register 3 (ADM3)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADM3	ADCS3	0	FR32	FR31	FR30	0	0	0	FF80H	00H	R/W

ADCS3	A/D conversion operation control
0	Stops conversion operation
1	Enables conversion operation

FR32	FR31	FR30	Conversion time select[Note]
0	0	0	288/fx (45.8 μs)
0	0	1	240/fx (38.1 μs)
0	1	0	192/fx (30.5 μs)
1	0	0	144/fx (22.9 μs)
1	0	1	120/fx (19.1 μs)
1	1	0	96/fx (15.3 μs)
Other than above			Setting prohibited

Note Set so that the A/D conversion time is 14 μs or longer.

Cautions 1. Be sure to set bits 0 to 2 and 6 of ADM3 to 0.

2. A/D conversion operation must be stopped before rewriting bits FR30 to FR32 to values other than the identical data.

Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

(2) Analog input channel specification register 3 (ADS3)

This register specifies the analog voltage input port to be A/D converted.

ADS3 is set with an 8-bit memory manipulation.

$\overline{\text{RESET}}$ input sets ADS3 to 00H.

Figure 11-4. Format of Analog Input Channel Specification Register 3 (ADS3)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADS3	0	0	0	0	ADS33	ADS32	ADS31	ADS30	FF81H	00H	R/W

ADS33	ADS32	ADS31	ADS30	Analog input channel specification
0	0	0	0	ANI0
0	0	0	1	ANI1
0	0	1	0	ANI2
0	0	1	1	ANI3
0	1	0	0	ANI4
0	1	0	1	ANI5
0	1	1	0	ANI6
0	1	1	1	ANI7
1	0	0	0	ANI8
1	0	0	1	ANI9
1	0	1	0	ANI10
1	0	1	1	ANI11
1	1	0	0	ANI12
1	1	0	1	ANI13
1	1	1	0	ANI14
1	1	1	1	ANI15

Caution Be sure to set bits 4 to 7 of ADS3 to 0.

(3) Power-fail comparison mode register 3 (PFM3)

Power-fail comparison mode register 3 (PFM3) is a register that controls the comparison operation.

PFM3 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PFM3 to 00H.

Figure 11-5. Format of Power-Fail Comparison Mode Register 3 (PFM3)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PFM3	PFEN3	PFCM3	PFHRM3	0	0	0	0	0	FF84H	00H	R/W

PFEN3	Power-fail comparison enable
0	Prohibit power-fail comparison (used as a normal A/D converter)
1	Enable power-fail comparison (used for power-fail detection)

PFCM3		Power-fail comparison mode select
0	ADCR3 ≥ PFT3	Interrupt request signal (INTAD) generation
	ADCR3 < PFT3	No INTAD generation
1	ADCR3 ≥ PFT3	No INTAD generation
	ADCR3 < PFT3	INTAD generation

PFHRM3	Power-fail HALT repeat mode select
0	Prohibit power-fail HALT repeat mode operation
1	Enable power-fail HALT repeat mode operation

Caution Be sure to set bits 0 to 4 of PFM3 to 0.

(4) Power-fail comparison threshold register 3 (PFT3)

Power-fail comparison threshold register 3 (PFT3) is a register that sets the threshold when comparing the values with the A/D conversion result.

PFT3 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PFT3 to 00H.

Figure 11-6. Format of Power-Fail Comparison Threshold Register 3 (PFT3)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PFT3	PFT37	PFT36	PFT35	PFT34	PFT33	PFT32	PFT31	PFT30	FF83H	00H	R/W

11.4 A/D Converter Operations

11.4.1 Basic operations of A/D converter

<1> Select one channel for A/D conversion with analog input channel specification register 3 (ADS3).

<2> The voltage input to the selected analog input channel is sampled by the sample & hold circuit.

<3> When sampling has been done for a certain time, the sample & hold circuit is placed in the hold state and the input analog voltage is held until the A/D conversion operation is ended.

<4> Bit 7 of the successive approximation register (SAR) is set. The series resistor string voltage tap is set to (1/2) AV_{REF} by the tap selector.

<5> The voltage difference between the series resistor string voltage tap and analog input is compared by the voltage comparator. If the analog input is greater than (1/2) AV_{REF}, the MSB of SAR remains set to 1. If the analog input is smaller than (1/2) AV_{REF0}, the MSB is reset to 0.

<6> Next, bit 6 of SAR is automatically set to 1, and the operation proceeds to the next comparison. The series resistor string voltage tap is selected according to the preset value of bit 7, as described below.

- Bit 7 = 1: (3/4) AV_{DD}
- Bit 7 = 0: (1/4) AV_{DD}

The voltage tap and analog input voltage are compared and bit 6 of SAR is manipulated as follows.

- Analog input voltage ≥ Voltage tap: Bit 6 = 1
- Analog input voltage < Voltage tap: Bit 6 = 0

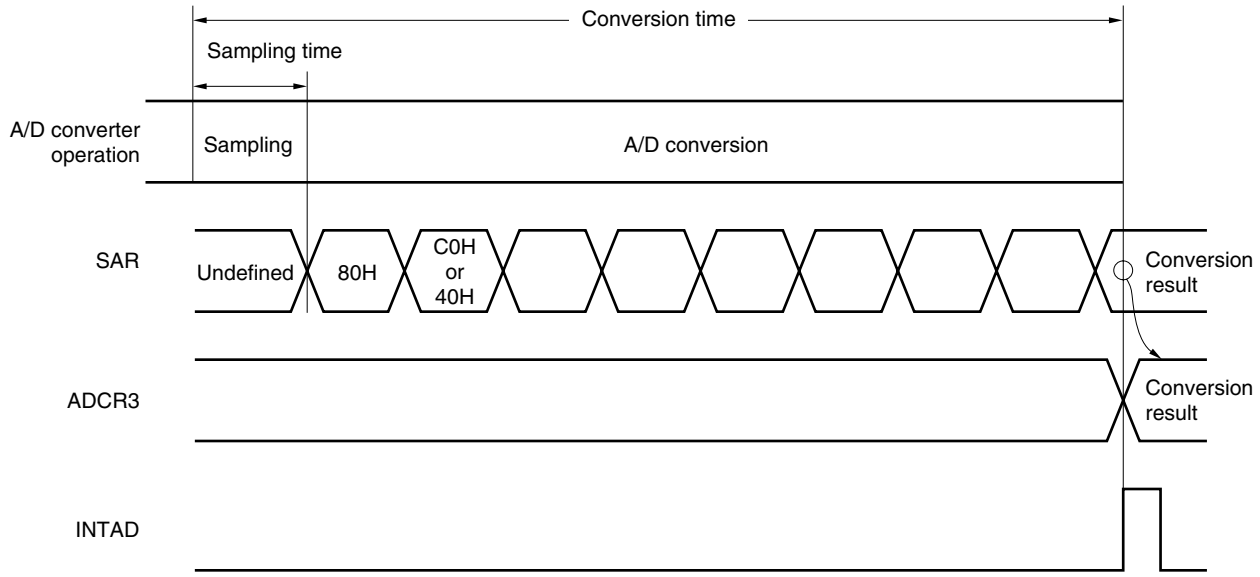
<7> Comparison is continued in this way up to bit 0 of SAR.

<8> Upon completion of the comparison of 8 bits, an effective digital result value remains in SAR, and the result value is transferred to the A/D conversion result register 3 (ADCR3) and then latched.

At the same time, the A/D conversion end interrupt request (INTAD) can also be generated.

Caution The first A/D conversion value immediately after A/D conversion operations start may not fall within the rating.

Figure 11-7. Basic Operation of A/D Converter



A/D conversion operations are performed continuously until bit 7 (ADCS3) of the A/D converter mode register 3 (ADM3) is reset (0) by software.

If a write operation is performed to ADM3 or analog input channel specification register 3 (ADS3) during an A/D conversion operation, the conversion operation is initialized, and if the ADCS3 bit is set (1), conversion starts again from the beginning.

$\overline{\text{RESET}}$ input sets A/D conversion result register 3 (ADCR3) to undefined.

11.4.2 Input voltage and conversion results

The relationship between the analog input voltage input to the analog input pins (ANI0 to ANI15) and the A/D conversion result (stored in A/D conversion result register 3 (ADCR3)) is shown by the following expression.

$$ADCR3 = INT\left(\frac{V_{IN}}{AV_{REF}} \times 256 + 0.5\right)$$

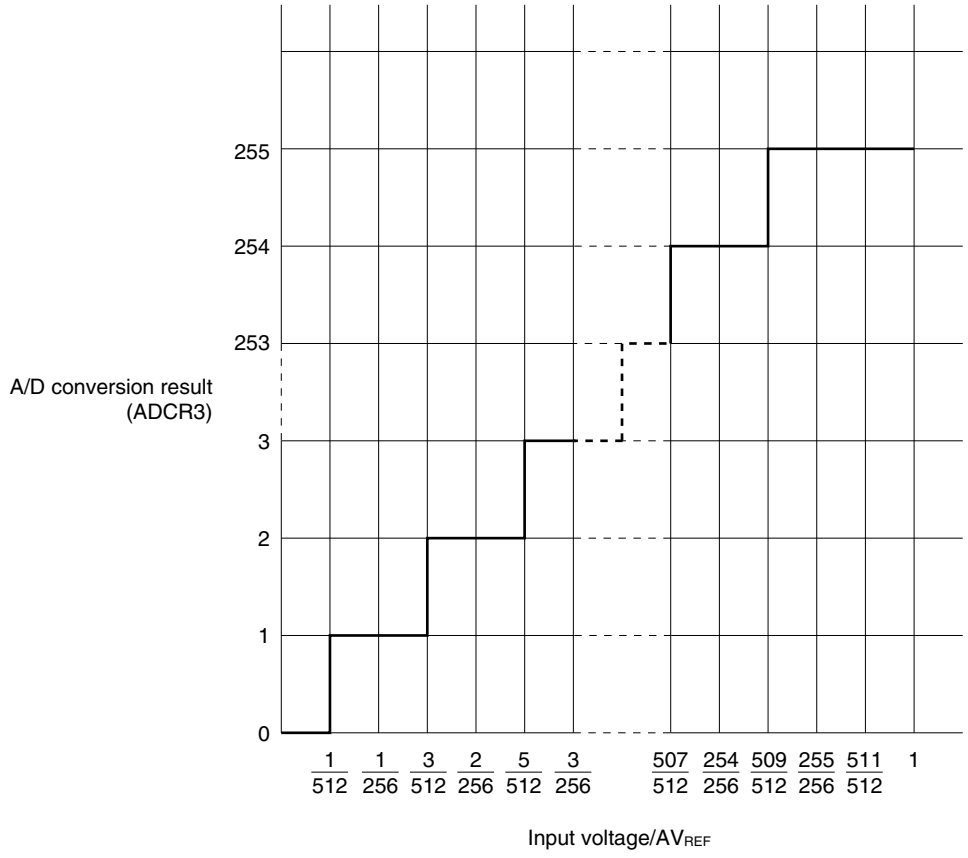
or

$$(ADCR3 - 0.5) \times \frac{AV_{REF}}{256} - V_{IN} < (ADCR3 + 0.5) \times \frac{AV_{REF}}{256}$$

where, INT( ): Function which returns integer part of value in parentheses
V_{IN}: Analog input voltage
AV_{REF}: AV_{REF} pin voltage
ADCR3: A/D conversion result register 3 (ADCR3) value

Figure 11-8 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 11-8. Relationship between Analog Input Voltage and A/D Conversion Result



11.4.3 A/D converter operation mode

The operation mode of the A/D converter is the select mode. One channel of analog input is selected from ANI0 to ANI15 by the analog input channel specification register 3 (ADS3) and A/D conversion is executed.

In addition, the following two functions can be selected by setting of bit 7 (PFEN3) of power-fail comparison mode register 3 (PFM3).

- Normal 8-bit A/D converter (PFEN3 = 0)
- Power-fail detection function (PFEN3 = 1)

(1) A/D conversion operation (when PFEN3 = 0)

By setting bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) to 1 and bit 7 (PFEN3) of power-fail comparison mode register 3 (PFM3) to 0, the A/D conversion operation of the voltage, which is applied to the analog input pin specified by the analog input channel specification register 3 (ADS3), is started.

When A/D conversion has been completed, the result of the A/D conversion is stored in the A/D conversion result register 3 (ADCR3), and an interrupt request signal (INTAD) is generated. Once the A/D conversion has started and when one A/D conversion has been completed, the next A/D conversion operation is immediately started. ★

If ADS3 is rewritten during A/D conversion, the A/D conversion under execution is suspended, and the A/D conversion of the newly selected analog input channel is started.

If 0 is written to ADCS3 of ADM3 during A/D conversion, the conversion operation is immediately stopped.

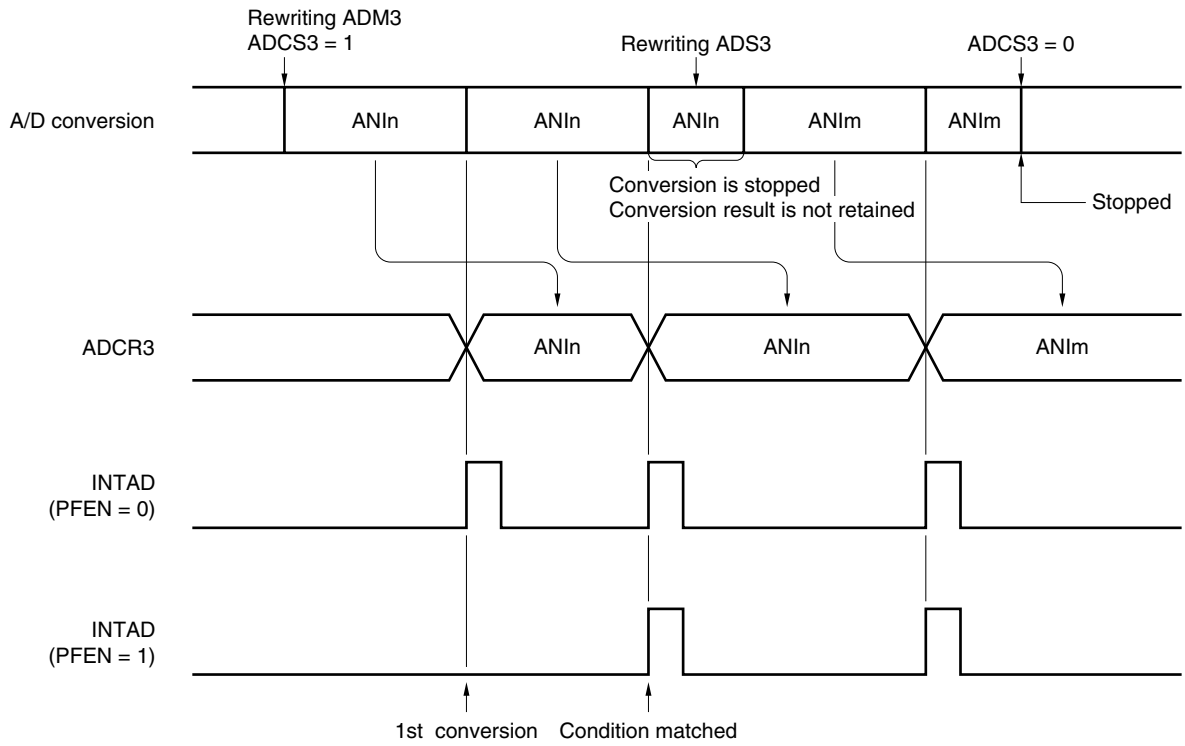
(2) Power-fail detection function (when PFEN3 = 1)

By setting bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) to 1 and bit 7 (PFEN3) of power-fail comparison mode register 3 (PFM3) to 1, the A/D conversion operation of the voltage, which applied to the analog input pin specified by the analog input channel specification register 3 (ADS3), is started.

When the A/D conversion has been completed, the result of the A/D conversion is stored in the A/D conversion result register 3 (ADCR3), the values are compared with power-fail comparison threshold register 3 (PFT3), and an interrupt request signal (INTAD) is generated under the condition specified by bit 6 (PFCM3) of PFM3.

Caution When performing power-fail comparison, the interrupt request signal (INTAD) of the first conversion is not generated. INTAD from the second conversion is valid.

Figure 11-9. A/D Conversion Operation



Remarks 1. n = 0, 1, ..., 15

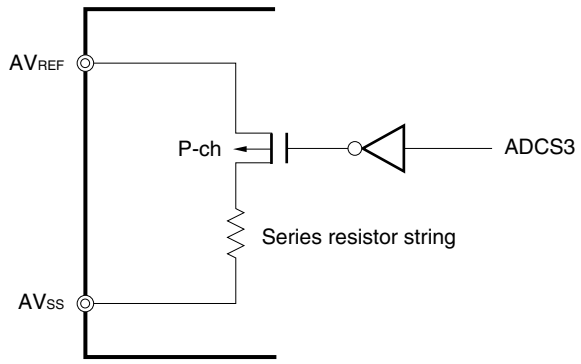
2. m = 0, 1, ..., 15

11.5 Cautions for A/D Converter

(1) Current consumption in standby mode

The A/D converter stops operating in the standby mode. At this time, current consumption can be reduced by stopping the conversion operation (by setting bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) to 0). Figure 11-10 shows how to reduce the current consumption in the standby mode.

Figure 11-10. Example of Method of Reducing Current Consumption in Standby Mode



(2) Input range of ANI0 to ANI15

Observe the rated range of the ANI0 to ANI15 input voltage. If a voltage of AV_{REF} or higher or a voltage of AV_{SS} or lower (even in the range of absolute maximum ratings) is input to an analog input channel, the converted value of that channel becomes undefined. In addition, the converted values of the other channels may also be affected.

(3) Contending operations

<1> Contention between A/D conversion result register 3 (ADCR3) write and ADCR3 read by instruction upon the end of conversion

ADCR3 read has priority. After the read operation, the new conversion result is written to ADCR3.

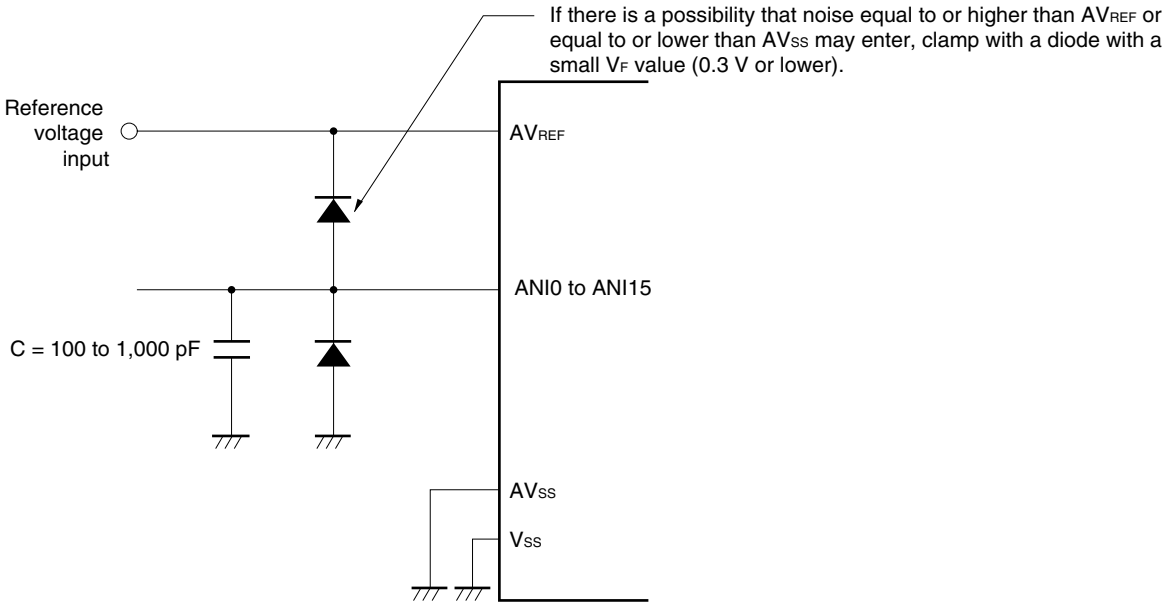
<2> Contention between ADCR3 write and A/D converter mode register 3 (ADM3) write or analog input channel specification register 3 (ADS3) write

ADM3 or ADS3 write has priority. ADCR3 write is not performed, nor is the conversion end interrupt request signal (INTAD) generated.

(4) Noise countermeasures

To maintain the 8-bit resolution, attention must be paid to noise input to the AV_{REF} pin and pins ANI0 to ANI15. Because the effect increases in proportion to the output impedance of the analog input source, it is recommended that a capacitor be connected externally, as shown in Figure 11-11, to reduce noise.

Figure 11-11. Analog Input Pin Connection



(5) ANI0/P80 to ANI7/P87, ANI8/P90 to ANI15/P97

The analog input pins (ANI0 to ANI15) are also used as I/O port pins (P80 to P87 and P90 to P97).

When A/D conversion is performed with any of ANI0 to ANI15 selected, do not execute the input instruction to ports 8 and 9 while conversion is in progress; otherwise the conversion resolution may be degraded.

If a digital pulse is applied to the pins adjacent to the pins currently used for A/D conversion, the expected value of the A/D conversion may not be obtained due to coupling noise. Therefore, do not apply a pulse to the pins adjacent to the pin undergoing A/D conversion.

(6) AV_{REF} pin input impedance

A series resistor string of several tens of kΩ is connected between the AV_{REF} and AV_{SS} pins.

Therefore, if the output impedance of the reference voltage source is high, this will result in a parallel connection to the series resistor string between the AV_{REF} and AV_{SS} pins, resulting in a large reference voltage error.

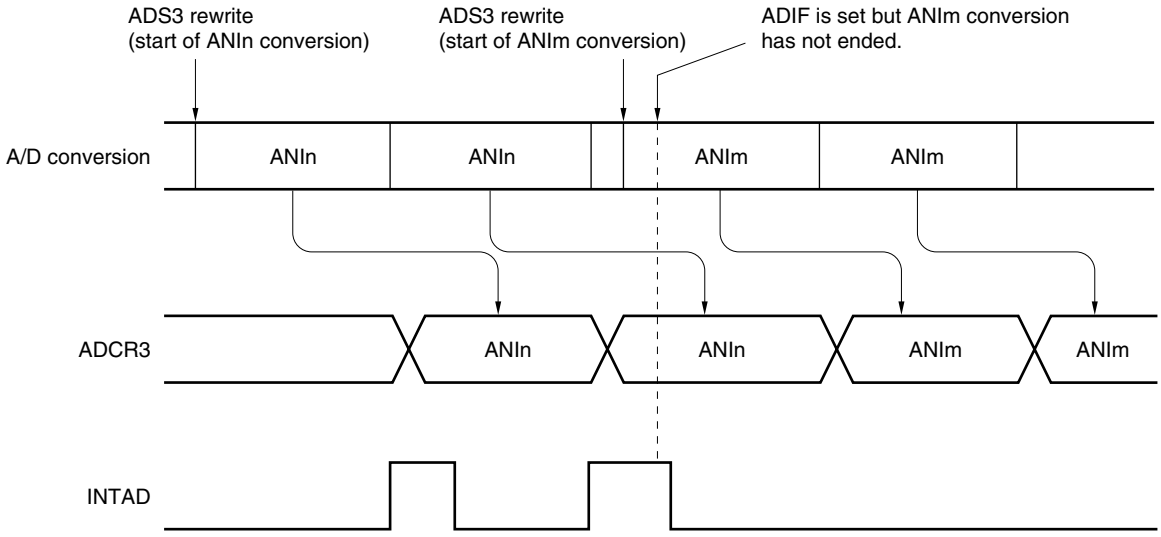
(7) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if analog input channel specification register 3 (ADS3) is changed.

Therefore, if an analog input pin is changed during A/D conversion, the A/D conversion result and conversion end interrupt request flag (ADIF) for the pre-change analog input may be set just before the ADS3 rewrite. Caution is therefore required since, at this time, when ADIF is read immediately after the ADS3 rewrite, ADIF is set despite the fact A/D conversion for the post-change analog input has not ended.

When A/D conversion is stopped and then resumed, clear ADIF before the A/D conversion operation is resumed.

Figure 11-12. Timing of A/D Conversion End Interrupt Request Generation



Remarks 1. n = 0, 1, ..., 15

2. m = 0, 1, ..., 15

(8) Conversion results just after A/D conversion start

The first A/D conversion value immediately after A/D conversion operations start may not fall within the rating. Poll the A/D conversion end interrupt request (INTAD) and take measures such as removing the first conversion results.

(9) A/D conversion result register 3 (ADCR3) read operation

When a write operation is performed to A/D converter mode register 3 (ADM3) and analog input channel specification register 3 (ADS3), the contents of ADCR3 may become undefined. Read the conversion result following conversion completion before writing to ADM3, ADS3. Using timing other than the above may cause an incorrect conversion result to be read.

CHAPTER 12 SERIAL INTERFACE (UART0)

12.1 Serial Interface Functions

The serial interface (UART0) has the following two modes.

(1) Operation stop mode

This mode is used when serial transfers are not performed. If can therefore be used to reduce power consumption.

For details, see **12.4.1 Operation stop mode**.

(2) Asynchronous serial interface (UART) mode

This mode enables full-duplex operation wherein one byte of data starting with the start bit is transmitted and received.

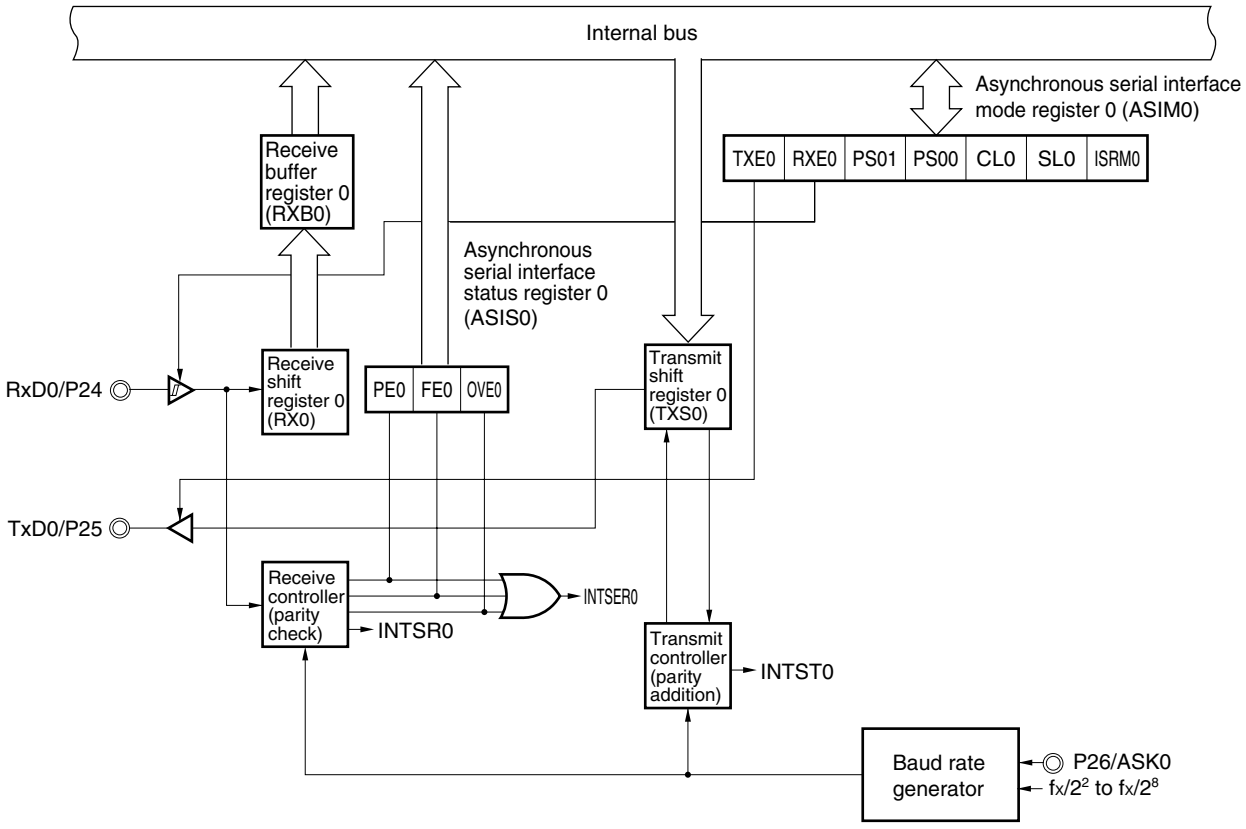
The on-chip UART-dedicated baud rate generator enables communication using a wide range of selectable baud rates. In addition, a baud rate can also be defined by dividing clocks input to the ASCK0 pin.

The UART baud rate generator can also be used to generate a MIDI-standard baud rate (31.25 kbps).

For details, see **12.4.2 Asynchronous serial interface (UART) mode**.

Figure 12-1 shows a block diagram of the serial interface (UART0) macro.

Figure 12-1. Block Diagram of Serial Interface (UART0)



12.2 Serial Interface Configuration

The serial interface (UART0) consists of the following hardware.

Table 12-1. Configuration of Serial Interface (UART0)

Item	Configuration
Registers	Transmit shift register 0 (TXS0) Receive shift register 0 (RX0) Receive buffer register 0 (RXB0)
Control registers	Asynchronous serial interface mode register 0 (ASIM0) Asynchronous serial interface status register 0 (ASIS0) Baud rate generator control register 0 (BRGC0)

(1) Transmit shift register 0 (TXS0)

This is the register for setting transmit data. Data written to TXS0 is transmitted as serial data.

When the data length is set as 7 bits, bits 0 to 6 of the data written to TXS0 are transferred as transmit data.

Writing data to TXS0 starts the transmit operation.

TXS0 can be written with an 8-bit memory manipulation instruction. It cannot be read.

$\overline{\text{RESET}}$ input sets TXS0 to FFH.

Caution Do not write to TXS0 during a transmit operation.
The same address is assigned to TXS0 and receive buffer register 0 (RXB0). A read operation reads values from RXB0.

(2) Receive shift register 0 (RX0)

This register converts serial data input via the RxD0 pin to parallel data. When one byte of data is received at this register, the receive data is transferred to receive buffer register 0 (RXB0).

RX0 cannot be manipulated directly by a program.

(3) Receive buffer register 0 (RXB0)

This register is used to hold receive data. When one byte of data is received, one byte of new receive data is transferred from receive shift register 0 (RX0).

When the data length is set as 7 bits, receive data is sent to bits 0 to 6 of RXB0. In this case, the MSB of RXB0 always becomes 0.

RXB0 can be read with an 8-bit memory manipulation instruction. It cannot be written to.

$\overline{\text{RESET}}$ input sets RXB0 to FFH.

Caution The same address is assigned to RXB0 and transmit shift register 0 (TXS0). During a write operation, values are written to TXS0.

(4) Transmission control circuit

The transmission control circuit controls transmit operations, such as adding a start bit, parity bit, and stop bit to data that is written to transmit shift register 0 (TXS0), based on the values set to asynchronous serial interface mode register 0 (ASIM0).

(5) Reception control circuit

The reception control circuit controls receive operations based on the values set to asynchronous serial interface mode register 0 (ASIM0). During a receive operation, it performs error checking, such as for parity errors, and sets various values to asynchronous serial interface status register 0 (ASIS0) according to the type of error that is detected.

12.3 Serial Interface Control Registers

The following three types of registers are used to control the serial interface (UART0).

- Asynchronous serial interface mode register 0 (ASIM0)
- Asynchronous serial interface status register 0 (ASIS0)
- Baud rate generator control register 0 (BRGC0)

(1) Asynchronous serial interface mode register 0 (ASIM0)

This is an 8-bit register that controls the serial interface (UART0)'s serial transfer operations.

ASIM0 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIM0 to 00H.

Figure 12-2 shows the format of ASIM0.

Caution In UART mode, set the port mode register (PMXX) as follows. Set the output latch to 0.

- **During receive operation**
  Set P24 (RxD0) to input mode (PM24 = 1)
- **During transmit operation**
  Set P25 (TxD0) to output mode (PM25 = 0)
- **During transmit/receive operation**
  Set P24 (RxD0) to input mode, and P25 (RxD0) to output mode

Figure 12-2. Format of Asynchronous Serial Interface Mode Register 0 (ASIM0)

Address: FFA0H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0

TXE0	RXE0	Operation mode	RxD0/P24 pin function	TxD0/P25 pin function
0	0	Operation stop	Port function (P24)	Port function (P25)
0	1	UART mode (receive only)	Serial function (RxD0)	
1	0	UART mode (transmit only)	Port function (P24)	Serial function (TxD0)
1	1	UART mode (transmit and receive)	Serial function (RxD0)	

PS01	PS00	Parity bit specification
0	0	No parity
0	1	Zero parity always added during transmission No parity detection during reception (parity errors do not occur)
1	0	Odd parity
1	1	Even parity

CL0	Character length specification
0	7 bits
1	8 bits

SL0	Stop bit length specification for transmit data
0	1 bit
1	2 bits

ISRM0	Receive completion interrupt control when error occurs
0	Receive completion interrupt request is issued when an error occurs
1	Receive completion interrupt request is not issued when an error occurs

Cautions 1. Always set bit 0 to 0.

2. Do not switch the operation mode until the current serial transmit/receive operation has stopped.

(2) Asynchronous serial interface status register 0 (ASIS0)

When a receive error occurs in UART mode, this register indicates the type of error.

ASIS0 can be read with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIS0 to 00H.

Figure 12-3. Format of Asynchronous Serial Interface Status Register 0 (ASIS0)

Address: FFA1H After reset: 00H R

Symbol	7	6	5	4	3	2	1	0
ASIS0	0	0	0	0	0	PE0	FE0	OVE0

PE0	Parity error flag
0	No parity error
1	Parity error (Transmit data's parity does not match)

FE0	Framing error flag
0	No framing error
1	Framing error**Note 1** (Stop bit not detected)

OVE0	Overrun error flag
0	No overrun error
1	Overrun error**Note 2** (Next receive operation was completed before data was read from receive buffer register)

Notes 1. Even if a stop bit length is set to two bits by setting bit 2 (SL0) of asynchronous serial interface mode register 0 (ASIM0), stop bit detection during a receive operation only applies to a stop bit length of 1 bit.

2. Be sure to read the contents of receive buffer register 0 (RXB0) when an overrun error has occurred. Until the contents of RXB0 are read, further overrun errors will occur when receiving data.

(3) Baud rate generator control register 0 (BRGC0)

This register sets the serial clock for serial interface.

BRGC0 can be set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC0 to 00H.

Figure 12-4 shows the format of BRGC0.

Figure 12-4. Format of Baud Rate Generator Control Register 0 (BRGC0)

Address: FFA2H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00

(f_X = 6.29 MHz)

TPS02	TPS01	TPS00	Source clock (f_{SCK}) selection for 5-bit counter	n
0	0	0	P26/ASCK0	0
0	0	1	$f_X/2^2$	2
0	1	0	$f_X/2^3$	3
0	1	1	$f_X/2^4$	4
1	0	0	$f_X/2^5$	5
1	0	1	$f_X/2^6$	6
1	1	0	$f_X/2^7$	7
1	1	1	$f_X/2^8$	8

MDL03	MDL02	MDL01	MDL00	Input clock selection for baud rate generator	k
0	0	0	0	f_{SCK}/16	0
0	0	0	1	f_{SCK}/17	1
0	0	1	0	f_{SCK}/18	2
0	0	1	1	f_{SCK}/19	3
0	1	0	0	f_{SCK}/20	4
0	1	0	1	f_{SCK}/21	5
0	1	1	0	f_{SCK}/22	6
0	1	1	1	f_{SCK}/23	7
1	0	0	0	f_{SCK}/24	8
1	0	0	1	f_{SCK}/25	9
1	0	1	0	f_{SCK}/26	10
1	0	1	1	f_{SCK}/27	11
1	1	0	0	f_{SCK}/28	12
1	1	0	1	f_{SCK}/29	13
1	1	1	0	f_{SCK}/30	14
1	1	1	1	Setting prohibited	—

Caution Writing to BRGC0 during a communication operation may cause abnormal output from the baud rate generator and further communication will perform abnormally. Therefore, do not write to BRGC0 during a communication operation.

Remarks 1. f_{SCK}: Source clock for 5-bit counter

2. n: Value set via TPS00 to TPS02 (0, $2 \leq n \leq 8$)

3. k: Value set via MDL00 to MDL03 ($0 \leq k \leq 14$)

12.4 Serial Interface Operations

This section explains the two modes of the serial interface (UART0).

12.4.1 Operation stop mode

Because serial transfer is not performed during this mode, the power consumption can be reduced.
In addition, pins can be used as ordinary ports.

(1) Register settings

Operation stop mode is set by asynchronous serial interface mode register 0 (ASIM0).
ASIM0 can be set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets ASIM0 to 00H.

Address: FFA0H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0

TXE0	RXE0	Operation mode	RxD0/P24 pin function	TxD0/P25 pin function
0	0	Operation stop	Port function (P24)	Port function (P25)
0	1	UART mode (receive only)	Serial function (RxD0)	
1	0	UART mode (transmit only)	Port function (P24)	Serial function (TxD0)
1	1	UART mode (transmit and receive)	Serial function (RxD0)	

Caution Do not switch the operation mode until the current serial transmit/receive operation has stopped.

12.4.2 Asynchronous serial interface (UART) mode

This mode enables full-duplex operation wherein one byte of data starting with the start bit is transmitted or received.

The on-chip UART-dedicated baud rate generator enables communications using a wide range of selectable baud rates.

The UART baud rate generator can also be used to generate a MIDI-standard baud rate (31.25 kbps).

(1) Register settings

UART mode settings are performed by asynchronous serial interface mode register 0 (ASIM0), asynchronous serial interface status register 0 (ASIS0), and baud rate generator control register 0 (BRGC0).

(a) Asynchronous serial interface mode register 0 (ASIM0)

ASIM0 can be set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets ASIM0 to 00H.

Caution In UART mode, set the port mode register (PMXX) as follows. Set the output latch to 0.

- **During receive operation**
  Set P24 (RxD0) to input mode (PM24 = 1)
- **During transmit operation**
  Set P25 (TxD0) to output mode (PM25 = 0)
- **During transmit/receive operation**
  Set P24 (RxD0) to input mode, and P25 (TxD0) to output mode

Address: FFA0H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0

TXE0	RXE0	Operation mode	RxD0/P24 pin function	TxD0/P25 pin function
0	0	Operation stop	Port function (P24)	Port function (P25)
0	1	UART mode (receive only)	Serial function (RxD0)	
1	0	UART mode (transmit only)	Port function (P24)	Serial function (TxD0)
1	1	UART mode (transmit and receive)	Serial function (RxD0)	

PS01	PS00	Parity bit specification
0	0	No parity
0	1	Zero parity always added during transmission No parity detection during reception (parity errors do not occur)
1	0	Odd parity
1	1	Even parity

CL0	Character length specification
0	7 bits
0	8 bits

SL0	Stop bit length specification for transmit data
0	1 bit
1	2 bits

ISRM0	Receive completion interrupt control when error occurs
0	Receive completion interrupt request is issued when an error occurs
1	Receive completion interrupt request is not issued when an error occurs

Cautions 1. Always set bit 0 to 0.

2. Do not switch the operation mode until the current serial transmit/receive operation has stopped.

(b) Asynchronous serial interface status register 0 (ASIS0)

ASIS0 can be read with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIS0 to 00H.

Address: FFA1H After reset: 00H R

Symbol	7	6	5	4	3	2	1	0
ASIS0	0	0	0	0	0	PE0	FE0	OVE0

PE0	Parity error flag
0	No parity error
1	Parity error (Transmit data's parity does not match)

FE0	Framing error flag
0	No framing error
1	Framing error[Note 1] (Stop bit not detected)

OVE0	Overrun error flag
0	No overrun error
1	Overrun error[Note 2] (Next receive operation was completed before data was read from receive buffer register)

Notes 1. Even if a stop bit length is set to two bits by setting bit 2 (SL0) of asynchronous serial interface mode register 0 (ASIM0), stop bit detection during a receive operation only applies to a stop bit length of 1 bit.

2. Be sure to read the contents of receive buffer register 0 (RXB0) when an overrun error has occurred.

Until the contents of RXB0 are read, further overrun errors will occur when receiving data.

(c) Baud rate generator control register 0 (BRGC0)

BRGC0 can be set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC0 to 00H.

Address: FFA2H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00

(f_X = 6.29 MHz)

TPS02	TPS01	TPS00	Source clock selection for 5-bit counter	n
0	0	0	P26/ASCK0	0
0	0	1	$f_X/2^2$	2
0	1	0	$f_X/2^3$	3
0	1	1	$f_X/2^4$	4
1	0	0	$f_X/2^5$	5
1	0	1	$f_X/2^6$	6
1	1	0	$f_X/2^7$	7
1	1	1	$f_X/2^8$	8

MDL03	MDL02	MDL01	MDL00	Input clock selection for baud rate generator	k
0	0	0	0	f_{SCK}/16	0
0	0	0	1	f_{SCK}/17	1
0	0	1	0	f_{SCK}/18	2
0	0	1	1	f_{SCK}/19	3
0	1	0	0	f_{SCK}/20	4
0	1	0	1	f_{SCK}/21	5
0	1	1	0	f_{SCK}/22	6
0	1	1	1	f_{SCK}/23	7
1	0	0	0	f_{SCK}/24	8
1	0	0	1	f_{SCK}/25	9
1	0	1	0	f_{SCK}/26	10
1	0	1	1	f_{SCK}/27	11
1	1	0	0	f_{SCK}/28	12
1	1	0	1	f_{SCK}/29	13
1	1	1	0	f_{SCK}/30	14
1	1	1	1	Setting prohibited	—

Caution Writing to BRGC0 during a communication operation may cause abnormal output from the baud rate generator and further communication will perform abnormally. Therefore, do not write to BRGC0 during a communication operation.

Remarks 1. f_{SCK}: Source clock for 5-bit counter

2. n: Value set via TPS00 to TPS02 ($0, 2 \leq n \leq 8$)

3. k: Value set via MDL00 to MDL03 ($0 \leq k \leq 14$)

The transmit/receive clock that is used for the baud rate to be generated is obtained by dividing the system clock.

- Transmit/receive clock generation for baud rate by using system clock
  The system clock is divided to generate the transmit/receive clock. The baud rate generated from the system clock is determined according to the following formula.

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1}(k + 16)} \quad [\text{Hz}]$$

  f_X: Oscillation frequency of system clock
  n: Value set via TPS00 to TPS02 (0, $2 \leq n \leq 8$)
  For details, see Table 12-2.
  k: Value set via MDL00 to MDL03 ($0 \leq k \leq 14$)

Table 12-2 shows the relationship between the 5-bit counter's source clock assigned to bits 4 to 6 (TPS00 to TPS02) of BRGC0 and the "n" value in the above formula.

Table 12-2. Relationship between 5-Bit Counter's Source Clock and "n" Value

TPS02	TPS01	TPS00	5-bit Counter's Source Clock Selected	n
0	0	0	P26/ASCK0	0
0	0	1	$f_X/2^2$	2
0	1	0	$f_X/2^3$	3
0	1	1	$f_X/2^4$	4
1	0	0	$f_X/2^5$	5
1	0	1	$f_X/2^6$	6
1	1	0	$f_X/2^7$	7
1	1	1	$f_X/2^8$	8

Remark f_X: Oscillation frequency of system clock

- **Error tolerance range for baud rates**

  The tolerance range for baud rates depends on the number of bits per frame and the counter's division rate [1/(16 + k)].

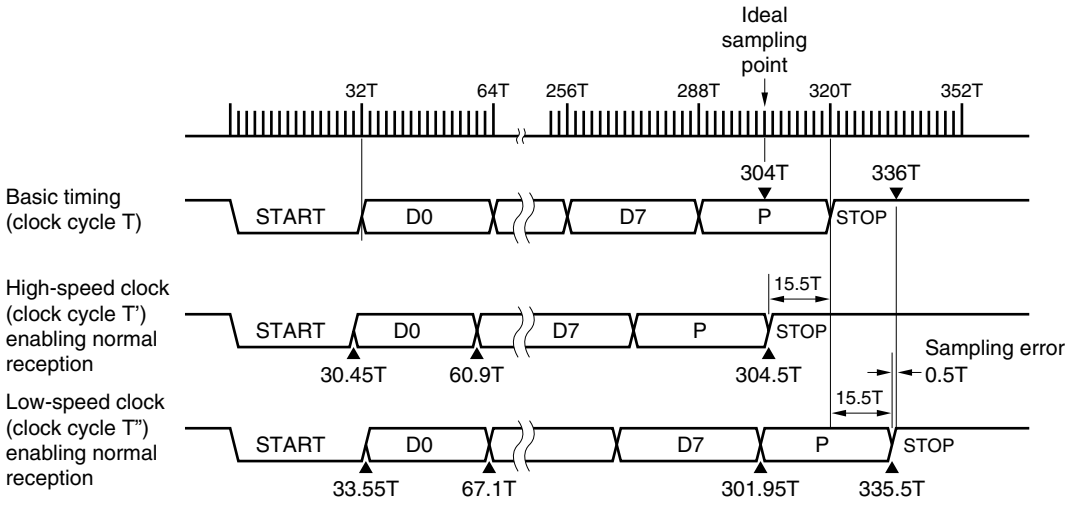
  Table 12-3 describes the relationship between the system clock and the baud rate and Figure 12-5 shows an example of a baud rate error tolerance range.

Table 12-3. Relationship between System Clock and Baud Rate

Baud Rate (bps)	f_X = 6.291456 MHz	
	BRGC0	ERR (%)
600	74H	2.40
1,200	64H	2.40
2,400	54H	2.40
4,800	44H	2.40
9,600	34H	2.40
19,200	24H	2.40
31,250	19H	0.66
38,400	14H	2.40
38,400	15H	–2.34

Remark f_X: Oscillation frequency of system clock
k: Value set via MDL00 to MDL03 ($0 \leq k \leq 14$)

Figure 12-5. Error Tolerance (When k = 0), Including Sampling Errors



Remark T: 5-bit counter's source clock cycle

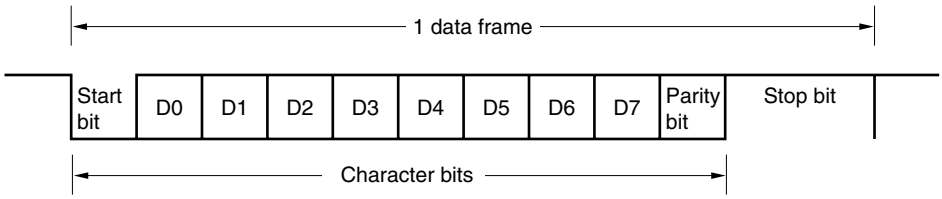
$$\text{Baud rate error tolerance (when k = 0)} = \frac{\pm 15.5}{320} \times 100 = 4.8438\ (\%)$$

(2) Communication operations

(a) Data format

Figure 12-6 shows the format of the transmit/receive data.

Figure 12-6. Format of Transmit/Receive Data in Asynchronous Serial Interface



1 data frame consists of the following bits.

- Start bit ..... 1 bit
- Character bits ... 7 bits or 8 bits
- Parity bit ..... Even parity, odd parity, zero parity, or no parity
- Stop bit(s) ..... 1 bit or 2 bits

Asynchronous serial interface mode register 0 (ASIM0) is used to set the character bit length, parity selection, and stop bit length within each data frame.

When “7 bits” is selected as the number of character bits, only the low-order 7 bits (bits 0 to 6) are valid, so that during a transmission the highest bit (bit 7) is ignored and during reception the highest bit (bit 7) must be set to 0.

ASIM0 and baud rate generator control register 0 (BRGC0) are used to set the serial transfer rate.

If a receive error occurs, information about the receive error can be recognized by reading asynchronous serial interface status register 0 (ASIS0).

(b) Parity types and operations

The parity bit is used to detect bit errors in transfer data. Usually, the same type of parity bit is used by the transmitting and receiving sides. When odd parity or even parity is set, errors in the parity bit (the odd-number bit) can be detected. When zero parity or no parity is set, errors are not detected.

(i) Even parity

- During transmission
  The number of bits in transmit data that includes a parity bit is controlled so that there are an even number of bits whose value is 1. The value of the parity bit is as follows.

  If the transmit data contains an odd number of bits whose value is 1: the parity bit is 1
  If the transmit data contains an even number of bits whose value is 1: the parity bit is 0

- During reception
  The number of bits whose value is 1 is counted among the transfer data that include a parity bit, and a parity error occurs when the counted result is an odd number.

(ii) Odd parity

- During transmission
  The number of bits in transmit data that includes a parity bit is controlled so that there is an odd number of bits whose value is 1. The value of the parity bit is as follows.

  If the transmit data contains an odd number of bits whose value is 1: the parity bit is 0
  If the transmit data contains an even number of bits whose value is 1: the parity bit is 1

- During reception
  The number of bits whose value is 1 is counted among the transfer data that include a parity bit, and a parity error occurs when the counted result is an even number.

(iii) Zero parity

During transmission, the parity bit is set to 0 regardless of the transmit data.
During reception, the parity bit is not checked. Therefore, no parity errors will occur regardless of whether the parity bit is a 0 or a 1.

(iv) No parity

No parity bit is added to the transmit data.
During reception, receive data is regarded as having no parity bit. Since there is no parity bit, no parity errors will occur.

(c) Transmission

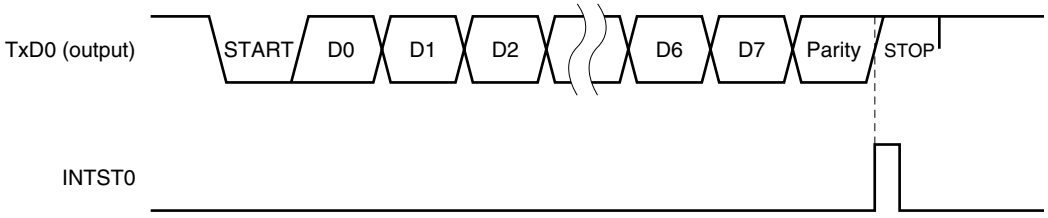
The transmit operation is started when transmit data is written to transmit shift register 0 (TXS0). A start bit, parity bit, and stop bit are automatically added to the data.

Starting the transmit operation shifts out the data in TXS0, thereby emptying TXS0, after which a transmit completion interrupt request (INTST0) is issued.

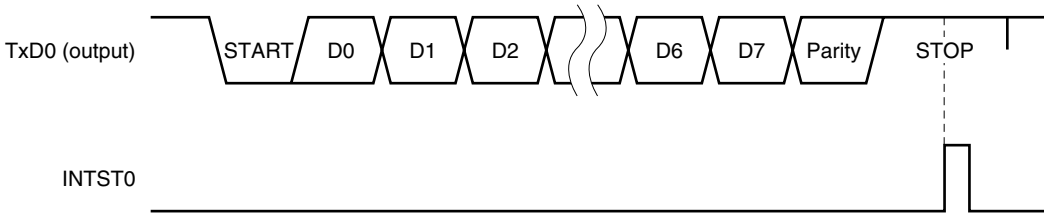
The timing of the transmit completion interrupt request is shown in Figure 12-7.

Figure 12-7. Timing of Asynchronous Serial Interface Transmit Completion Interrupt Request

(i) Stop bit length: 1 bit



(ii) Stop bit length: 2 bits



Caution Do not rewrite asynchronous serial interface mode register 0 (ASIM0) during a transmit operation. Rewriting the ASIM0 register during a transmit operation may disable further transmit operations (in such cases, enter $\overline{\text{RESET}}$ to restore normal operation).
Whether or not a transmit operation is in progress can be determined via software using the transmit completion interrupt request (INTST0) or the interrupt request flag (STIF0) that is set by INTST0.

(d) Reception

The receive operation is enabled when bit 6 (RXE0) of asynchronous serial interface mode register 0 (ASIM0) is set to 1, and input via the RxD0 pin is sampled.

The serial clock specified by ASIM0 is used to sample the RxD0 pin.

When the RxD0 pin input goes low, the 5-bit counter of the baud rate generator begins counting and the start timing signal for data sampling is output when half of the specified baud rate time has elapsed. If sampling the RxD0 pin input with this start timing signal yields a low-level result, a start bit is recognized, after which the 5-bit counter is initialized and starts counting, and data sampling begins. After the start bit is recognized, the character data, parity bit, and one-bit stop bit are detected, at which point reception of one data frame is completed.

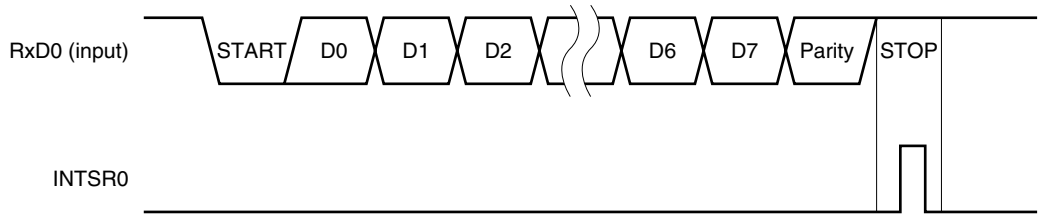
Once reception of one data frame is completed, the receive data in the shift register is transferred to receive buffer register 0 (RXB0) and a receive completion interrupt request (INTSR0) occurs.

Even if an error has occurred, the receive data in which the error occurred is still transferred to RXB0. When ASIM0 bit 1 (ISRM0) is cleared to 0 upon occurrence of an error, INTSR0 occurs (see Figure 12-9). When the ISRM0 bit is set to 1, INTSR0 does not occur.

If the RXE0 bit is reset to 0 during a receive operation, the receive operation is stopped immediately. At this time, the contents of RXB0 and ASIS0 do not change, nor does INTSR0 or INTSER0 occur.

Figure 12-8 shows the timing of the asynchronous serial interface receive completion interrupt request.

Figure 12-8. Timing of Asynchronous Serial Interface Receive Completion Interrupt Request



Caution Be sure to read the contents of receive buffer register 0 (RXB0) even when a receive error has occurred. Overrun errors will occur during the next data receive operations and the receive error status will remain until the contents of RXB0 are read.

(e) Receive errors

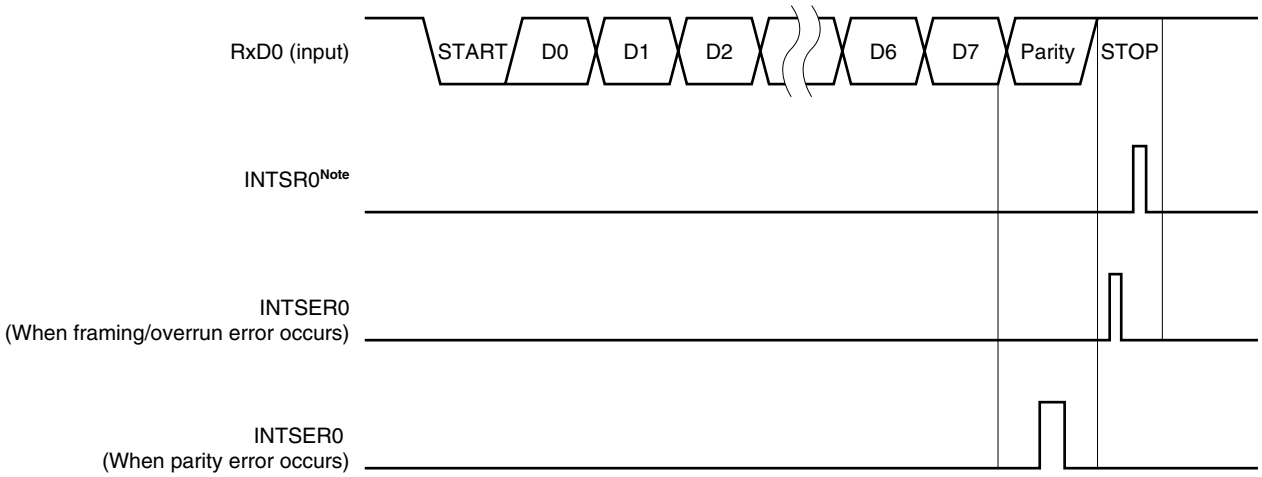
Three types of errors can occur during a receive operation: a parity error, framing error, or overrun error. If, as the result of data reception, an error flag in asynchronous serial interface status register 0 (ASIS0) is set, a receive error interrupt request (INTSER0) will occur. The receive error interrupt request is generated before the receive completion interrupt request (INTSR0). Table 12-4 lists the causes of receive errors. As part of receive error interrupt request (INTSER0) servicing, the contents of ASIS0 can be read to determine which type of error occurred during the receive operation (see Table 12-4 and Figure 12-9).

The contents of ASIS0 are reset to 0 when receive buffer register 0 (RXB0) is read or when the next data is received (if the next data contains an error, its error flag will be set).

Table 12-4. Causes of Receive Errors

Receive Error	Cause	ASIS0 Value
Parity error	Parity specified during transmission does not match parity of receive data	04H
Framing error	Stop bit was not detected	02H
Overrun error	Reception of the next data was completed before data was read from the receive buffer register	01H

Figure 12-9. Receive Error Timing



Note If a receive error occurs when the ISRM0 bit has been set to 1, INTSR0 does not occur.

Cautions 1. The contents of asynchronous serial interface status register 0 (ASIS0) are reset to 0 when receive buffer register 0 (RXB0) is read or when the next data is received. To obtain information about the error, be sure to read the contents of ASIS0 before reading RXB0.

2. Be sure to read the contents of receive buffer register 0 (RXB0) even when a receive error has occurred. Overrun errors will occur during the subsequent data receive operations and the receive error status will remain until the contents of RXB0 are read.

CHAPTER 13 SERIAL INTERFACE (SIO30, SIO31)

The μPD780701Y Subseries incorporates two 3-wire serial I/O mode channels (SIO30, SIO31).

These two channels have exactly the same functions.

Therefore, unless otherwise specified, the SIO30 is used throughout this chapter to describe the functions of both the SIO30 and SIO31. If using the SIO31, refer to Table 13-1 for the register, bit, and pin names.

Table 13-1. SIO30 and SIO31 Naming Differences

Item	SIO30	SIO31
Pins	P30/SI30 P31/SO30 P32/$\overline{\text{SCK30}}$	P20/SI31 P21/SO31 P22/$\overline{\text{SCK31}}$
Serial operation mode register 3	CSIM30	CSIM31
Address of serial operation mode register 3	FFB0H	FFB8H
Bit name of serial operation mode register 3	CSIE30 MODE0 SCL301 SCL300	CSIE31 MODE1 SCL311 SCL310
Serial I/O shift register 3	SIO30	SIO31
Address of serial I/O shift register 3	FF1AH	FF1BH
Interrupt request	INTCSI30	INTCSI31
Names of interrupt control registers and bits mentioned in this chapter	CSIIF30 CSIMK30 CSIPR30	CSIIF31 CSIMK31 CSIPR31

13.1 Serial Interface Functions

The serial interface (SIO30) has the following two modes.

(1) Operation stop mode

This mode is used when serial transfers are not performed. For details, see **13.4.1 Operation stop mode**.

(2) 3-wire serial I/O mode (fixed as MSB first)

This is an 8-bit data transfer mode using three lines: a serial clock line ($\overline{\text{SCK30}}$), serial output line (SO30), and serial input line (SI30).

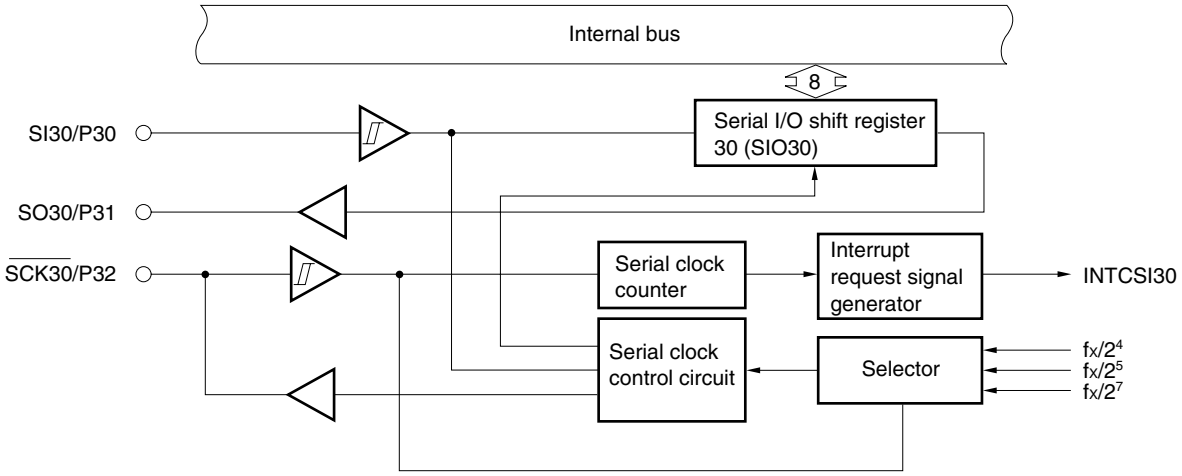
Since simultaneous transmit and receive operations are enabled in 3-wire serial I/O mode, the processing time for data transfers is reduced.

The first bit of the 8-bit data to be serial transferred is fixed as the MSB.

3-wire serial I/O mode is useful for connection to a peripheral I/O incorporating a clock-synchronous serial interface, or a display controller, etc. For details, see **13.4.2 3-wire serial I/O mode**.

Figure 13-1 shows a block diagram of the serial interface (SIO30).

Figure 13-1. Block Diagram of Serial Interface (SIO30)



13.2 Serial Interface Configuration

The serial interface (SIO30) consists of the following hardware.

Table 13-2. Configuration of Serial Interface (SIO30)

Item	Configuration
Register	Serial I/O shift register 30 (SIO30)
Control register	Serial operation mode register 30 (CSIM30)

(1) Serial I/O shift register 30 (SIO30)

This is an 8-bit register that performs parallel-serial conversion and serial transmit/receive (shift operations) synchronized with the serial clock.

SIO30 is set with an 8-bit memory manipulation instruction.

When bit 7 (CSIE30) of serial operation mode register 30 (CSIM30) is set to 1, a serial operation can be started by writing data to or reading data from SIO30.

When transmitting, data written to SIO30 is output to the serial output (SO30).

When receiving, data is read from the serial input (SI30) and written to SIO30.

$\overline{\text{RESET}}$ input sets SIO30 to 00H.

Caution Do not access SIO30 during a transmit operation unless the access is triggered by a transfer start. (Read operation is disabled when MODE0 = 0 and write operation is disabled when MODE0 = 1.)

13.3 Serial Interface Control Registers

The following type of register is used to control the serial interface (SIO30).

- Serial operation mode register 30 (CSIM30)

(1) Serial operation mode register 30 (CSIM30)

This register is used to set SIO30's serial clock, operation modes, and to enable or disable specific operations.

CSIM30 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CSIM30 to 00H.

Caution In 3-wire serial I/O mode, set the port mode register (PMXX) as follows. Set the output latch to 0.

- **During serial clock output (master transmission or master reception)**
  Set P32 ($\overline{\text{SCK30}}$) to output mode (PM32 = 0)
- **During serial clock input (slave transmission or slave reception)**
  Set P32 to input mode (PM32 = 1)
- **During transmit/transmit and receive mode**
  Set P31 (SO30) to output mode (PM31 = 0)
- **During receive mode**
  Set P30 (SI30) to input mode (PM30 = 1)

Figure 13-2. Format of Serial Operation Mode Register 30 (CSIM30)

Address: FFB0H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CSIM30	CSIE30	0	0	0	0	MODE0	SCL301	SCL300

CSIE30	Enable/disable specification for SIO30		
	Shift register operation	Serial counter	Port
0	Operation stop	Clear	Port function[Note 1]
1	Operation enable	Count operation enable	Serial function + port function[Note 2]

MODE0	Transfer operation mode flag		
	Operation mode	Transfer start trigger	SO30 output
0	Transmit/transmit and receive mode	Write to SIO30	Normal output
1	Receive-only mode	Read from SIO30	Fixed at low level

SCL301	SCL300	Clock selection
0	0	External clock input to $\overline{\text{SCK30}}$
0	1	$f_X/2^4$ (393 kHz)
1	0	$f_X/2^5$ (197 kHz)
1	1	$f_X/2^7$ (49.2 kHz)

Notes 1. When CSIE30 = 0 (SIO30 operation stop status), the pins SI30, SO30, and $\overline{\text{SCK30}}$ can be used for port functions.

2. When CSIE30 = 1 (SIO30 operation enabled state), the SI30 pin can be used as a port pin if only the transmit function is used, and the SO30 pin can be used as a port pin if only the receive-only mode is used.

Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

13.4 Serial Interface Operations

This section explains the two modes of serial interface (SIO30).

13.4.1 Operation stop mode

Because serial transfer is not performed during this mode, the power consumption can be reduced.
In addition, pins can be used as normal I/O ports.

(1) Register settings

Operation stop mode is set by serial operation mode register 30 (CSIM30).
CSIM30 can be set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CSIM30 to 00H.

Address: FFB0H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CSIM30	CSIE30	0	0	0	0	MODE0	SCL301	SCL300

CSIE30	SIO30 operation enable/disable specification		
	Shift register operation	Serial counter	Port
0	Operation stop	Clear	Port function[Note 1]
1	Operation enable	Count operation enable	Serial function + port function[Note 2]

Notes 1. When CSIE30 = 0 (SIO30 operation stop status), the pins SI30, SO30, and $\overline{\text{SCK30}}$ can be used for port functions.

2. When CSIE30 = 1 (SIO30 operation enabled state), the SI30 pin can be used as a port pin if only the transmit function is used, and the SO30 pin can be used as a port pin if only the receive-only mode is used.

13.4.2 3-wire serial I/O mode

The 3-wire serial I/O mode is useful for connection to a peripheral I/O incorporating a clock-synchronous serial interface, a display controller, etc.

This mode executes data transfers via three lines: a serial clock line ($\overline{SCK30}$), serial output line (SO30), and serial input line (SI30).

(1) Register settings

3-wire serial I/O mode is set by serial operation mode register 30 (CSIM30).

CSIM30 can be set with a 1-bit or 8-bit memory manipulation instructions.

$\overline{RESET}$ input sets CSIM30 to 00H .

Caution In 3-wire serial I/O mode, set the port mode register (PMXX) as follows. Set the output latch to 0.

- **During serial clock output (master transmission or master reception)**
  Set P32 ($\overline{SCK30}$) to output mode (PM32 = 0)
- **During serial clock input (slave transmission or slave reception)**
  Set P32 to input mode (PM32 = 1)
- **During transmit/transmit and receive mode**
  Set P31 (SO30) to output mode (PM31 = 0)
- **During receive mode**
  Set P30 (SI30) to input mode (PM30 = 1)

Address: FFB0H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CSIM30	CSIE30	0	0	0	0	MODE0	SCL301	SCL300

CSIE30	Enable/disable specification for SIO30		
	Shift register operation	Serial counter	Port
0	Operation stop	Clear	Port function[Note 1]
1	Operation enable	Count operation enable	Serial function + port function[Note 2]

MODE0	Transfer operation mode flag		
	Operation mode	Transfer start trigger	SO30 output
0	Transmit/transmit and receive mode	Write to SIO30	Normal output
1	Receive-only mode	Read from SIO30	Fixed at low level

SCL301	SCL300	Clock selection
0	0	External clock input to $\overline{SCK30}$
0	1	$f_X/2^4$ (393 kHz)
1	0	$f_X/2^5$ (197 kHz)
1	1	$f_X/2^7$ (49.2 kHz)

Notes 1. When CSIE30 = 0 (SIO30 operation stop status), the pins SI30, SO30, and $\overline{SCK30}$ can be used for port functions.

2. When CSIE30 = 1 (SIO30 operation enabled state), the SI30 pin can be used as a port pin if only the transmit function is used, and the SO30 pin can be used as a port pin if only the receive-only mode is used.

Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

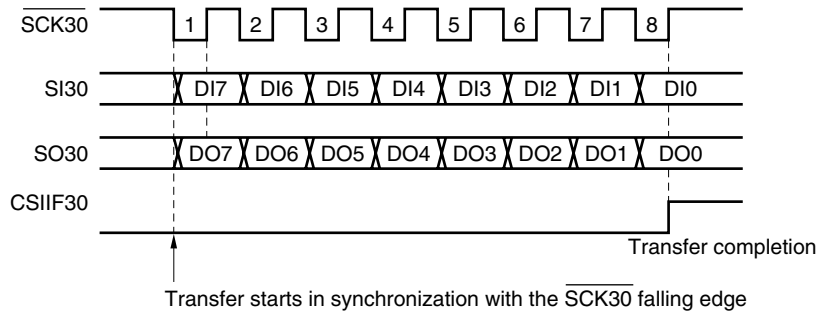
(2) Communication Operations

In 3-wire serial I/O mode, data is transmitted and received in 8-bit units. Each bit of data is transmitted or received in synchronization with the serial clock.

Serial I/O shift register 30 (SIO30) is shifted in synchronization with the falling edge of the serial clock. Transmission data is held in the SO30 latch and is output from the SO30 pin. Data that is received via the SI30 pin in synchronization with the rising edge of the serial clock is latched to SIO30.

Completion of an 8-bit transfer automatically stops operation of SIO30 and sets an interrupt request flag (CSIIF30).

Figure 13-3. Timing of 3-Wire Serial I/O Mode



(3) Transfer start

A serial transfer starts when the following two conditions have been satisfied and transfer data has been set (or read) to serial I/O shift register 30 (SIO30).

- The SIO30 operation control bit (CSIE30) = 1
- After an 8-bit serial transfer, either the internal serial clock is stopped or $\overline{\text{SCK30}}$ is set to high level.

Transmit/transmit and receive mode

When CSIE30 = 1 and MODE0 = 0, transfer starts when writing to SIO30.

Receive-only mode

When CSIE30 = 1 and MODE0 = 1, transfer starts when reading from SIO30.

Caution After data has been written to SIO30, transfer will not start even if the CSIE30 bit value is set to 1.

Completion of an 8-bit transfer automatically stops the serial transfer operation and an interrupt request flag (CSIIF30) is set.

CHAPTER 14 SERIAL INTERFACE (IIC0)

14.1 Serial Interface Functions

The serial interface (IIC0) has the following two modes.

(1) Operation stop mode

This mode is used when serial transfers are not performed. It can therefore be used to reduce power consumption.

(2) I^2C bus mode (multimaster supported)

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock (SCL0) line and a serial data bus (SDA0) line.

This mode complies with the I^2C bus format and can output “start condition”, “data”, and “stop condition” data segments when transmitting via the serial data bus. These data segments are automatically detected by hardware during reception.

Since SCL0 and SDA0 are open-drain outputs, the IIC0 requires pull-up resistors for the serial clock line (SCL0) and the serial data bus line (SDA0).

Figure 14-1 shows a block diagram of the serial interface (IIC0).

Figure 14-1. Block Diagram of Serial Interface (IIC0)

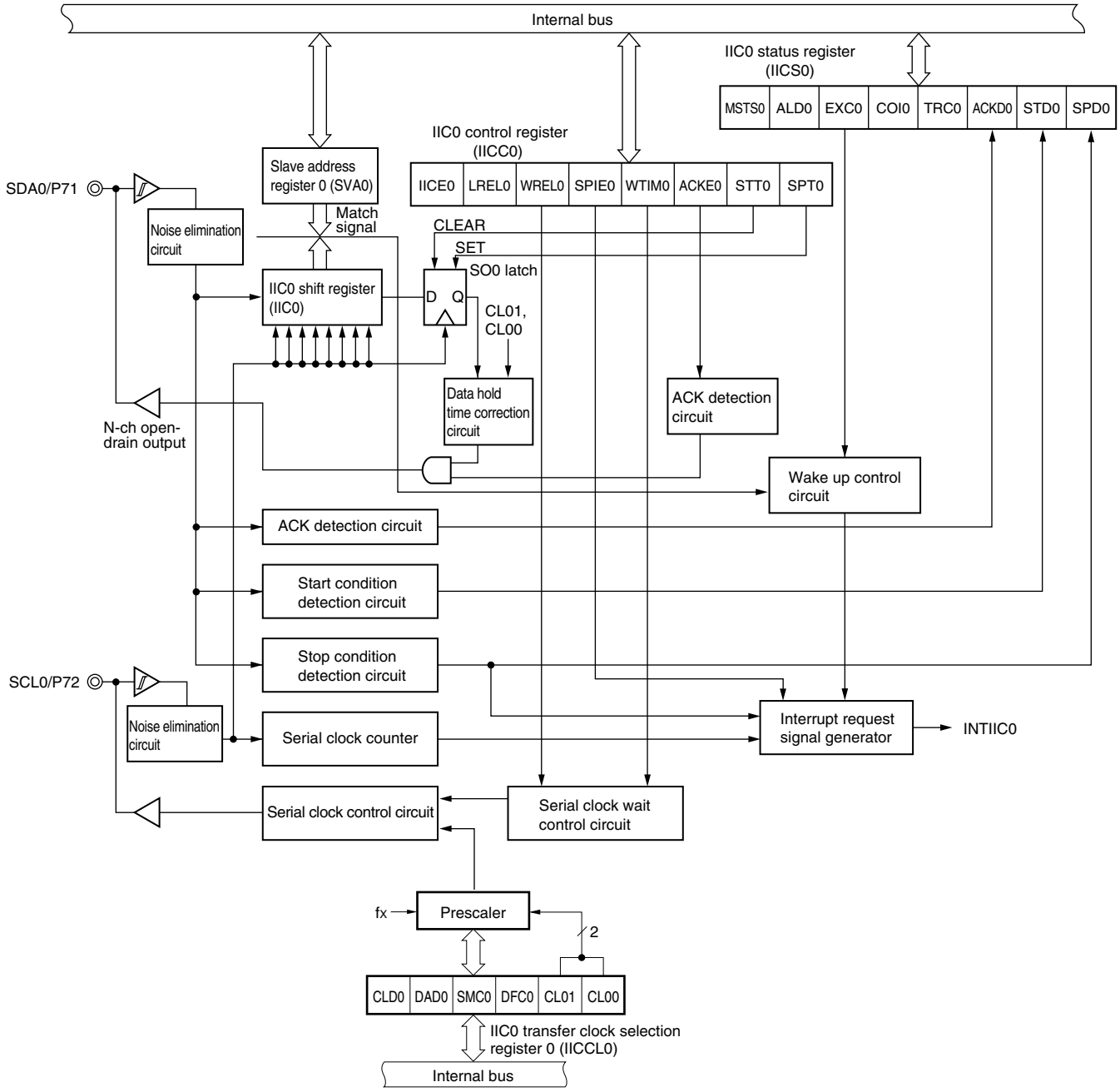
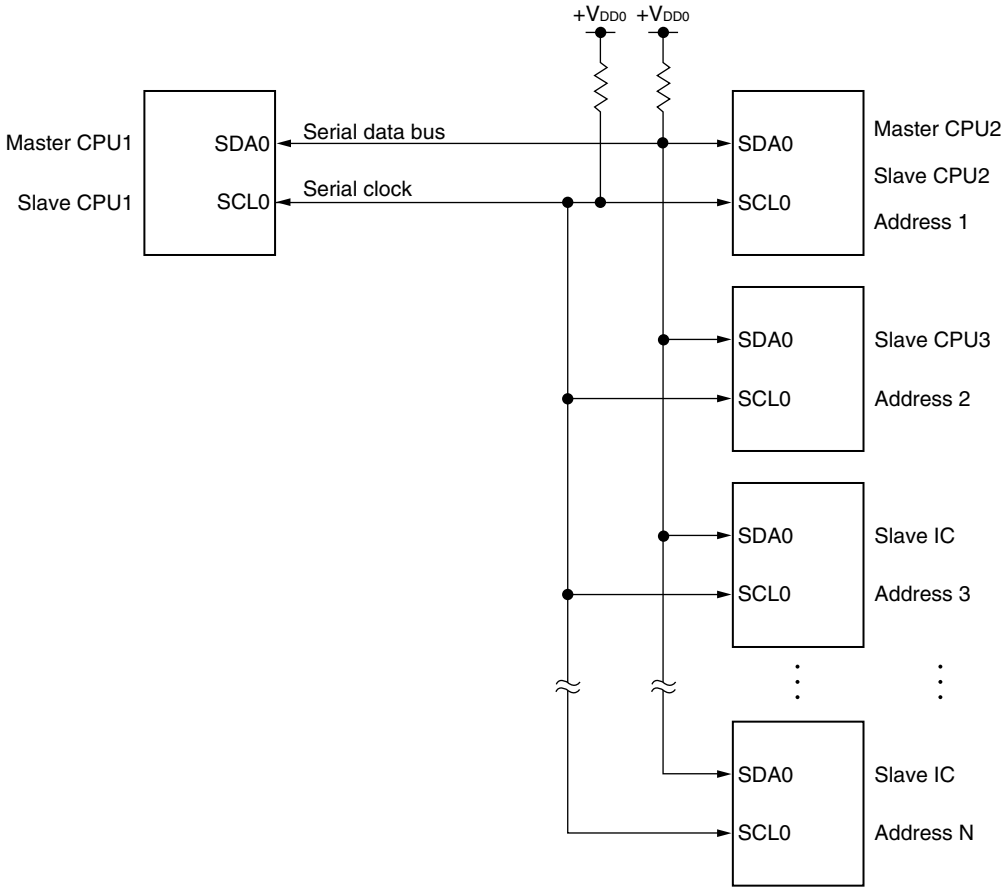


Figure 14-2 shows a serial bus configuration example.

Figure 14-2. Example of Serial Bus Configuration Using I²C Bus



14.2 Serial Interface Configuration

The serial interface (IIC0) consists of the following hardware.

Table 14-1. Configuration of Serial Interface (IIC0)

Item	Configuration
Registers	IIC0 shift register (IIC0) Slave address register 0 (SVA0)
Control registers	IIC0 control register (IICC0) IIC0 status register (IICS0) IIC0 transfer clock select register (IICCL0)

(1) IIC0 shift register (IIC0)

IIC0 is used to convert 8-bit serial data to 8-bit parallel data and to convert 8-bit parallel data to 8-bit serial data.

IIC0 can be used for both transmission and reception.

Write and read operations to and from IIC0 are used to control the actual transmit and receive operations.

IIC0 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets IIC0 to 00H.

(2) Slave address register 0 (SVA0)

This register sets local addresses when in slave mode.

SVA0 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets SVA0 to 00H.

(3) SO0 latch

The SO0 latch is used to retain the SDA0 pin's output level.

(4) Wake-up control circuit

This circuit generates an interrupt request when the address received by this register matches the address value set to slave address register 0 (SVA0) or when an extension code is received.

(5) Clock selector

This selects the sampling clock to be used.

(6) Serial clock counter

This counter counts the serial clocks that are output or input during transmit/receive operations and is used to verify that 8-bit data was transmitted or received.

(7) Interrupt request signal generator

This circuit controls the generation of interrupt request signals (INTIIC0).

An I^2C interrupt request is generated following either of two triggers.

- Eighth or ninth clock of the serial clock (set by the WTIM0 bit**Note**)
- Interrupt request generated when a stop condition is detected (set by the SPIE0 bit**Note**)

Note WTIM0 bit: bit 3 of the IIC0 control register (IICC0)
SPIE0 bit: bit 4 of the IIC0 control register (IICC0)

(8) Serial clock control circuit

During master mode, this circuit generates the clock output via the SCL0 pin from a sampling clock.

(9) Serial clock wait control circuit

This circuit controls the wait timing.

(10) ACK output circuit, stop condition detection circuit, start condition detection circuit, and ACK detection circuit

These circuits are used to output and detect various control signals.

(11) Data hold time correction circuit

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

14.3 Serial Interface Control Registers

The following three types of registers are used to control the serial interface (IIC0).

- IIC0 control register (IICC0)
- IIC0 status register (IICS0)
- IIC0 transfer clock select register (IICCL0)

The following registers are also used.

- IIC0 shift register (IIC0)
- Slave address register 0 (SVA0)

(1) IIC0 control register (IICC0)

This register is used to enable/disable I^2C operations, set wait timing, and set other I^2C operations.

IICC0 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets IICC0 to 00H.

Caution In I^2C bus mode, set the port mode register (PMXX) as follows. Set the output latch to 0.

- **Set P71 (SDA0) to output mode (PM71 = 0)**
- **Set P72 (SCL0) to output mode (PM72 = 0)**

Figure 14-3. Format of IIC0 Control Register (IICC0) (1/3)

Address: FFA8H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
IICC0	IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0

IICE0	I^2C operation enable
0	Stops operation. Presets IIC0 status register (IICS0). Stops internal operation.
1	Enables operation.

Condition for clearing (IICE0 = 0)	Condition for setting (IICE0 = 1)
• Cleared by instruction • When $\overline{\text{RESET}}$ is input	• Set by instruction

LREL0	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 SCL0 and SDA0 lines go into the high impedance state. The following flags are cleared. • STD0 • ACKD0 • TRC0 • COI0 • EXC0 • MSTS0 • STT0 • SPT0

The standby mode following exit from communication 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 or extension code reception occurs after the start condition.

Condition for clearing (LREL0 = 0)**Note**	Condition for setting (LREL0 = 1)
• Automatically cleared after execution • When $\overline{\text{RESET}}$ is input	• Set by instruction

WREL0	Cancel wait
0	Does not cancel wait
1	Cancels wait. This setting is automatically cleared after wait is cancelled.

If WREL0 is set (wait release) during the wait period of the ninth clock in the transmission status, the SDA0 line becomes high impedance (TRC0 = 0).

Condition for clearing (WREL0 = 0)**Note**	Condition for setting (WREL0 = 1)
• Automatically cleared after execution • When $\overline{\text{RESET}}$ is input	• Set by instruction

SPIE0	Enable/disable generation of interrupt request when stop condition is detected
0	Disable
1	Enable

Condition for clearing (SPIE0 = 0)**Note**	Condition for setting (SPIE0 = 1)
• Cleared by instruction • When $\overline{\text{RESET}}$ is input	• Set by instruction

Note This flag's signal is invalid when IICE0 = 0.

Figure 14-3. Format of IIC0 Control Register (IICC0) (2/3)

WTIM0	Control of wait and interrupt request generation
0	Interrupt request is generated at the eighth clock's falling edge. Master mode:After output of eight clocks, clock output is set to low level and master device enters wait state. Slave mode: After input of eight clocks, the clock is set to low level and master device enters wait state.
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 master device enters wait state. Slave mode: After input of nine clocks, the clock is set to low level and master device enters wait state.
This bit's setting is invalid during an address transfer and is valid after the transfer is completed. When in master mode, a wait is inserted at the falling edge of the ninth clock during address transfers. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after an ACK signal is issued. When the slave device has received an extension code, it enters wait state at the falling edge of the eighth clock.	
Condition for clearing (WTIM0 = 0)**Note**	Condition for setting (WTIM0 = 1)
• Cleared by instruction • When $\overline{\text{RESET}}$ is input	• Set by instruction

ACKE0	Acknowledge control
0	Disable acknowledge.
1	Enable acknowledge. During the ninth clock period, the SDA0 line is set to low level. However, ACK is invalid during address transfers and is valid when EXC0 = 1.
Condition for clearing (ACKE0 = 0)**Note**	Condition for setting (ACKE0 = 1)
• Cleared by instruction • When $\overline{\text{RESET}}$ is input	• Set by instruction

STT0	Start condition trigger
0	Does not generate a start condition.
1	When bus is released (during STOP mode): Generates a start condition (for starting as master). The SDA0 line is changed from high level to low level and then the start condition is generated. Next, after the rated amount of time has elapsed, SCL0 is changed to low level. When bus is not used: This trigger functions as a start condition reserve flag. When set, it releases the bus and then automatically generates a start condition. Wait status (during master mode): Generates a restart condition after wait is released.
Cautions concerning set timing • For master reception: Cannot be set during transfer. Can be set only at the waiting period when ACKE0 has been set to 0 and slave has been notified of final reception. • For master transmission: A start condition may not be generated normally during the ACK period. Therefore, set it during the waiting period. • Cannot be set at the same time with SPT0	
Condition for clearing (STT0 = 0)**Note**	Condition for setting (STT0 = 1)
• Cleared by instruction • Cleared by loss in arbitration • Cleared after start condition is generated by master device • When LREL0 = 1 • When $\overline{\text{RESET}}$ is input	• Set by instruction

Note This flag's signal is invalid when IICE0 = 0.

Figure 14-3. Format of IIC0 Control Register (IICC0) (3/3)

SPT0	Stop condition trigger
0	Stop condition is not generated.
1	Stop condition is generated (termination of master device's transfer). After the SDA0 line goes to low level, either set the SCL0 line to high level or wait until it goes to high level. Next, after the rated amount of time has elapsed, the SDA0 line changes from low level to high level and a stop condition is generated.

Cautions concerning set timing

- For master reception: Cannot be set during transfer.
  Can be set only at the waiting period when ACKE0 has been set to 0 and slave has been notified of final reception.
- For master transmission: A stop condition cannot be generated normally during the ACK0 period. Therefore, set it during the waiting period.
- Cannot be set at the same time as STT0.
- SPT0 can be set only when in master mode.[Note 1]
- When WTIM0 has been set to 0, if SPT0 is set 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.
  When a ninth clock must be output, WTIM0 should be changed from 0 to 1 during the wait period following output of eight clocks, and SPT0 should be set during the wait period that follows output of the ninth clock.

Condition for clearing (SPT0 = 0)[Note 2]	Condition for setting (SPT0 = 1)
• Cleared by instruction • Cleared by loss in arbitration • Automatically cleared after stop condition is detected • When $\overline{\text{LREL0}}$ = 1 • When $\overline{\text{RESET}}$ is input	• Set by instruction

Notes 1. Set SPT0 only during master mode. However, SPT0 must be set and generate a stop condition before the first stop condition is detected following the switch to operation enable status. For details, see **14.5.15 Other cautions**.

2. This flag's signal is invalid when IICE0 = 0.

Caution When bit 3 (TRC0) of the IIC0 status register (IICS0) is set to 1, WREL0 is set during the ninth clock and wait is cancelled, after which TRC0 is cleared and the SDA0 line is set to high impedance.

Remarks 1. STD0: Bit 1 of IIC0 status register (IICS0)
ACKD0: Bit 2 of IIC0 status register (IICS0)
TRC0: Bit 3 of IIC0 status register (IICS0)
COI0: Bit 4 of IIC0 status register (IICS0)
EXC0: Bit 5 of IIC0 status register (IICS0)
MSTS0: Bit 7 of IIC0 status register (IICS0)

2. Bits 0 and 1 (SPT0, STT0) become 0 when they are read after data setting.

(2) IIC0 status register (IICS0)

This register indicates the status of the I^2C.

IICS0 can be set with a 1-bit or 8-bit memory manipulation instruction. IICS0n is a read-only register.

$\overline{\text{RESET}}$ input sets the value to 00H.

Figure 14-4. Format of IIC0 Status Register (IICS0) (1/3)

Address: FFA9H After reset: 00H R

Symbol	7	6	5	4	3	2	1	0
IICS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0

MSTS0	Master device status
0	Slave device status or communication standby status
1	Master device communication status

Condition for clearing (MSTS0 = 0)	Condition for setting (MSTS0 = 1)
• When a stop condition is detected • When ALD0 = 1 • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When a start condition is generated

ALD0	Detection of arbitration loss
0	This status means either that there was no arbitration or that the arbitration result was a “win”.
1	This status indicates the arbitration result was a “loss”. MSTS0 is cleared.

Condition for clearing (ALD0 = 0)	Condition for setting (ALD0 = 1)
• Automatically cleared after IICS0 is read**Note** • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When the arbitration result is a “loss”.

EXC0	Detection of extension code reception
0	Extension code was not received.
1	Extension code was received.

Condition for clearing (EXC0 = 0)	Condition for setting (EXC0 = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When the high-order 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 other bits of IICS0.

Figure 14-4. Format of IIC0 Status Register (IICS0) (2/3)

COI0	Detection of matching addresses
0	Addresses do not match.
1	Addresses match.

Condition for clearing (COI0 = 0)	Condition for setting (COI0 = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When the received address matches the local address (SVA0) (set at the rising edge of the eighth clock).

TRC0	Detection of transmit/receive status
0	Receive status (other than transmit status). The SDA0 line is set to high impedance.
1	Transmit status. The value in the SO0 latch is enabled for output to the SDA0 line (valid starting at the rising edge of the first byte's ninth clock).

Condition for clearing (TRC0 = 0)	Condition for setting (TRC0 = 1)
• When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • Cleared by WREL0 = 1 • When ALD0 changes from 0 to 1 • When $\overline{\text{RESET}}$ is input Master • When 1 is output to the first byte's LSB (transfer direction specification bit) Slave • When a start condition is detected • When 0 is input at the first byte's LSB (transfer direction specification bit) • When not used for communication	Master • When a start condition is generated Slave • When 1 is input at the first byte's LSB (transfer direction specification bit)

ACKD0	Detection of ACK
0	ACK was not detected.
1	ACK was detected.

Condition for clearing (ACKD0 = 0)	Condition for setting (ACKD0 = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• After the SDA0 line is set to low level at the rising edge of the SCL0's ninth clock

Figure 14-4. Format of IIC0 Status Register (IICS0) (3/3)

STD0	Detection of start condition
0	Start condition was not detected.
1	Start condition was detected. This indicates that the address transfer period is in effect

Condition for clearing (STD0 = 0)	Condition for setting (STD0 = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock following address transfer • Cleared by LREL0 = 1 • When $\overline{\text{IICE0}}$ changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When a start condition is detected

SPD0	Detection of stop condition
0	Stop condition was not detected.
1	Stop condition was detected. The master device's communication is terminated and the bus is released.

Condition for clearing (SPD0 = 0)	Condition for setting (SPD0 = 1)
• At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition • When $\overline{\text{IICE0}}$ changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When a stop condition is detected

Remark LREL0: Bit 6 of IIC0 control register (IICC0)
IICE0: Bit 7 of IIC0 control register (IICC0)

(3) IIC0 transfer clock select register (IICCL0)

This register is used to set the transfer clock for I^2C.

IICCL0 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{RESET}$ input sets IICCL0 to 00H.

Figure 14-5. Format of IIC0 Transfer Clock Select Register (IICCL0) (1/2)

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

Symbol	7	6	5	4	3	2	1	0
IICCL0	0	0	CLD0	DAD0	SMC0	DFC0	CL01	CL00

CLD0	Detection of SCL0 line level (valid only when IICE0 = 1)
0	SCL0 line was detected at low level.
1	SCL0 line was detected at high level.

Condition for clearing (CLD0 = 0)	Condition for setting (CLD0 = 1)
• When the SCL0 line is at low level • When $\overline{IICE0}$ = 0 • When $\overline{RESET}$ is input	• When the SCL0 line is at high level

DAD0	Detection of SDA0 line level (valid only when IICE0 = 1)
0	SDA0 line was detected at low level.
1	SDA0 line was detected at high level.

Condition for clearing (DAD0 = 0)	Condition for setting (DAD0 = 1)
• When the SDA0 line is at low level • When $\overline{IICE0}$ = 0 • When $\overline{RESET}$ is input	• When the SDA0 line is at high level

SMC0	Operation mode switching
0	Operation in standard mode
1	Operation in high-speed mode

Condition for clearing (SMC0 = 0)	Condition for setting (SMC0 = 1)
• Cleared by instruction • When $\overline{RESET}$ is input	• Set by instruction

Note Bits 4 and 5 are read-only bits.

Figure 14-5. Format of IIC0 Transfer Clock Select Register (IICCL0) (2/2)

DFC0	Control of digital filter operation[Note]
0	Digital filter OFF
1	Digital filter ON

CL01	CL00	Selection of transfer rate	
		Standard mode	High-speed mode
0	0	fx/44 (setting prohibited)	fx/24 (262.1 kHz)
0	1	fx/86 (73.2 kHz)	
1	0	fx/172 (36.6 kHz)	fx/48 (131.1 kHz)
1	1	fx/66 (95.3 kHz)	fx/18 (349.5 kHz)

Note The digital filter can be used when in high-speed mode. Response time is slower when the digital filter is used.

Caution Stop serial transfer once before rewriting CL01, CL00 to other than the same value.

Remarks 1. IICE0: Bit 7 of IIC0 control register (IICC0)

2. fx: System clock oscillation frequency

3. Figures in parentheses apply to operation with fx = 6.29 MHz.

(4) IIC0 shift register (IIC0)

This register is used for serial transmission/reception (shift operations) that are synchronized with the serial clock. It can be read from or written to in 8-bit units, but data should not be written to IIC0 during a data transfer.

Address: FF1FH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
IIC0								

(5) Slave address register 0 (SVA0)

This register holds the I^2C's slave addresses.

It can be read from or written to in 8-bit units, but bit 0 is fixed to 0.

Address: FFABH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SVA0								0

14.4 I²C Bus Mode Functions

14.4.1 Pin configuration

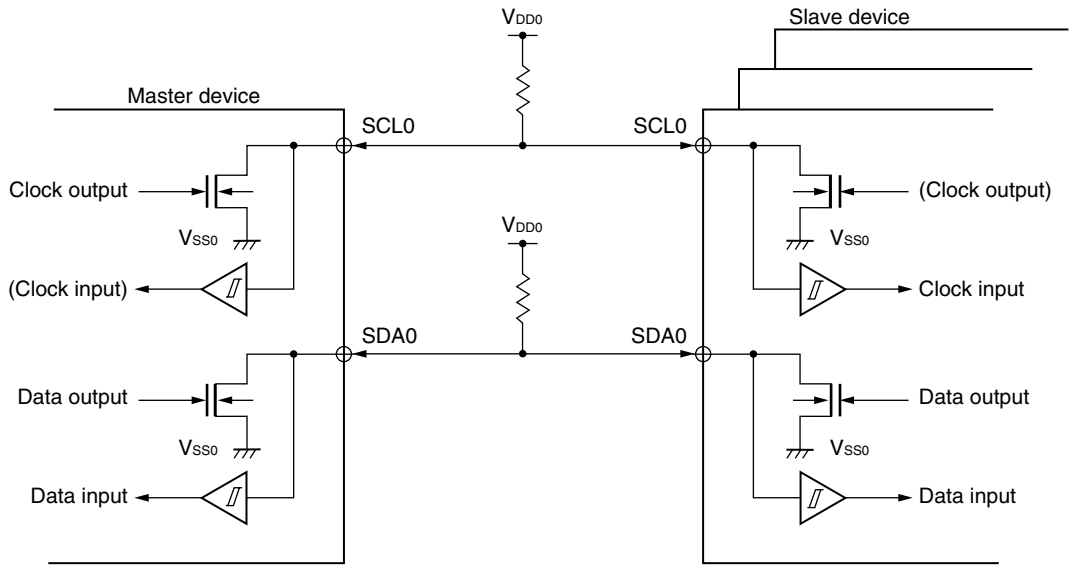
The serial clock pin (SCL0) and serial data bus pin (SDA0) are configured as follows.

(1) SCL0 ····· This pin is used for serial clock input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

(2) SDA0 ····· This pin is used for serial data input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Since outputs from the serial clock line and the serial data bus line are N-ch open-drain outputs, an external pull-up resistor is required.

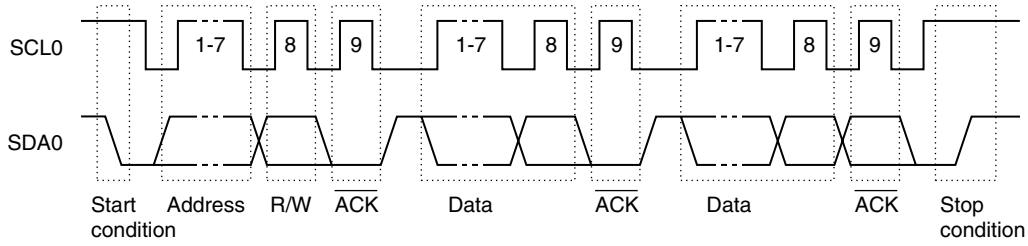
Figure 14-6. Pin Configuration Diagram



14.5 I²C Bus Definitions and Control Methods

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus. Figure 14-7 shows the transfer timing for the "start condition", "data", and "stop condition" output via the I²C bus's serial data bus.

Figure 14-7. Timing of I²C Bus Serial Data Transfer



The master device outputs the start condition, slave address, and stop condition.

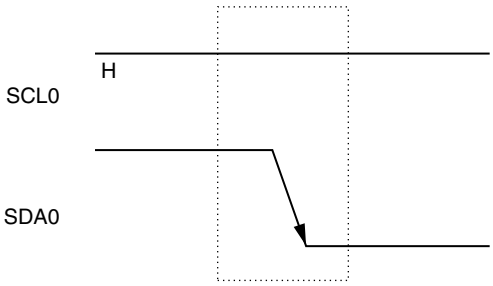
The acknowledge signal ($\overline{ACK}$) can be output by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCL0) is continuously output by the master device. However, in the slave device, the SCL0's low-level period can be extended and a wait can be inserted.

14.5.1 Start conditions

A start condition is met when the SCL0 pin is at high level and the SDA0 pin changes from high level to low level. The start conditions for the SCL0 pin and SDA0 pin are signals that the master device outputs to the slave device when starting a serial transfer. The slave device includes hardware for detecting start conditions.

Figure 14-8. Start Conditions



A start condition is output when the IIC0 control register (IICC0)'s bit 1 (STT0) is set to 1 after a stop condition has been detected (SPD0: Bit 0 = 1 in the IIC0 status register (IICS0)). When a start condition is detected, IICS0's bit 1 (STD0) is set to 1.

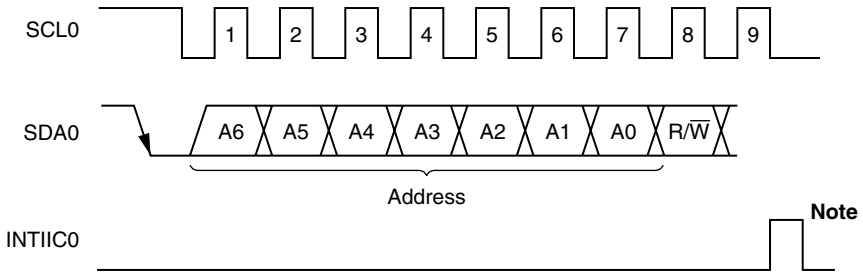
14.5.2 Addresses

The 7 bits of data that follow the start condition are defined as the 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 bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave devices include hardware that detects the start condition and checks whether or not the 7-bit address data matches the data values stored in slave address register 0 (SVA0). If the address data matches the SVA0 values, the slave device is selected and communicates with the master device until the master device transmits a start condition or stop condition.

Figure 14-9. Address



Note INTIIC0 is not issued if data other than a local address or extension code is received during slave device operation.

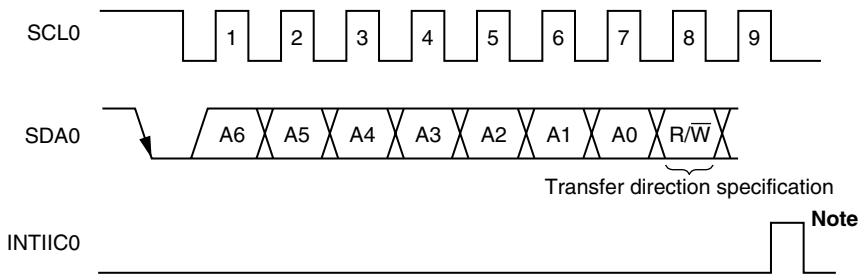
The slave address and the eighth bit, which specifies the transfer direction as described in **14.5.3 Transfer direction specification** below, are together written to the IIC0 shift register (IIC0) and are then output. Received addresses are written to IIC0.

The slave address is assigned to the high-order 7 bits of IIC0.

14.5.3 Transfer direction specification

In addition to the 7-bit address data, the master device sends 1 bit that specifies the transfer direction. When this transfer direction specification bit has a value of 0, it indicates that the master device is transmitting data to a slave device. When the transfer direction specification bit has a value of 1, it indicates that the master device is receiving data from a slave device.

Figure 14-10. Transfer Direction Specification



Note INTIIC0 is not issued if data other than a local address or extension code is received during slave device operation.

14.5.4 Acknowledge ($\overline{\text{ACK}}$) signal

The acknowledge ($\overline{\text{ACK}}$) signal is used by the transmitting and receiving devices to confirm serial data reception.

The receiving device returns one $\overline{\text{ACK}}$ signal for each 8 bits of data it receives. The transmitting device normally receives an $\overline{\text{ACK}}$ signal after transmitting 8 bits of data. However, when the master device is the receiving device, it does not output an $\overline{\text{ACK}}$ signal after receiving the final data to be transmitted. The transmitting device detects whether or not an $\overline{\text{ACK}}$ signal is returned after it transmits 8 bits of data. When an $\overline{\text{ACK}}$ signal is returned, the reception is judged as normal and processing continues. If the slave device does not return an $\overline{\text{ACK}}$ signal, the master device outputs either a stop condition or a restart condition and then stops the current transmission. Failure to return an $\overline{\text{ACK}}$ signal may be caused by the following two factors.

(a) Reception was not performed normally.
(b) The final data was received.

When the receiving device sets the SDA0 line to low level during the ninth clock, the $\overline{\text{ACK}}$ signal becomes active (normal receive response).

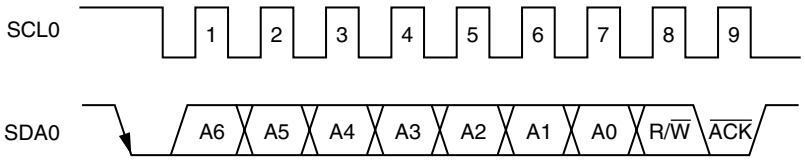
When bit 2 (ACKE0) of IIC0 control register (IICC0) is set to 1, automatic $\overline{\text{ACK}}$ signal generation is enabled.

Transmission of the eighth bit following the 7 address data bits causes bit 3 (TRC0) of the IIC0 status register (IICS0) to be set. When this TRC0 bit's value is 0, it indicates receive mode. Therefore, ACKE0 should be set to 1.

When the slave device is receiving (when TRC0 = 0), if the slave device does not need to receive any more data after receiving several bytes, setting ACKE0 to 0 will prevent the master device from starting transmission of the subsequent data.

Similarly, when the master device is receiving (when TRC0 = 0) and the subsequent data is not needed and when either a restart condition or a stop condition should therefore be output, setting ACKE0 to 0 will prevent the $\overline{\text{ACK}}$ signal from being returned. This prevents the MSB data from being output via the SDA0 line (i.e., stops transmission) during transmission from the slave device.

Figure 14-11. $\overline{\text{ACK}}$ Signal



When the local address is received, an $\overline{\text{ACK}}$ signal is automatically output in synchronization with the falling edge of the SCL's eighth clock regardless of the ACKE0 value. No $\overline{\text{ACK}}$ signal is output if the received address is not a local address.

The $\overline{\text{ACK}}$ signal output method during data reception is based on the wait timing setting, as described below.

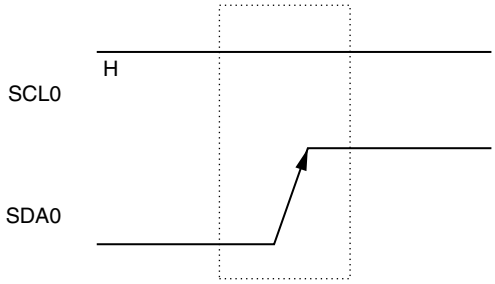
- When 8-clock wait is selected: $\overline{\text{ACK}}$ signal is output when ACKE0 is set to 1 before wait cancellation.
- When 9-clock wait is selected: $\overline{\text{ACK}}$ signal is automatically output at the falling edge of the SCL0's eighth clock if ACKE0 has already been set to 1.

14.5.5 Stop condition

When the SCL0 pin is at high level, changing the SDA0 pin from low level to high level generates a stop condition.

A stop condition is a signal that the master device outputs to the slave device when serial transfer has been completed. The slave device includes hardware that detects stop conditions.

Figure 14-12. Stop Condition



A stop condition is generated when bit 0 (SPT0) of the IIC0 control register (IICC0) is set to 1. When the stop condition is detected, bit 0 (SPD0) of the IIC0 status register (IICS0) is set to 1 and INTIIC0 is generated when bit 4 (SPIE0) of IICC0 is set to 1.

14.5.6 Wait signal ($\overline{\text{WAIT}}$)

The wait signal ($\overline{\text{WAIT}}$) is used to notify the communication partner that a device (master or slave) is preparing to transmit or receive data (i.e., is in a wait state).

Setting the SCL0 pin to low level notifies the communication partner of the wait status. When wait status has been cancelled for both the master and slave devices, the next data transfer can begin.

Figure 14-13. Wait Signal (1/2)

(1) When master device has a nine-clock wait and slave device has an eight-clock wait (master transmits, slave receives, and ACKE0 = 1)

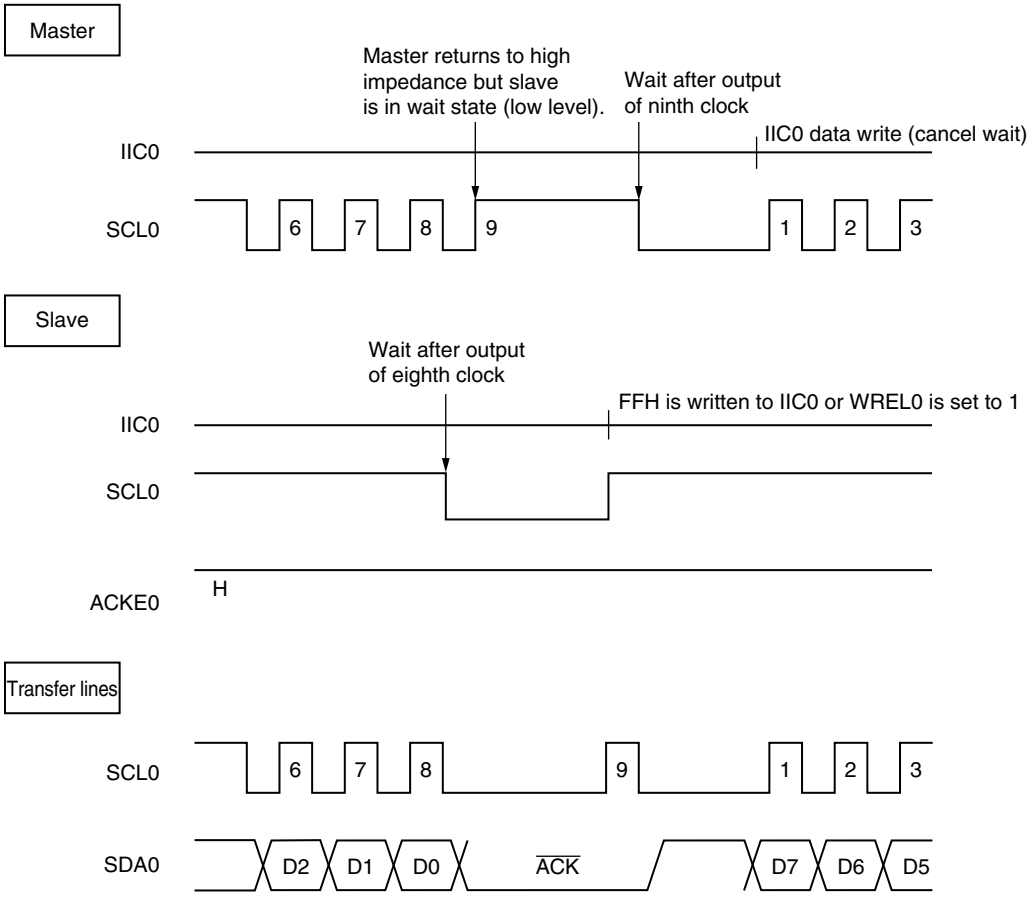
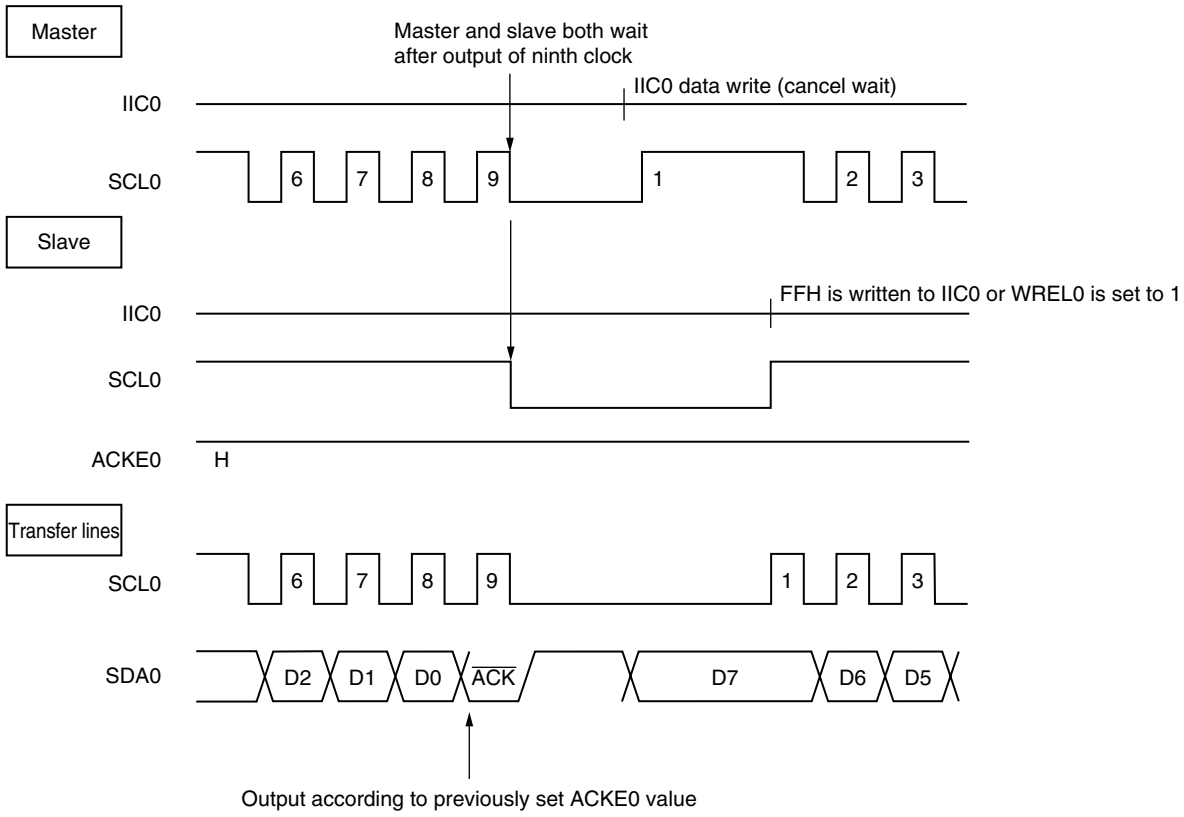


Figure 14-13. Wait Signal (2/2)

(2) When master and slave devices both have a nine-clock wait (master device transmits, slave receives, and ACKE0 = 1)



Remark ACKE0: Bit 2 of IIC0 control register (IICC0)
WREL0: Bit 5 of IIC0 control register (IICC0)

A wait is automatically generated depending on the setting for bit 3 (WTIM0) of the IIC0 control register (IICC0).

Normally, when bit 5 (WREL0) of IICC0 is set to 1 or when FFH is written to the IIC0 shift register (IIC0), the wait status is canceled and the transmitting side writes data to IIC0 to cancel the wait status.

The master device can also cancel the wait status via either of the following methods.

- By setting bit 1 (STT0) of IICC0 to 1
- By setting bit 0 (SPT0) of IICC0 to 1

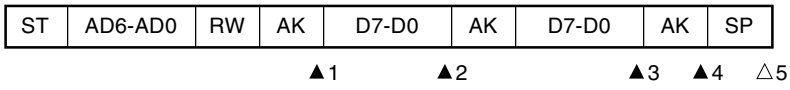
14.5.7 I^2C interrupt requests (INTIIC0)

The INTIIC0 interrupt request timing and the IIC0 status register (IICS0) settings corresponding to that timing are described below.

(1) Master device operation

(a) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

(i) When WTIM0 = 0

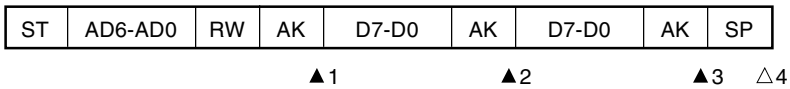


▲1 ▲2 ▲3 ▲4 △5

▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B
▲3: IICS0 = 1000×000B (Sets WTIM0)
▲4: IICS0 = 1000××00B (Sets SPT0)
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1



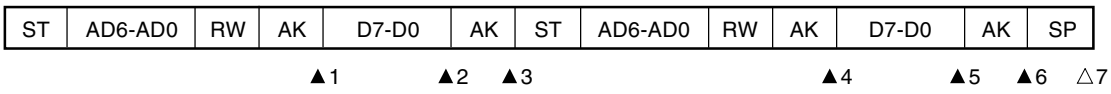
▲1 ▲2 ▲3 △4

▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×100B
▲3: IICS0 = 1000××00B (Sets SPT0)
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

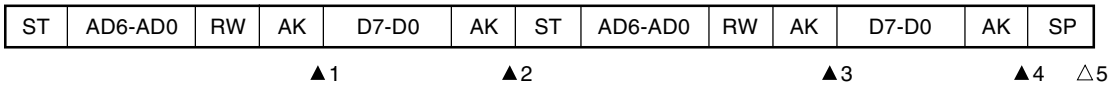
(i) When WTIM0 = 0



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B (Sets WTIM0)
▲3: IICS0 = 1000××00B (Clears WTIM0, sets STT0)
▲4: IICS0 = 1000×110B
▲5: IICS0 = 1000×000B (Sets WTIM0)
▲6: IICS0 = 1000××00B (Sets SPT0)
△7: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1

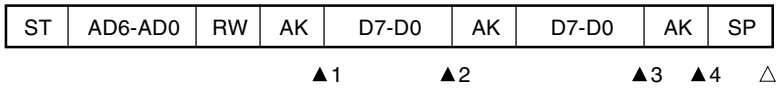


▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000××00B (Sets STT0)
▲3: IICS0 = 1000×110B
▲4: IICS0 = 1000××00B (Sets SPT0)
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(c) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

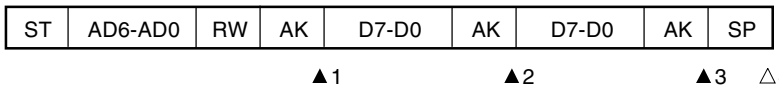
(i) When WTIM0 = 0



▲1: IICS0 = 1010×110B
▲2: IICS0 = 1010×000B
▲3: IICS0 = 1010×000B (Sets WTIM0)
▲4: IICS0 = 1010××00B (Sets SPT0)
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1



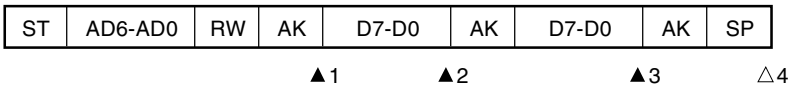
▲1: IICS0 = 1010×110B
▲2: IICS0 = 1010×100B
▲3: IICS0 = 1010××00B (Sets SPT0)
△4: IICS0 = 00001001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(2) Slave device operation (Slave address data reception time (matches with SVA0))

(a) Start ~ Address ~ Data ~ Data ~ Stop

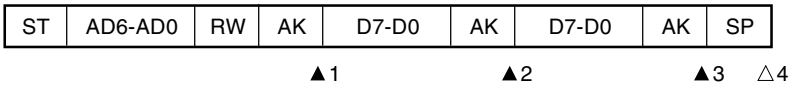
(i) When WTIM0 = 0



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×000B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1

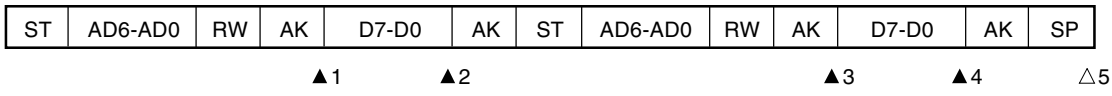


▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×100B
▲3: IICS0 = 0001××00B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

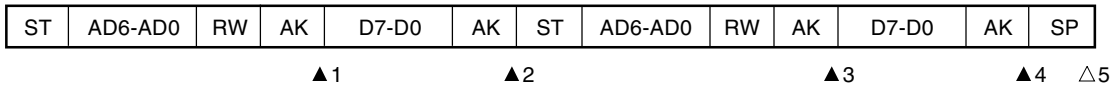
(i) When WTIM0 = 0 (after restart, matches with SVA0)



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001×000B
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1 (after restart, matches with SVA0)

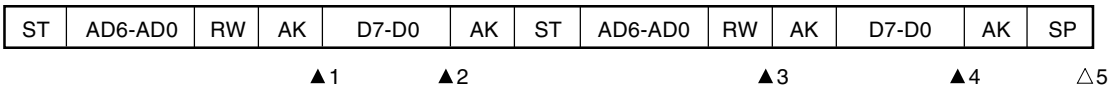


▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001××00B
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(c) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

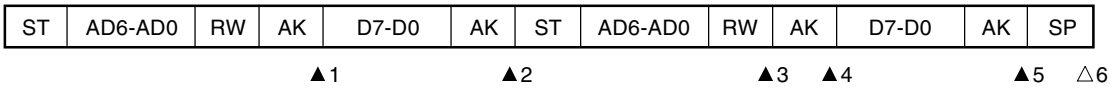
(i) When WTIM0 = 0 (after restart, extension code reception)



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×000B
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1 (after restart, extension code reception)

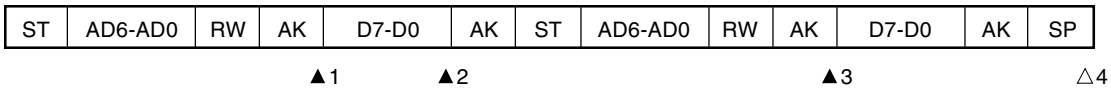


▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×110B
▲5: IICS0 = 0010××00B
△6: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(d) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

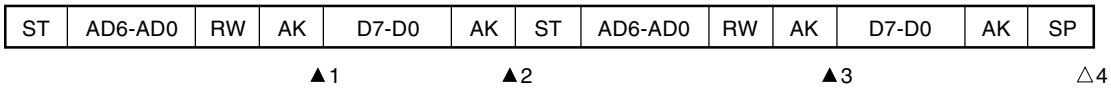
(i) When WTIM0 = 0 (after restart, does not match with address (= not extension code))



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 00000×10B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1 (after restart, does not match with address (= not extension code))



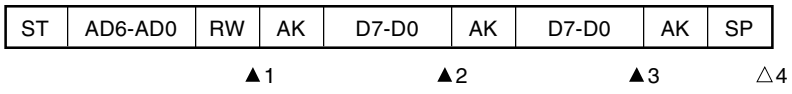
▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICS0 = 00000×10B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(3) Slave operation (when receiving extension code)

(a) Start ~ Code ~ Data ~ Data ~ Stop

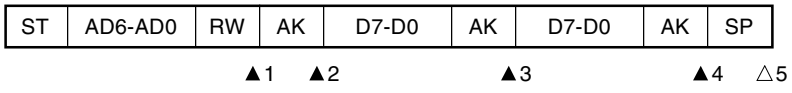
(i) When WTIM0 = 0



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×000B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1

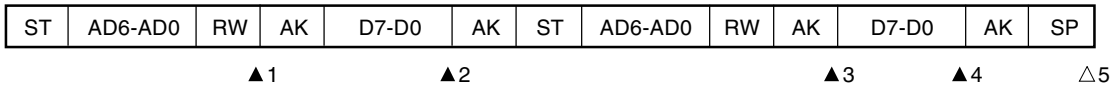


▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010×100B
▲4: IICS0 = 0010××00B
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(b) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

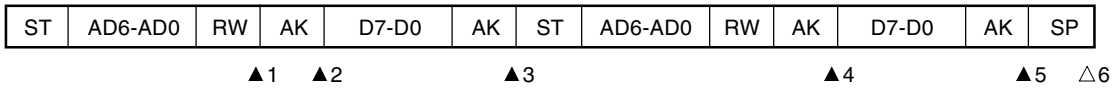
(i) When WTIM0 = 0 (after restart, matches with SVA0)



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001×000B
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1 (after restart, matches with SVA0)

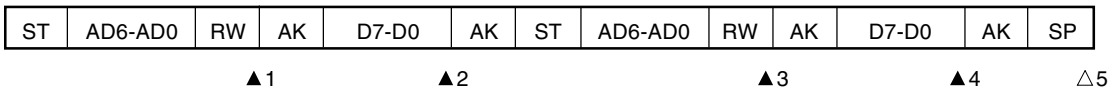


▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010××00B
▲4: IICS0 = 0001×110B
▲5: IICS0 = 0001××00B
△6: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(c) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

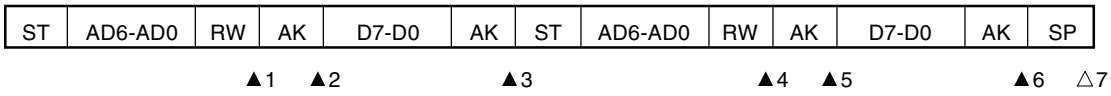
(i) When WTIM0 = 0 (after restart, extension code reception)



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×000B
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1 (after restart, extension code reception)

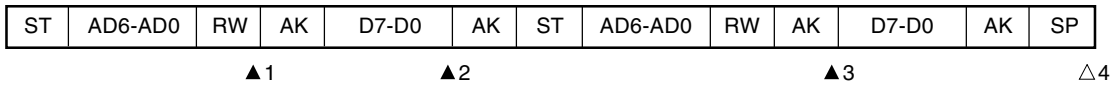


▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010××00B
▲4: IICS0 = 0010×010B
▲5: IICS0 = 0010×110B
▲6: IICS0 = 0010××00B
△7: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(d) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

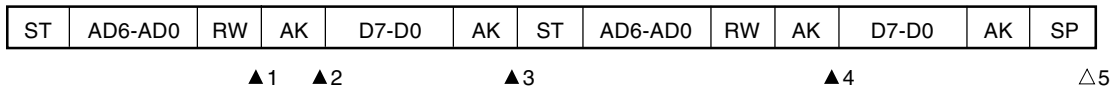
(i) When WTIM0 = 0 (after restart, does not match with address (= not extension code))



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 00000×10B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1 (after restart, does not match with address (= not extension code))

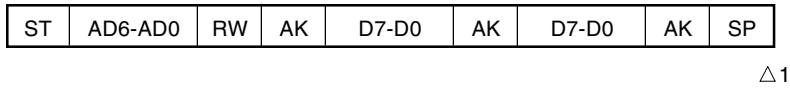


▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010××00B
▲4: IICS0 = 00000×10B
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(4) Operation without communication

(a) Start ~ Code ~ Data ~ Data ~ Stop



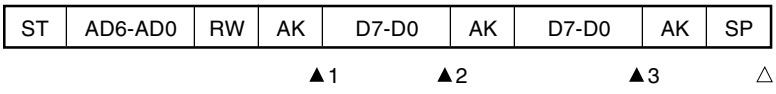
△1: IICS0 = 00000001B

Remark △: Generated only when SPIE0 = 1

(5) Arbitration loss operation (operation as slave after arbitration loss)

(a) When arbitration loss occurs during transmission of slave address data

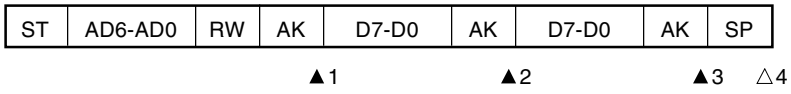
(i) When WTIM0 = 0



▲1: IICS0 = 0101×110B (Example: when ALD0 is read during interrupt servicing)
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×000B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1

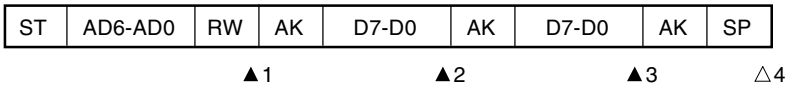


▲1: IICS0 = 0101×110B (Example: when ALD0 is read during interrupt servicing)
▲2: IICS0 = 0001×100B
▲3: IICS0 = 0001××00B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(b) When arbitration loss occurs during transmission of extension code

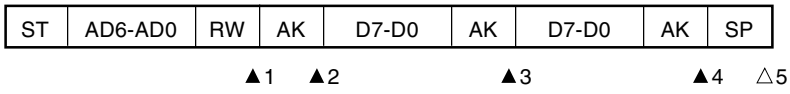
(i) When WTIM0 = 0



▲1: IICS0 = 0110×010B (Example: when ALD0 is read during interrupt servicing)
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×000B
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1

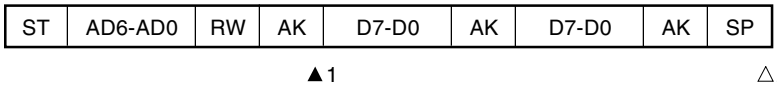


▲1: IICS0 = 0110×010B (Example: when ALD0 is read during interrupt servicing)
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010×100B
▲4: IICS0 = 0010××00B
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(6) Operation when arbitration loss occurs (no communication after arbitration loss)

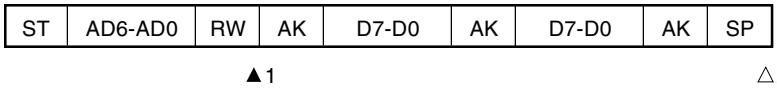
(a) When arbitration loss occurs during transmission of slave address data (when WTIM0 = 1)



▲1: IICS0 = 01000110B (Example: when ALD0 is read during interrupt servicing)
△2: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1

(b) When arbitration loss occurs during transmission of extension code

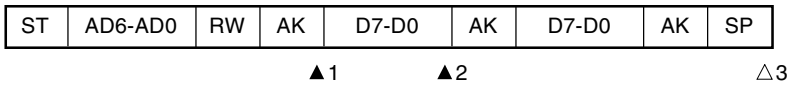


▲1: IICS0 = 0110×010B (Example: when ALD0 is read during interrupt servicing)
LREL0 is set to 1 by software
△2: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(c) When arbitration loss occurs during data transfer

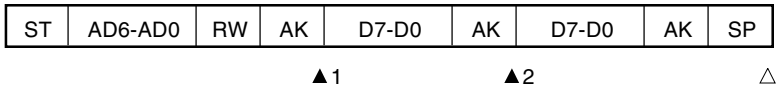
(i) When WTIM0 = 0



▲1: IICS0 = 10001110B
▲2: IICS0 = 01000000B (Example: when ALD0 is read during interrupt servicing)
△3: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1

(ii) When WTIM0 = 1

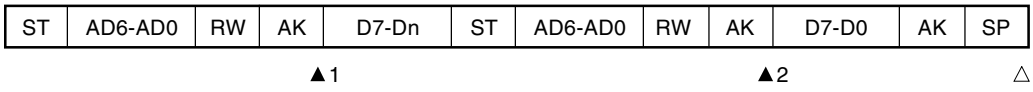


▲1: IICS0 = 10001110B
▲2: IICS0 = 01000100B (Example: when ALD0 is read during interrupt servicing)
△3: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1

(d) When loss occurs due to restart condition during data transfer

(i) Not extension code (Example: does not match with SVA0 or WTIM0 = 1)



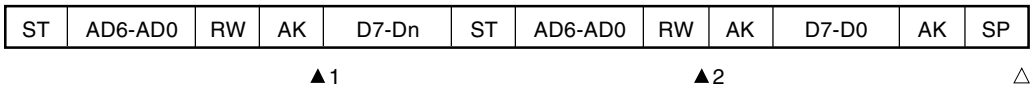
▲1: IICS0 = 1000×110B

▲2: IICS0 = 01000110B (Example: when ALD0 is read during interrupt servicing)

△3: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care
n = 6 to 0

(ii) Extension code



▲1: IICS0 = 1000×110B

▲2: IICS0 = 0110×010B (Example: when ALD0 is read during interrupt servicing)

Set LREL0 = 1 by software

△3: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care
n = 6 to 0

(e) When loss occurs due to stop condition during data transfer

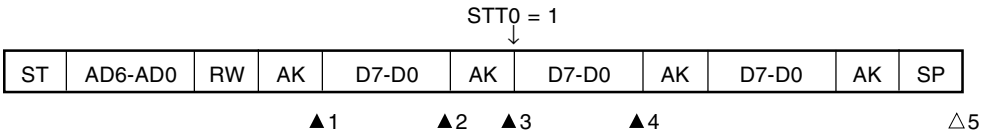
▲1: IICS0 = 1000×110B

△2: IICS0 = 01000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care
n = 6 to 0

(f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition

(i) When WTIM0 = 0



▲1: IICS0 = 1000×110B

▲2: IICS0 = 1000×000B (Sets WTIM0)

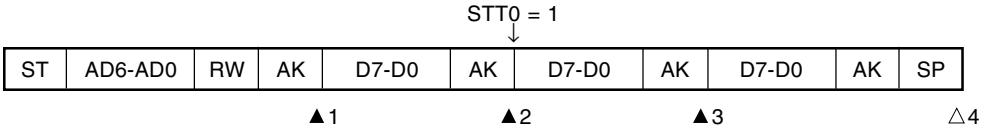
▲3: IICS0 = 1000××00B (Clears WTIM0 and sets STT0 = 1)

▲4: IICS0 = 01000000B (Example: when ALD0 is read during interrupt servicing)

△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1



▲1: IICS0 = 1000×110B

▲2: IICS0 = 1000×100B (Sets STT0)

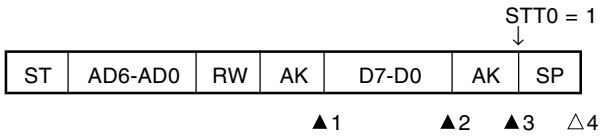
▲3: IICS0 = 01000100B (Example: when ALD0 is read during interrupt servicing)

△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(g) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

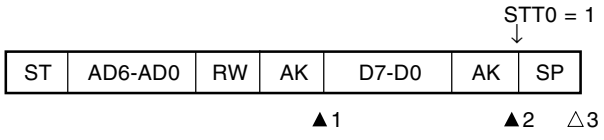
(i) When WTIM0 = 0



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B (Sets WTIM0)
▲3: IICS0 = 1000××00B (Sets STT0)
△4: IICS0 = 01000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1

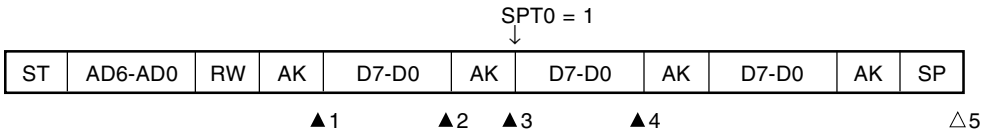


▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000××00B (Sets STT0)
△3: IICS0 = 01000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(h) When arbitration loss occurs due to low-level data when attempting to generate a stop condition

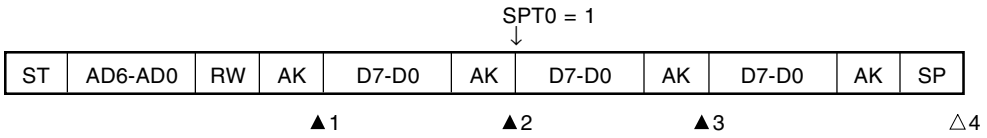
(i) When WTIM0 = 0



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B (Sets WTIM0)
▲3: IICS0 = 1000××00B (Clears WTIM0 and sets SPT0)
▲4: IICS0 = 01000000B (Example: when ALD0 is read during interrupt servicing)
△5: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000××00B (Sets SPT0)
▲3: IICS0 = 01000000B (Example: when ALD0 is read during interrupt servicing)
△4: IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

14.5.8 Interrupt request (INTIIC0) generation timing and wait control

The setting of bit 3 (WTIM0) in the IIC0 control register (IICC0) determines the timing by which INTIIC0 is generated and the corresponding wait control, as shown in Table 14-2.

Table 14-2. INTIIC0 Generation Timing and Wait Control

WTIM	During Slave Device Operation			During Master Device Operation		
	Address	Data reception	Data transmission	Address	Data reception	Data transmission
0	9**Notes 1, 2**	8**Note 2**	8**Note 2**	9	8	8
1	9**Notes 1, 2**	9**Note 2**	9**Note 2**	9	9	9

Notes 1. The slave device's INTIIC0 signal and wait period occur at the falling edge of the ninth clock only when there is a match with the address set to slave address register 0 (SVA0).
At this point, $\overline{ACK}$ is output regardless of the value set to IICC0's bit 2 (ACKE0). For a slave device that has received an extension code, INTIIC0 occurs at the falling edge of the eighth clock.

2. If the received address does not match the contents of slave address register 0 (SVA0), neither INTIIC0 nor a wait occurs.

Remark The numbers in the table indicate the number of the serial clock's clock signals. Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

(1) During address transmission/reception

- Slave device operation: Interrupt and wait timing are determined regardless of the WTIM0 bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIM0 bit.

(2) During data reception

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

(3) During data transmission

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

(4) Wait cancellation method

The four wait cancellation methods are as follows.

- By setting bit 5 (WREL0) of IIC0 control register (IICC0) to 1
- By writing to the IIC0 shift register (IIC0)
- By setting a start condition (setting bit 1 (STT0) of IIC0 control register (IICC0) to 1)
- By setting a stop condition (setting bit 0 (SPT0) of IIC0 control register (IICC0) to 1)

When 8-clock wait has been selected (WTIM0 = 0), the output level of $\overline{ACK}$ must be determined prior to wait cancellation.

(5) Stop condition detection

INTIIC0 is generated when a stop condition is detected.

14.5.9 Address match detection method

When in I^2C bus mode, the master device can select a particular slave device by transmitting the corresponding slave address.

An address match can be detected automatically by hardware. An interrupt request (INTIIC0) occurs when a local address has been set to slave address register 0 (SVA0) and when the address set to SVA0 matches the slave address sent by the master device, or when an extension code has been received.

14.5.10 Error detection

In I^2C bus mode, the status of the serial data bus (SDA0) during data transmission is captured by the IIC0 shift register (IIC0) of the transmitting device, so the IIC0 data prior to transmission can be compared with the transmitted IIC0 data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match.

14.5.11 Extension code

(1) When the high-order 4 bits of the receive address are either "0000" or "1111", the extension code flag (EXC0) is set for extension code reception and an interrupt request (INTIIC0) is issued at the falling edge of the eighth clock. The local address stored in slave address register 0 (SVA0) is not affected.

(2) If "111110××" is set to SVA0 by a 10-bit address transfer and "111110××0" is transferred from the master device, the results are as follows. Note that INTIIC0 occurs at the falling edge of the eighth clock.

- High-order four bits of data match: EXC0 = 1**Note**
- Seven bits of data match: COI0 = 1**Note**

Note EXC0: Bit 5 of IIC0 status register (IICS0)
COI0: Bit 4 of IIC0 status register (IICS0)

(3) Since the processing after the interrupt request occurs differs according to the data that follows the extension code, such processing is performed by software.
For example, after the extension code is received, if the target device is not to be operated as a slave device, bit 6 (LREL0) of the IIC0 control register (IICC0) can be set to 1 to set the standby mode for the next communication operation.

Table 14-3. Extension Code Bit Definitions

Slave Address	R/W Bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	×	CBUS address
0000 010	×	Address that is reserved for different bus format
1111 0××	×	10-bit slave address specification

14.5.12 Arbitration

When several master devices simultaneously output a start condition (when STT0 is set to 1 before STD0 is set to 1[Note]), communication among the master devices is performed as the number of clocks are adjusted until the data differs. This kind of operation is called arbitration.

When one of the master devices loses in arbitration, an arbitration loss flag (ALD0) in the IIC0 status register (IICS0) is set via the timing by which the arbitration loss occurred, and the SCL0 and SDA0 lines are both set to high impedance, which releases the bus.

The arbitration loss is detected based on the timing of the next interrupt request (the eighth or ninth clock, when a stop condition is detected, etc.) and the ALD0 = 1 setting that has been made by software.

For details of interrupt request timing, see **14.5.7 I^2C interrupt requests (INTIIC0)**.

Note STD0: Bit 1 of IIC0 status register (IICS0)
STT0: Bit 1 of IIC0 control register (IICC0)

Figure 14-14. Arbitration Timing Example

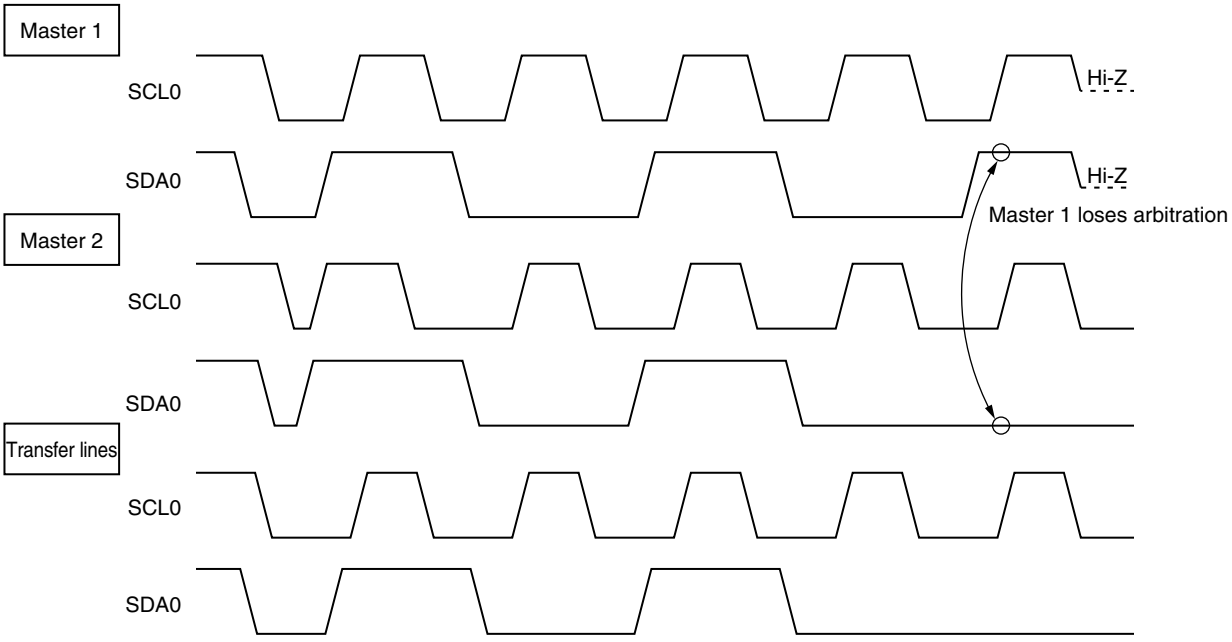


Table 14-4. Status during Arbitration and Interrupt Request Generation Timing

Status during Arbitration	Interrupt Request Generation Timing
During address transmission	At falling edge of eighth or ninth clock following byte transfer[Note 1]
Read/write data after address transmission	
During extension code transmission	
Read/write data after extension code transmission	
During data transmission	
During $\overline{\text{ACK}}$ signal transfer period after data transmission	
When restart condition is detected during data transfer	
When stop condition is detected during data transfer	When stop condition is output (when SPIE0 = 1)[Note 2]
When data is at low level while attempting to output a restart condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When stop condition is detected while attempting to output a restart condition	When stop condition is output (when SPIE0 = 1)[Note 2]
When data is at low level while attempting to output a stop condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When SCL0 is at low level while attempting to output a restart condition	

Notes 1. When WTIM0 (bit 3 of the IIC0 control register (IICC0)) = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIM0 = 0 and the extension code's slave address is received, an interrupt request occurs at the falling edge of the eighth clock.

2. When there is a chance that arbitration will occur, set SPIE0 = 1 for master device operation.

Remark SPIE0: Bit 5 of the IIC0 control register (IICC0)

14.5.13 Wake-up function

This is a I^2C bus slave's function that generates an interrupt request (INTIIC0) when a local address and extension code have been received.

This function makes processing more efficient by preventing unnecessary interrupt requests from occurring when addresses do not match.

When a start condition is detected, wake-up standby mode is set. This wake-up standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which has output a start condition) to a slave device.

However, when a stop condition is detected, bit 5 (SPIE0) of the IIC0 control register (IICC0) is set regardless of the wake up function, and this determines whether interrupt requests are enabled or disabled.

14.5.14 Communication reservation

To start master device communication when not currently using a bus, a communication reservation can be made to enable transmission of a start condition when the bus is released. There are two modes under which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ is not returned and the bus was released when bit 6 (LREL0) of the IIC0 control register (IICC0) was set to 1).

If bit 1 (STT0) of IICC0 is set while the bus is not used (after a stop condition is detected), a start condition is automatically generated and the wait status is set.

When the bus release is detected (when a stop condition is detected), writing to the IIC0 shift register (IIC0) causes the master's address transfer to start. At this point, IICC0's bit 4 (SPIE0) should be set.

When STT0 has been set, the operation mode (as start condition or as communication reservation) is determined according to the bus status.

- If the bus has been released ..... a start condition is generated
- If the bus has not been released (standby mode) ..... communication reservation

Check whether the communication reservation operates or not with MSTS0 (bit 7 of the IIC0 status register (IICS0)) after SST0 is set and a wait time elapses.

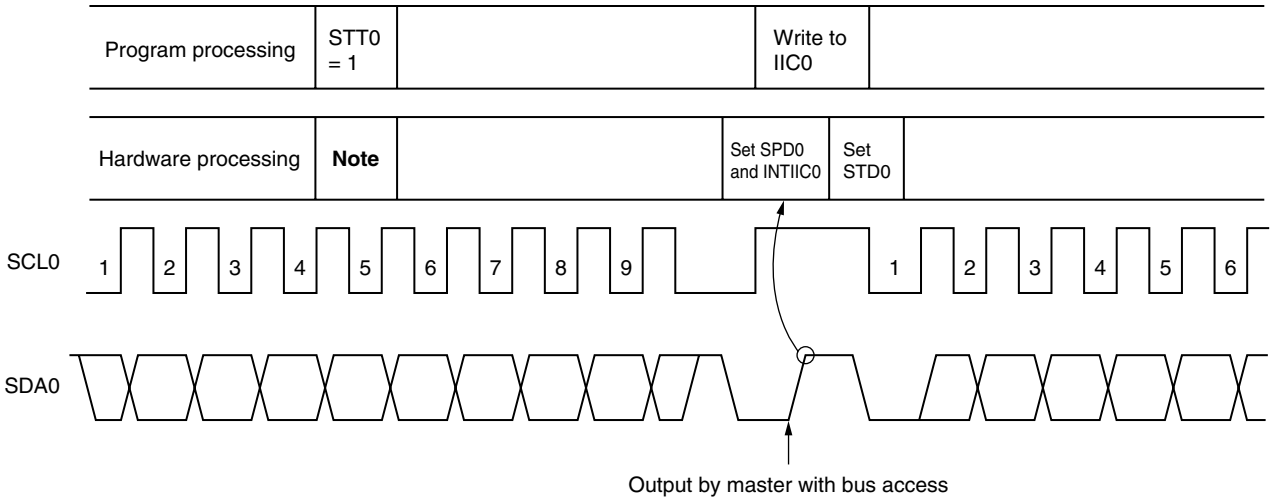
Wait periods, which should be set via software, are listed in Table 14-5. These wait periods can be set via the settings for bits 3, 1, and 0 (SMC0, CL01, and CL00) in the IIC0 clock select register (IICCL0).

Table 14-5. Wait Periods

SMC0	CL01	CL00	Wait Period
0	0	0	26 clocks × 1/fx
0	0	1	46 clocks × 1/fx
0	1	0	
0	1	1	37 clocks × 1/fx
1	0	0	16 clocks × 1/fx
1	0	1	
1	1	0	32 clocks × 1/fx
1	1	1	13 clocks × 1/fx

Figure 14-15 shows communication reservation timing.

Figure 14-15. Timing of Communication Reservation



Note Communication reservation

Remark IIC0: IIC0 shift register
STT0: Bit 1 of IIC0 control register (IICC0)
STD0: Bit 1 of IIC0 status register (IICS0)
SPD0: Bit 0 of IIC0 status register (IICS0)

Communication reservations are accepted via the following timing. After bit 1 (STD0) of the IIC0 status register (IICS0) is set to 1, a communication reservation can be made by setting bit 1 (STT0) of the IIC0 control register (IICC0) to 1 before a stop condition is detected.

Figure 14-16. Timing for Accepting Communication Reservations

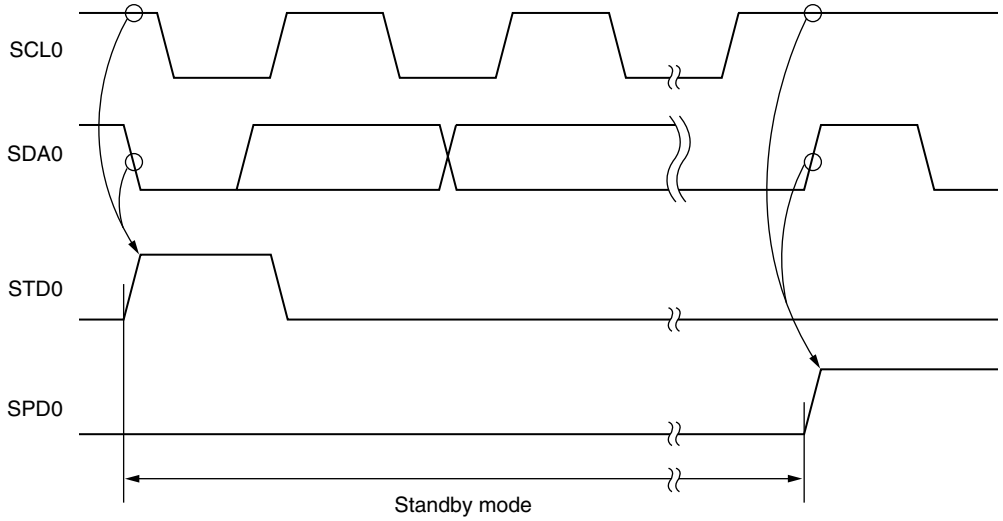
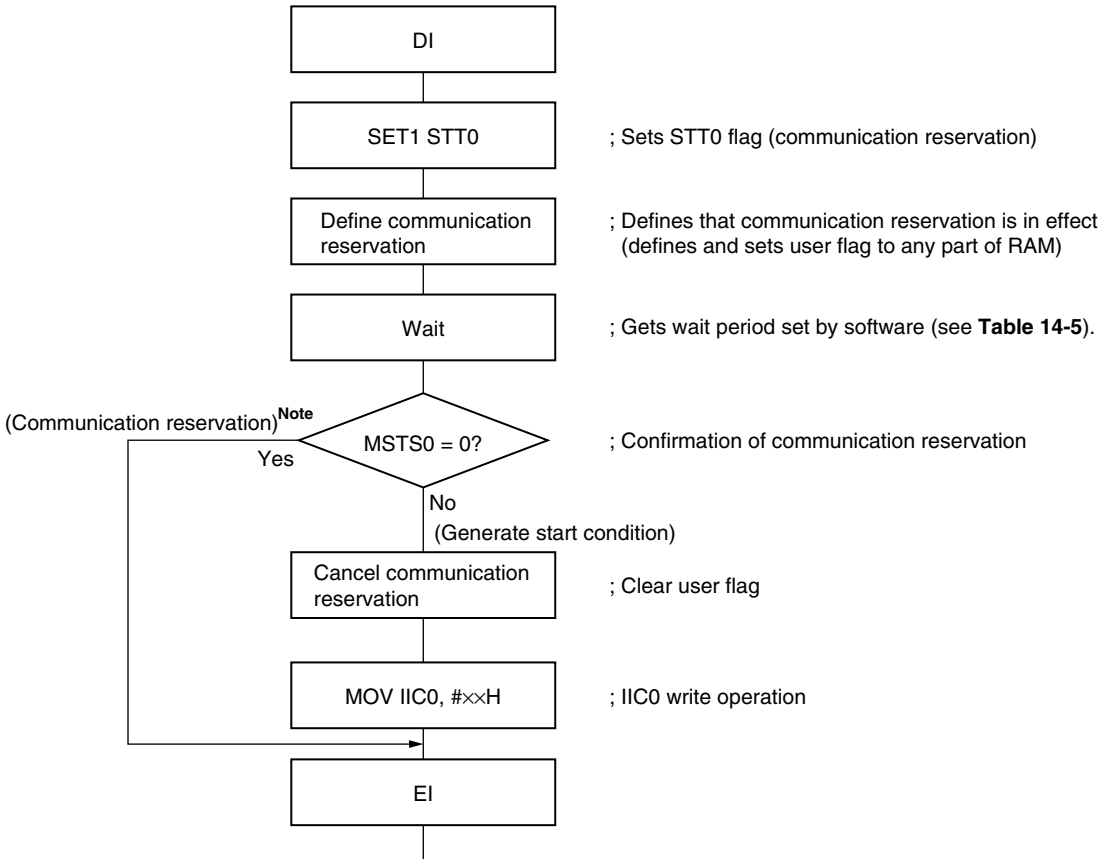


Figure 14-17 shows the communication reservation protocol.

Figure 14-17. Communication Reservation Protocol



Note The communication reservation operation executes a write to the IIC0 shift register (IIC0) when a stop condition interrupt request occurs.

Remark STT0: Bit 1 of IIC0 control register (IICC0)
MSTS0: Bit 7 of IIC0 status register (IICS0)
IIC0: IIC0 shift register

14.5.15 Other cautions

After a reset, when changing from a mode in which no stop condition has been detected (the bus has not been released) to a master device communication mode, first generate a stop condition to release the bus, then perform master device communication.

When using multiple masters, it is not possible to perform master device communication when the bus has not been released (when a stop condition has not been detected).

Use the following sequence for generating a stop condition.

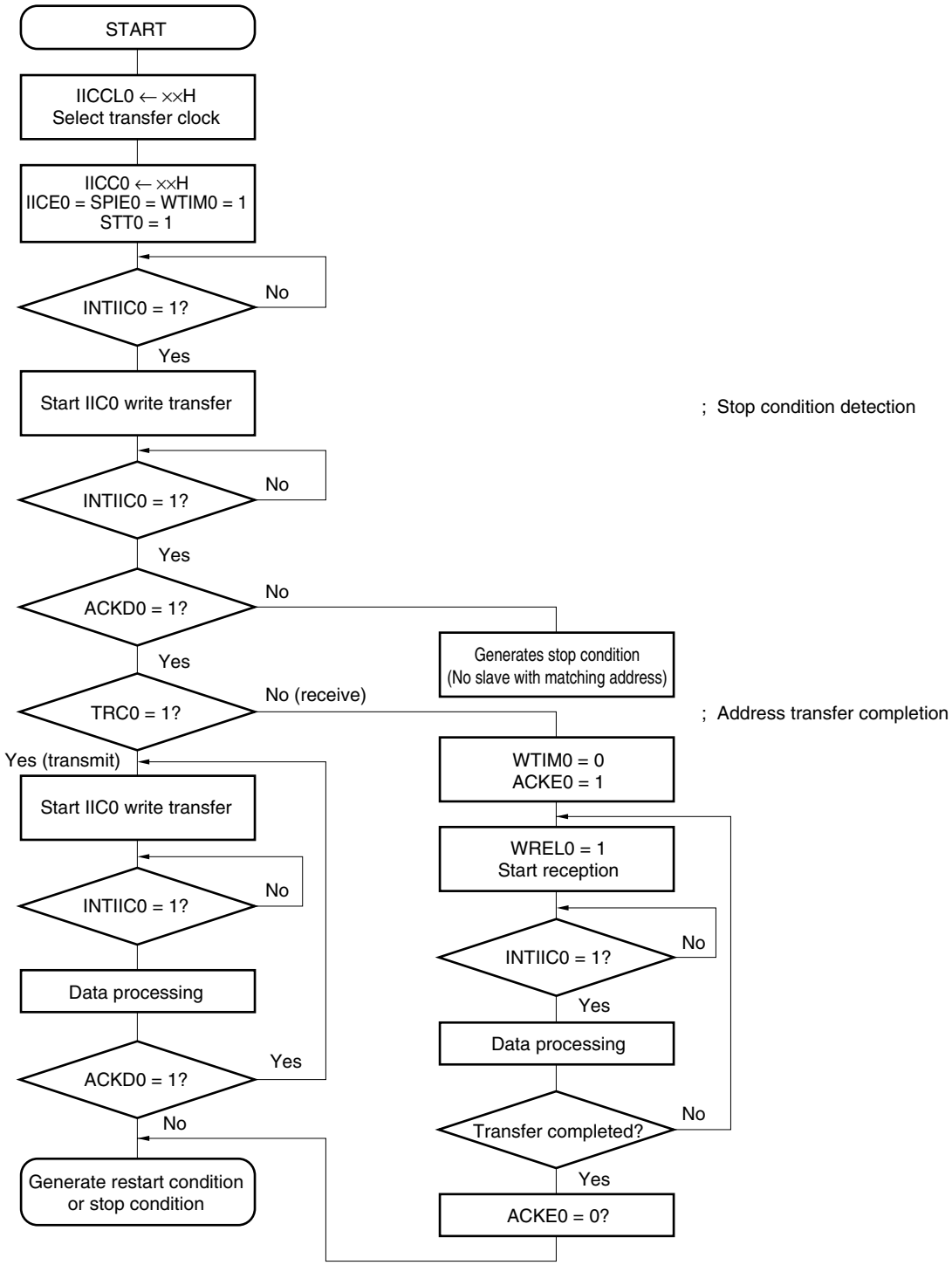
(a) Set the IIC0 transfer clock select register (IICCL0).
(b) Set bit 7 (IICE0) of the IIC0 control register (IICC0).
(c) Set bit 0 of IICC0.

14.5.16 Communication operations

(1) Master operations

The following is a flow chart of the master operations.

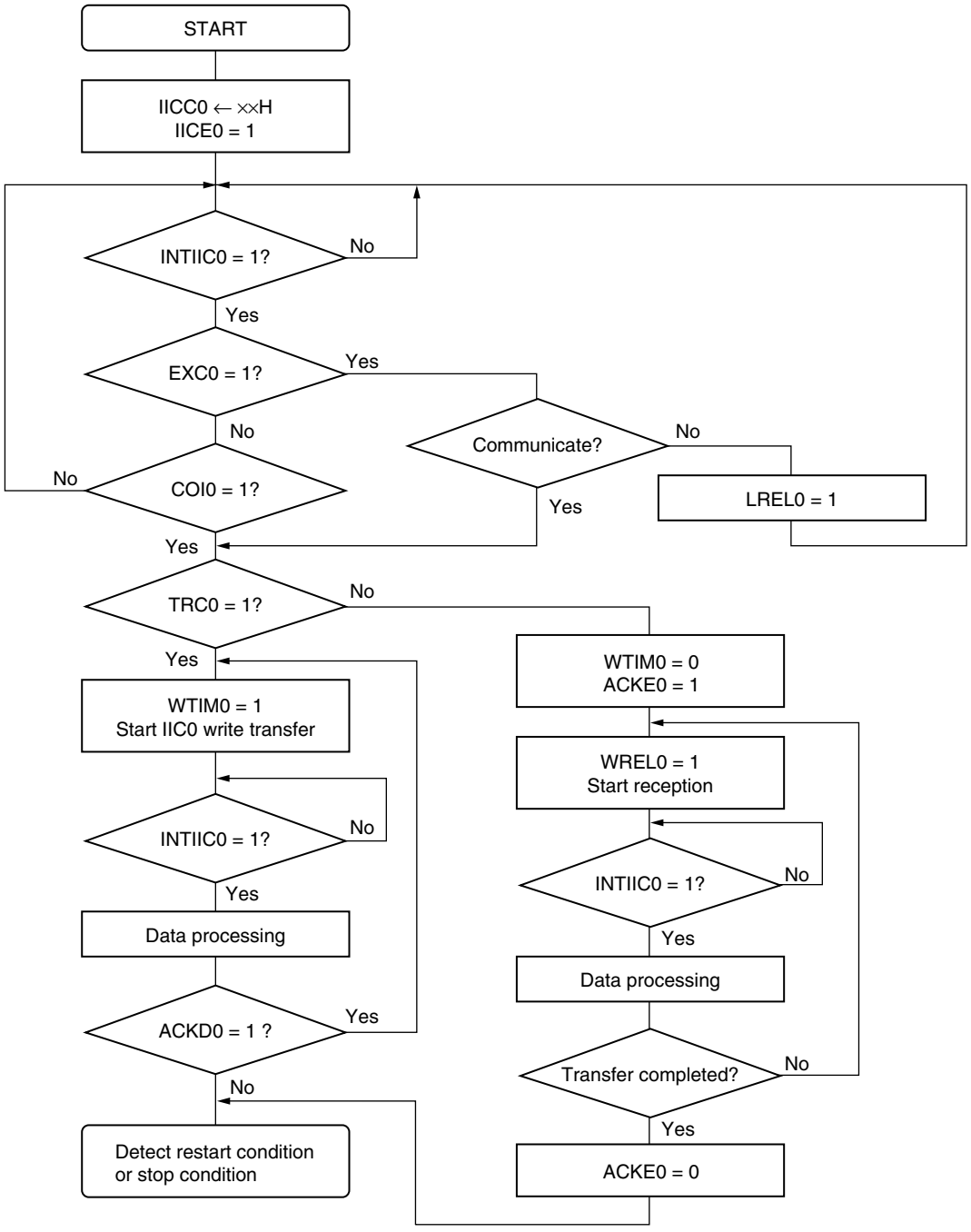
Figure 14-18. Master Operation Flow Chart



(2) Slave operation

An example of slave operation is shown below.

Figure 14-19. Slave Operation Flow Chart



14.6 Timing Charts

When using the I^2C bus mode, the master device outputs an address via the serial bus to select one of several slave devices as its communication partner.

After outputting the slave address, the master device transmits the TRC0 bit (bit 3 of the IIC0 status register (IICS0)) that specifies the data transfer direction and then starts serial communication with the slave device.

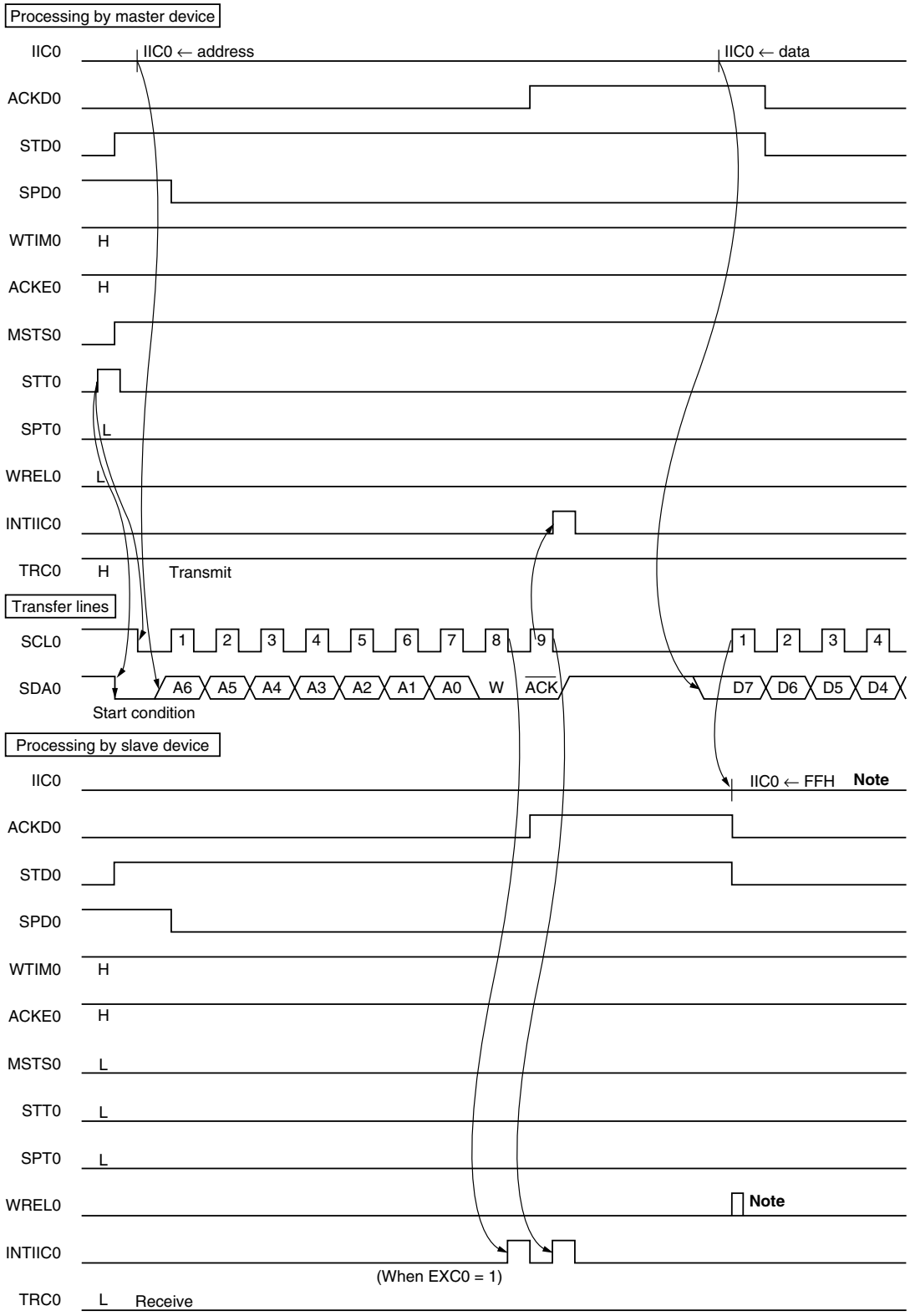
Figures 14-20 and 14-21 show timing charts of the data communication.

The IIC0 shift register (IIC0)'s shift operation is synchronized with the falling edge of the serial clock (SCL0). The transmit data is transferred to the SO0 latch and is output (MSB first) via the SDA0 pin.

Data input via the SDA0 pin is captured into IIC0 at the rising edge of SCL0.

Figure 14-20. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)

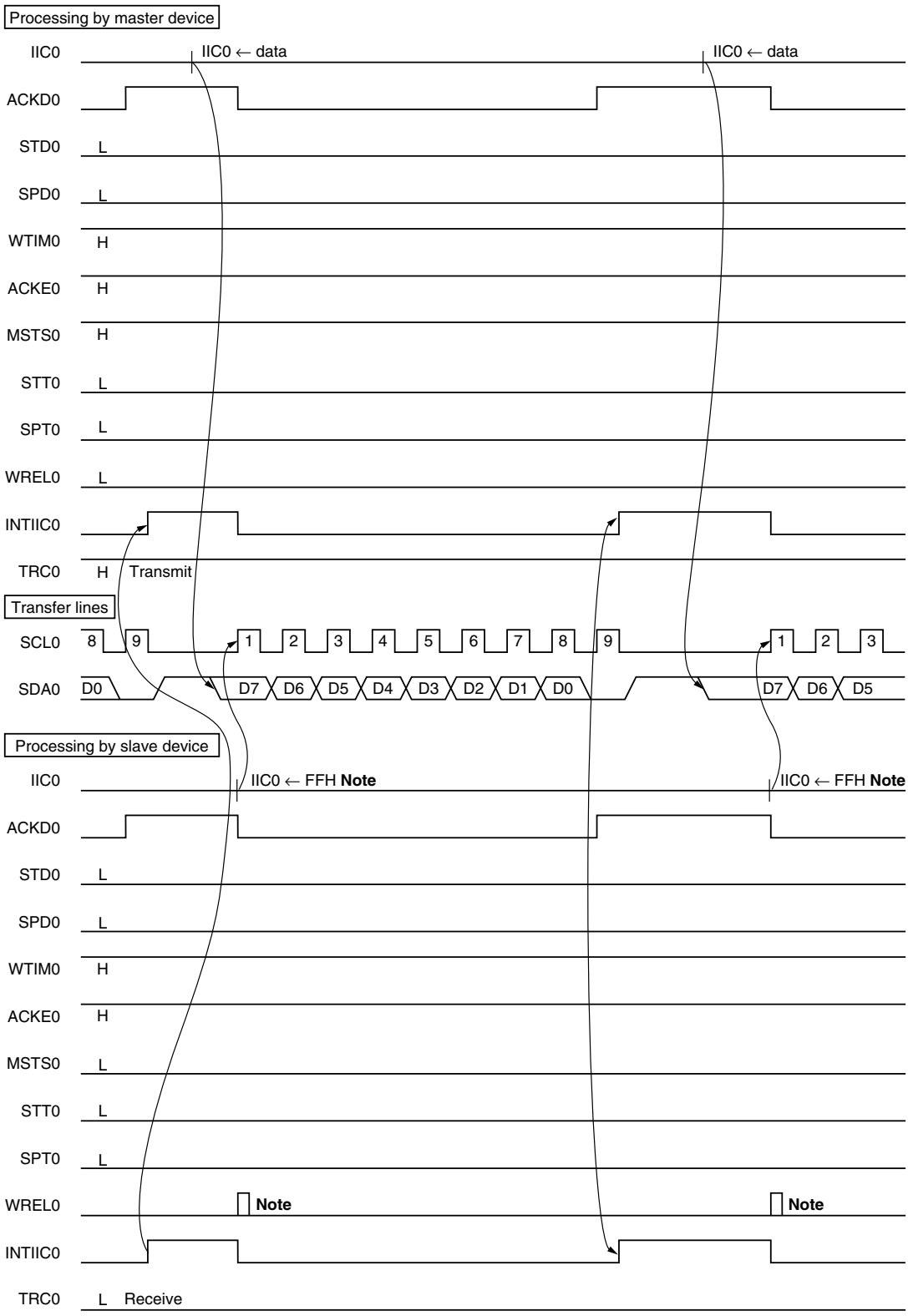
(1) Start condition ~ address



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

Figure 14-20. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)

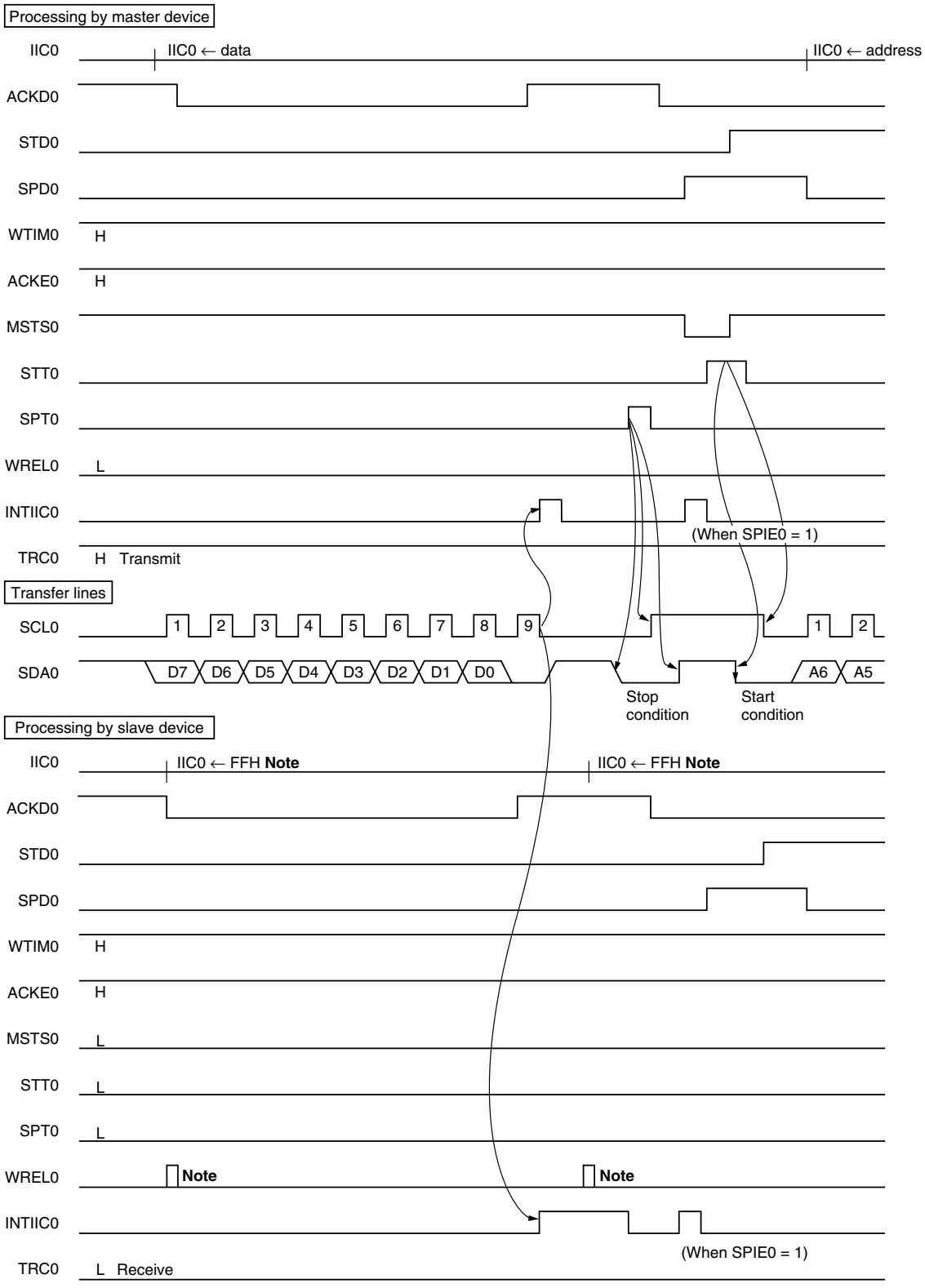
(2) Data



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

Figure 14-20. Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)

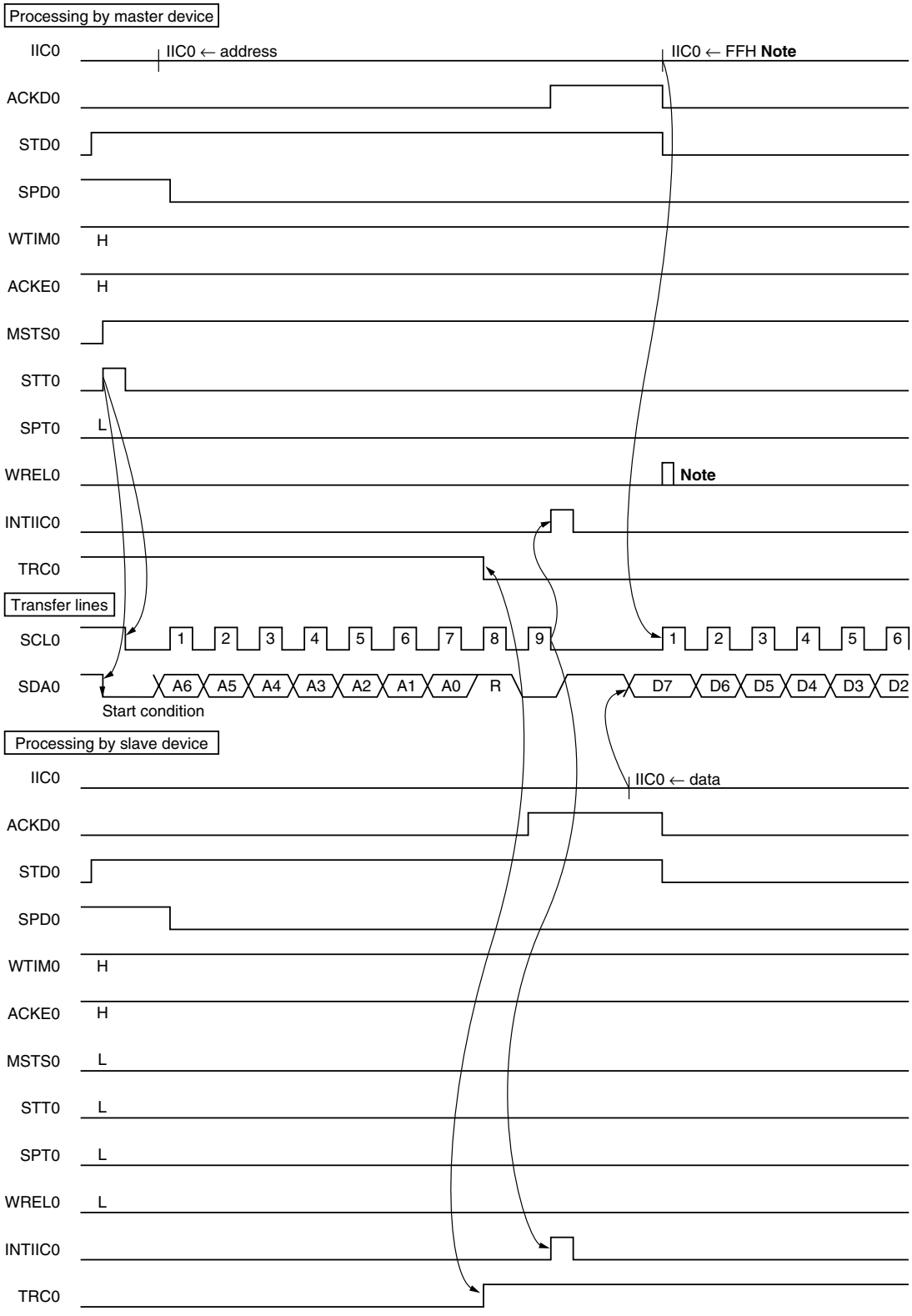
(3) Stop condition



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

Figure 14-21. Example of Slave to Master Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)

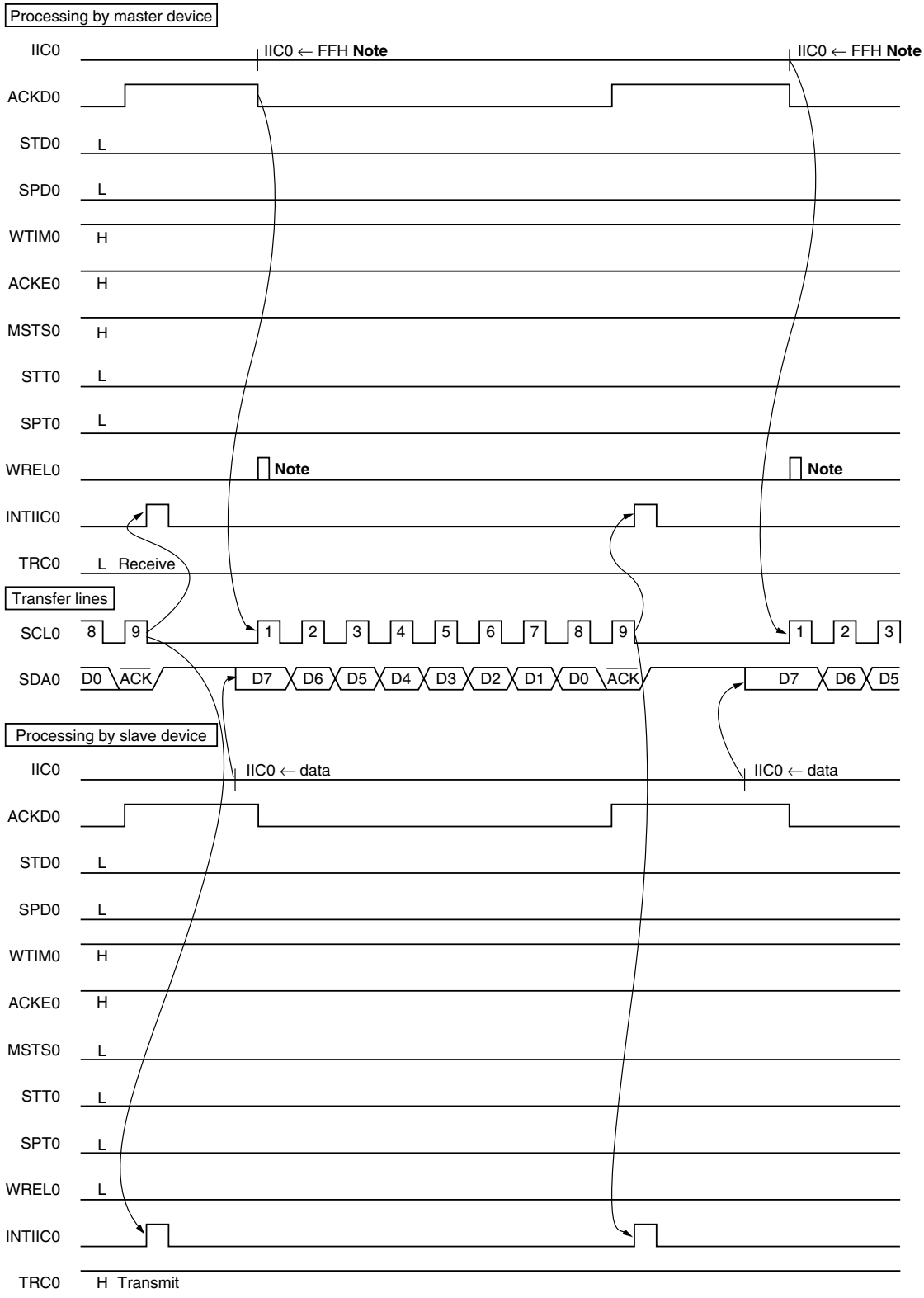
(1) Start condition ~ address



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

Figure 14-21. Example of Slave to Master Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)

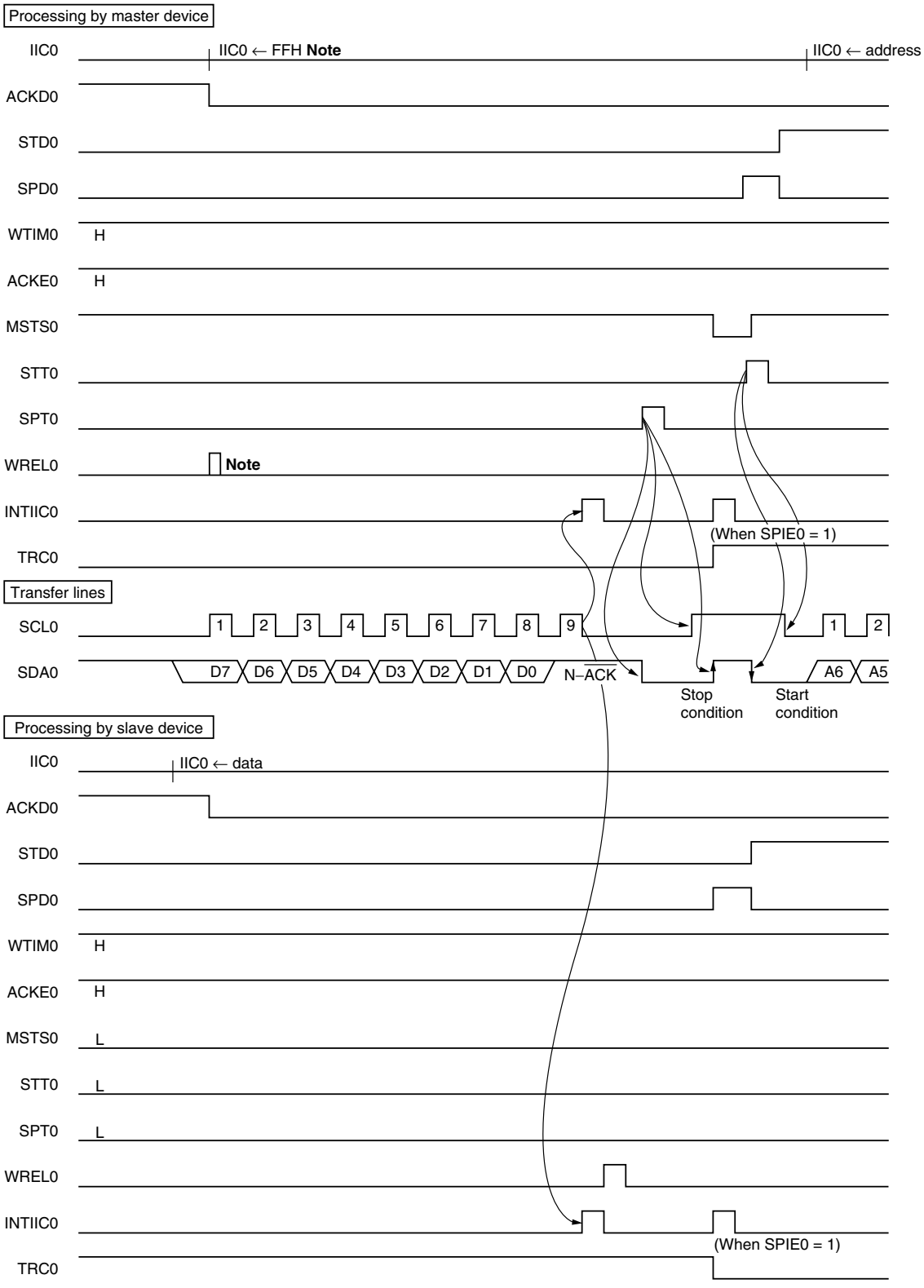
(2) Data



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

Figure 14-21. Example of Slave to Master Communication (When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)

(3) Stop condition



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

CHAPTER 15 DCAN CONTROLLER (μPD780701Y, 78F0701Y ONLY)

The μPD780701Y and 78F0701Y both incorporate a DCAN (Direct storage Control Area Network) controller, whereas the μPD780702Y does not. Moreover, because the DCAN controller pins in the μPD78F0701Y can also be used for an IEBus controller, it is not possible to use the DCAN controller at the same time the IEBus controller is being used.

Table 15-1. Overview of Functions

Function	Details
Protocol	CAN2.0 with active extended frame capability (Bosch specification 2.0 part B)
Baud rate	Max. 390 kbps @ 6.29-MHz operation
Bus line control	CMOS I/O for external transceiver
Clock	Selected by register
Data storage	Buffer RAM for DCAN : 288 bytes[Note]
Message configuration	Messages received via a message identifier are stored in the RAM area. Two transmit message buffers
Message number	Up to 16 receive messages including 2 mask Two transmit channels
Message sorting	Separate identifier can be set for all 16 receive messages 2 messages with mask identifier Global mask for all messages
Interrupts	One transmit interrupt request One receive interrupt request One error interrupt request
Time functions	A time stamp function is available
Other functions	A separate transmit/receive error counter is available A flag for checking the bus connection is available A dedicated receive mode is available (for use when detecting the baud rate on the bus)
Low power consumption modes	Sleep mode: Can be woken up from DCAN bus Stop mode: Cannot be woken up from DCAN bus

Note The buffer RAM for the DCAN has been allocated to the area F900H to FA1FH. When the DCAN is not in use, this buffer RAM can be used as normal expansion RAM.

15.1 Protocol

DCAN is an abbreviation of “Direct storage Control Area Network”, and is a class C high-speed multiplexed communication protocol for automotive real-time communication. CAN conforms to the ISO (International Organization for Standardization) and SAE (Society of Automotive Engineers) standards. For more detailed information, refer to **Bosch CAN Specification 2.0 September 1991**.

15.1.1 Protocol mode function

(1) Standard format mode

In the standard format the message identifier has 11 bits, and can differentiate between 2032 types of messages.

(2) Extension format mode

In the extension format mode, the message identifier has 29 bits (11 + 18) and can differentiate between 2032 $\times 2^{18}$ types of messages.

When the IDE bit of the arbitration field is “recessive”, the extension format mode is entered.

When the message of the extension format mode and the remote frame of the standard format mode are simultaneously transmitted, the node transmitting the message with the standard mode wins arbitration.

(3) Bus values

The bus can have one of two complementary logical values: “dominant” or “recessive”. During simultaneous transmission of “dominant” and “recessive” bits, the resulting bus value will be “dominant”.

For example, in case of a wired-AND implementation of the bus, the “dominant” level would be represented by a logical “0” and the “recessive” level by a logical 1.

Physical states (e.g. electrical voltage, light) that represent the logical levels are not given in this specification.

15.1.2 Message format

The DCAN protocol message supports different types of frames. The output conditions of each frame are as follows:

- Data frame: Carries the data from a transmitter to the receiver.
- Remote frame: Transmission demand frame from the requesting node.
- Error frame: Frame output on error detection.
- Overload frame: The frame output from the first bit of the intermission when the receiving node has not completed reception.

15.1.3 Data frame/remote frame

Figure 15-1. Data Frame

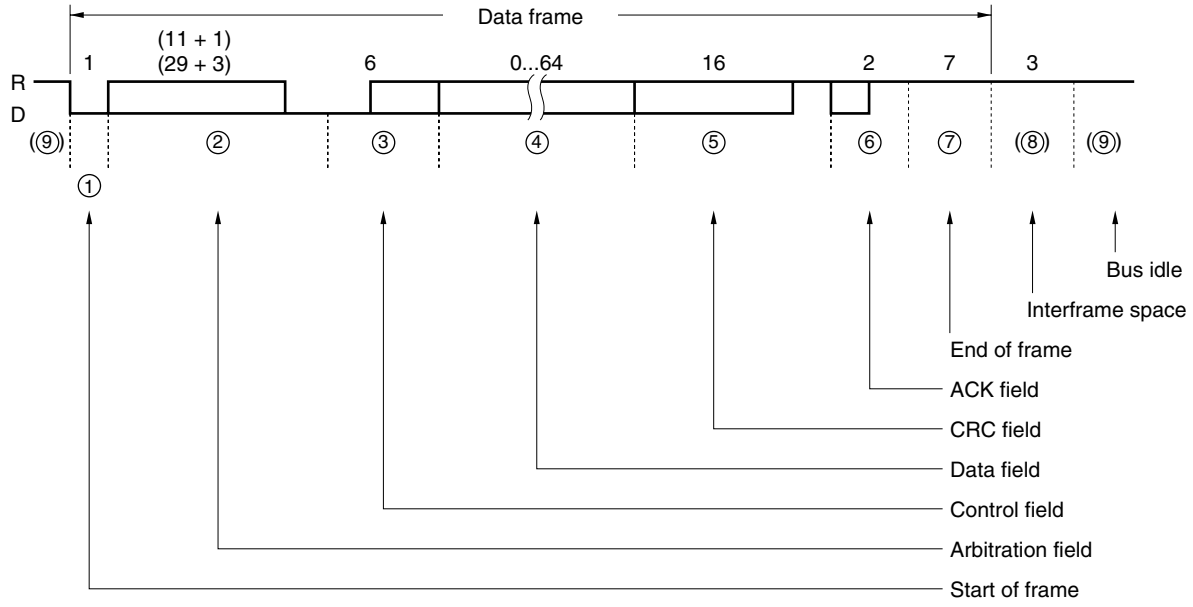
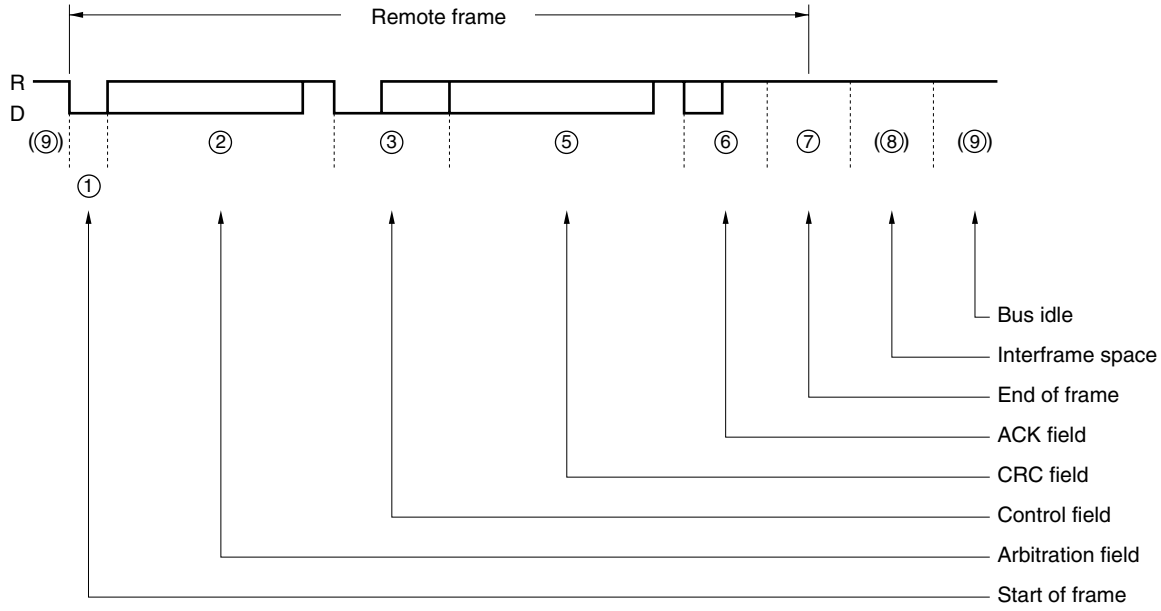


Figure 15-2. Remote Frame



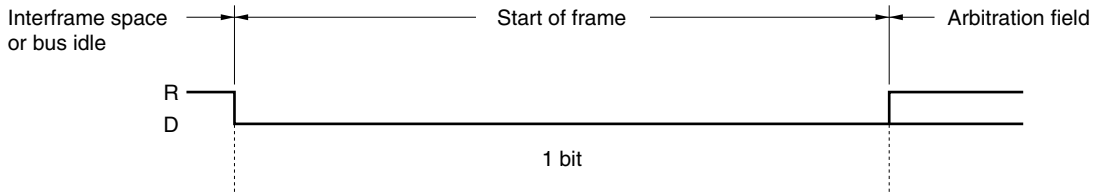
Caution The remote frame is transmitted when the reception node requests transmission. The data field is not transmitted even if the data length code ≠ 0 in the control field.

The description of each field is shown below.

(1) Start of frame

Start of frame is a frame that indicates the start of the data frame or remote frame.

Figure 15-3. Start of Frame



- The start of frame is denoted by the falling edge of the bus signal.
- Reception continues when a dominant level is detected at the sample point.
- The bus changes to the idle state when a recessive level is detected at the sample point.

(2) Arbitration field

The arbitration field sets priority, data frame/remote frame, and protocol mode.

Figure 15-4. Arbitration Field (Standard Format Mode)

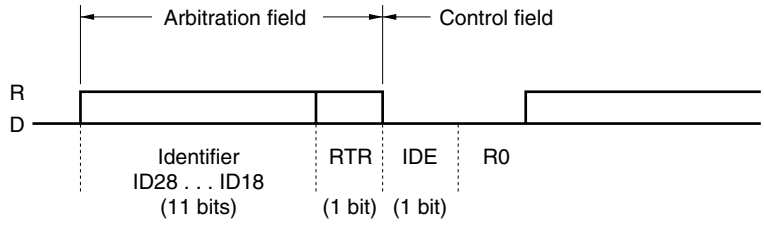
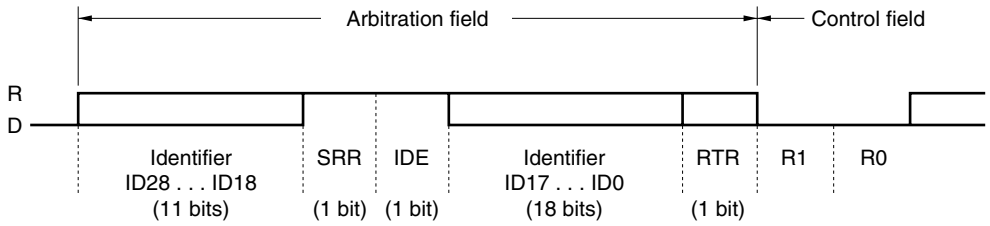


Figure 15-5. Arbitration Field (Expanded Format Mode)



- ID28 through ID0 are the identifiers.
- An identifier is transmitted by MSB first.

Table 15-2. Bit Number of Identifier

Protocol Mode	Bit Number
Standard format mode	11 bits
Expanded format mode	29 bits

Table 15-3. RTR Setting

Frame Type	RTR Bit
Data frame	0
Remote frame	1

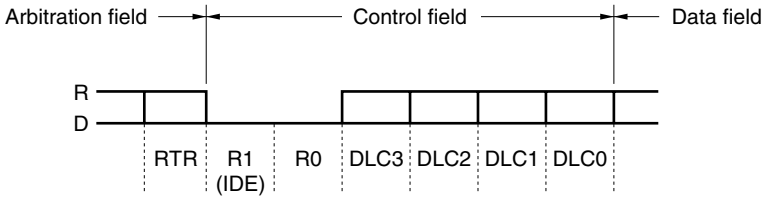
Table 15-4. Mode Setting

Protocol Mode	IDE Bit
Standard format mode	0
Expanded format mode	1

(3) Control field

The control field sets the number of data bytes in the data field.

Figure 15-6. Control Field



The IDE bit and R1 bit in the arbitration field are the same in the standard format mode.

Table 15-5. Data Length Code Setting

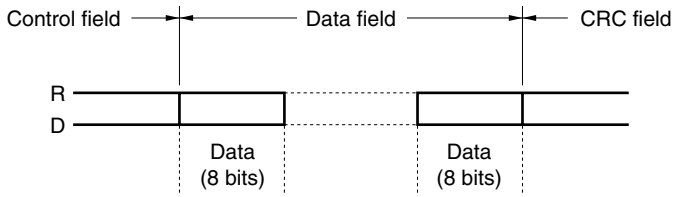
Data Length Code				Number of Data Bytes
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0
0	0	0	1	1
⋮	⋮	⋮	⋮	⋮
0	1	1	1	7
1	0	0	0	8

Caution In the case of the remote frame, data field is not generated even if data length code ≠ 0.

(4) Data field

The data field is the data group of the number of data bytes set in the control field. Up to 8 data bytes can be set.

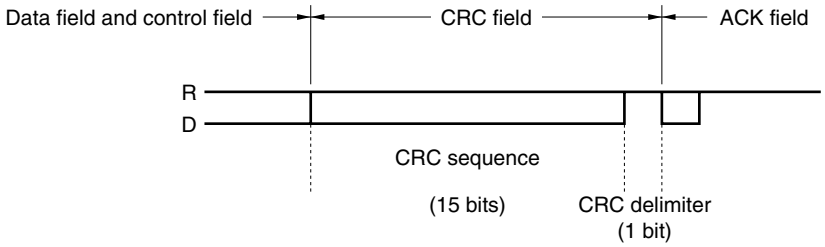
Figure 15-7. Data Field



(5) CRC field

The CRC field is a 15-bit CRC sequence for checking transmission data errors.

Figure 15-8. CRC Field

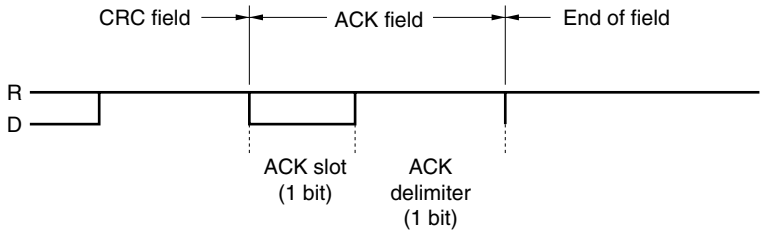


- The 15-bit CRC generation polynomial is expressed by $P(X) = X^{15} + X^{14} + X^{10} + X^8 + X^7 + X^4 + X^3 + 1$.
- Transmission node: Transmits the CRC sequence calculated from the basic data that have not been bit-stuffed the start of frame, arbitration field, control field, and data field.
- Reception node: Compares the CRC sequence calculated from data bits that do not include stuff bits of receive data and the CRC sequence in the CRC field. When these do not match, the node shifts to the error frame.

(6) ACK field

ACK field is the field for checking that the reception is normal.

Figure 15-9. ACK Field

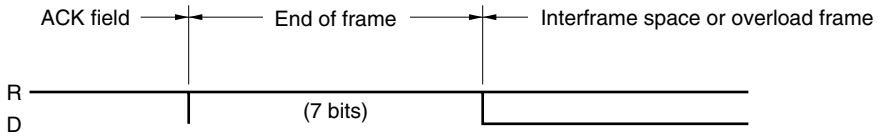


- The receive node sets the ACK slot to dominant level if no error was detected.
- The transmission node outputs a recessive level of 2 bits, and checks the reception state of the reception node.

(7) End of frame

The end of frame indicates the end of transmission/reception.

Figure 15-10. End of Frame



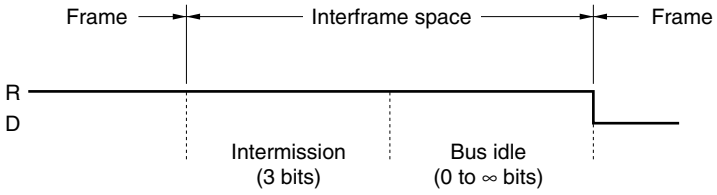
(8) Interframe space

This is inserted between the data frame, remote frame, error frame, overload frame, and the next frame in order to indicate partitions between each frame.

(a) Error active

This consists of a 3-bit intermission and a bus idle.

Figure 15-11. Interframe Space (Error Active)



(b) Error passive

This consists of an intermission, transmission suspend, and bus idle.

Figure 15-12. Interframe Space (Error Passive)

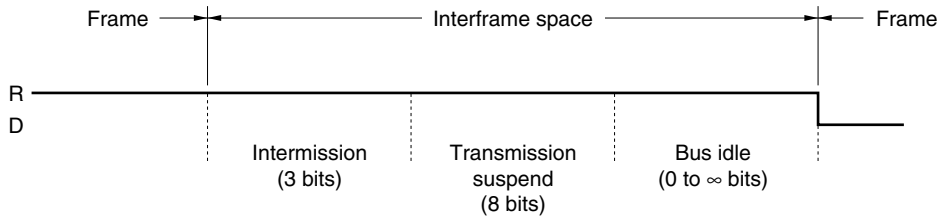


Table 15-6. Bit Length of Intermission

Protocol Mode	Bit Length
Standard format mode	3 bits

Table 15-7. Operation in the Error State

Error State	Operation
Error active	Each node that becomes bus idle changes to the transmit-enable state. The node that has requested transmission starts transmission.
Error passive	After the bus idle has continued for 8 bits, the node changes to the transmit-enable state. Once other nodes have begun transmission it changes to receive-enable.

15.1.4 Error frame

This frame is output from the node if an error is detected.

When other nodes output a dominant level to flag the passive error, the dominant level continues for 6 consecutive bits. The passive error flag consists of 6 consecutive recessive bits unless a dominant bit from other nodes overwrites it.

Figure 15-13. Error Frame

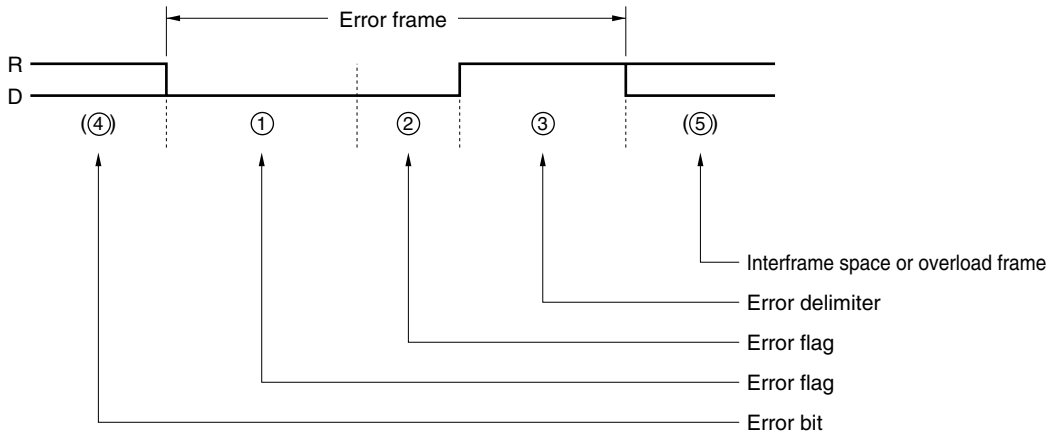


Table 15-8. Definition of Each Field (Error Frame)

No.	Name	Bit Number	Definition
<1>	Error flag	6	Error active node: outputs 6 bits of dominant level continuously. Error passive node: outputs 6 bits of recessive level continuously.
<2>	Error flag	0 to 6	The node that received the error flag detects the bit stuff error and outputs the error flag again.
<3>	Error delimiter	8	Outputs 8 bits recessive level continuously. If the dominant level was monitored at the 8th bit, the overload frame is transmitted from the next bit.
<4>	Error bit	–	Is output following the bit where error occurred (in case of a CRC error, it is output following the ACK delimiter).
<5>	Interframe space/ overload frame	3/14 20 MAX	Interframe space or overload frame continues.

15.1.5 Overload frame

This frame is output from the first bit of the intermission when the reception node has not completed reception. When a bit error is detected in the intermission, this frame is output following the next bit after the bit error detection.

Figure 15-14. Overload Frame

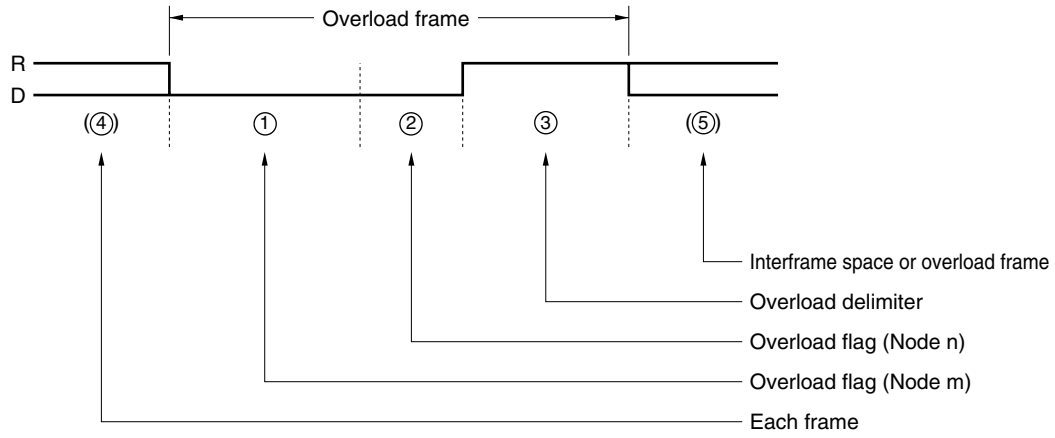


Table 15-9. Definition of Each Frame (Overload Frame)

No.	Name	Bit Number	Definition
<1>	Overload flag	6	Outputs 6 bits of dominant level continuously.
<2>	Overload flag from node n	0 to 6	Node n, which receives the overload flag in the interframe space, outputs the overload flag.
<3>	Overload delimiter	8	Outputs 8 bits of recessive level continuously. If the dominant level was monitored at the 8th bit, the overload frame is transmitted from the next bit.
<4>	Each frame	–	Is output following the end of frame, error delimiter, and overload delimiter.
<5>	Interframe space/ overload frame	3/14 20 MAX	Interframe space or overload frame continues.

Remark Node n/node m means any node.

15.2 Functions

15.2.1 Bus priority decision

(1) When one node starts transmission

- In bus idle, the node that output the data first transmits.

(2) When two or more nodes start transmission

- The node with the lower identifier has priority
- The transmission node compares its output arbitration field with the data level on the bus.
- A node loses arbitration if it outputs a recessive level and a dominant level on the bus is acknowledged.

Table 15-10. Bus Priority Decision

Conformity of Level	Continuous transmission
Non-Conformity of Level	The data output is stopped from the next bit and reception operation starts.

(3) Priority of data frame and remote frame

- When the data frame and remote frame are in contention on the bus, the data frame in which the final bit RTR is dominant level has priority.

15.2.2 Bit stuffing

When the same level continues for more than 5 bits, bit stuffing (inserting 1 bit with inverse data) takes place to prevent a burst error.

Table 15-11. Bit Stuffing

Transmission	During transmission of a data frame or a remote frame, when the same level continues for 5 bits in the data between the start of frame and ACK field, 1 bit with reverse level of the previous 5 bits of data is inserted before the next bit.
Reception	During reception of a data frame or a remote frame, when the same level continues for 5 bits in the data between the start of frame and ACK field, the reception is continued after deleting the next 1 bit.

15.2.3 Multi master

As the bus priority is determined by the identifier, any node can be the bus master.

15.2.4 Multi cast

Although there is only one transmission node, because it is possible to set the same identifier to more than one node, two or more nodes can simultaneously receive data.

15.2.5 Sleep mode/stop mode function

Using the sleep/stop modes, the DCAN controller can be put in waiting mode, which will reduce the power consumption. The sleep mode can be set using the same procedure as in the CAN specification.

Whereas the SLEEP mode can be woken up by bus operation, the STOP mode cannot (it is controlled by the CPU).

15.2.6 Error control function

(1) Error types

Table 15-12. Error Types

Type	Description of Error		Detection State	
	Detection Method	Detection Condition	Transmission/ Reception	Field/Frame
Bit error	Comparison of output level and level on the bus (except stuff bit)	Mismatch of both levels	Transmission/ reception node	Bit that output data on the bus at the start of frame to the end of frame, error frame, and overload frame.
Stuff error	Check of the reception data by the stuff bit	Continuous 6 bits of the same level data	Transmission/ reception node	Start of frame to CRC sequence
CRC error	Comparison of the CRC generated from the reception data and received CRC sequence	CRC Mismatch	Reception node	Start of frame to data field
Form error	Field/frame check of the fixed format	Detection of a fixed format error	Reception node	• CRC delimiter • ACK field • End of frame • Error frame • Overload frame
ACK error	Check of the ACK slot by the transmission node	Detection of recessive level in ACK slot	Transmission node	ACK slot

(2) Output timing of the error frame

Table 15-13. Output Timing of the Error Frame

Type	Output Timing
Bit error, stuff error, form error, ACK error	An error frame is output from the bit timing following that which detected an error
CRC error	An error frame is output from the bit timing following that of the ACK delimiter

(3) Processing when error occurs

The transmission node re-transmits the data frame or the remote frame after the error frame.

(4) Error state

(a) Types of error state

- There are three types of error state: Error active, error passive, and bus off.
- The transmission error counter and the reception error counter control the error state.
- The error counter is incremented at each error.
- The output error flag is different depending on whether the operation of the error state is transmission or reception.
- If the value of the error counter exceeds 96, a sign indicating that the warning level for an error passive has been reached will be displayed.
- When only one node is active at start-up, ACK is not returned even if data is transmitted, therefore the re-transmission of the error frame and data is repeated. In this case, the state does not become bus off. Neither is the bus off state entered even if the error state is repeated after a node has transmitted a wake up message.
- A reception operation can be executed even if the transmission operation is in the bus off state.

Table 15-14. Types of Error State

Type	Operation	Value of Error Counter	Output Error Flag Type
Error active	Transmission/ reception	0 to 127	Active error flag (consecutive 6 bits of dominant level)
Error passive	Transmission	128 to 255	Passive error flag (consecutive 6 bits of recessive level)
	Reception	128 or more	
Bus off	Transmission	256 or more	Communication cannot be made When the recessive level is generated 128 times in 11 bits, the state can be returned to error active by error counter = 0.
	Reception	–	Does not exist

(b) Error counter

The error counter counts up when an error has occurred, and counts down when transmission and reception operate normally. The timing of the count up and count down is first bit of the error delimiter.

Table 15-15. Error Counter

State	Transmission Error Counter	Reception Error Counter
When the reception node detects an error (except a bit error in the active error flag or overload flag).	No change	+1
When the reception node detects a dominant level next to the error flag of the error frame.	No change	+8
When transmission node transmits an error flag [Error counter = ±0] <1> When an ACK error is detected in the error passive state and dominant level is not detected during output of the passive error flag. <2> Stuff error generation in arbitration field.	+8	No change
Bit error detection during output of active error flag and overload flag (transmission node of error active).	+8	No change
Bit error detection during output of active error flag and overload flag (reception node of error active).	No change	+8
When each node detects fourteen continuous recessive levels from the beginning of the active error flag or overload flag, and every time eight continuous dominant levels after that are detected. Every time when each node detects eight continuous dominant levels following the passive error flag.	+8	+8
When the transmission node has completed transmission without error. (±0 when error counter = 0)	–1	No change
When the reception node has completed reception without error.	No change	• –1 (1 ≤ REC[Note] ≤ 127) • ±0 (REC = 0) • Sets 127 (REC > 127)

Note REC: Reception error counter

(c) Bit error generation during intermission

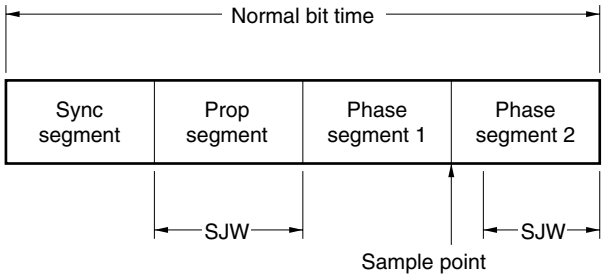
Overload frame is generated.

15.2.7 Baud rate control function

(1) Normal bit time (8 to 25 time quantum)

- The definition of 1 data bit time is as follows.

Figure 15-15. Normal Bit Time (8 to 25 Time Quantum)



[1 Minimum Time Quantum = 1/fx]

Sync segment	This segment becomes active when bit synchronization occurs.
Prop segment	This segment is for absorbing delays of the output buffer, DCAN bus, and input buffer. It is set to make ACK return by the time phase segment 1 starts. Prop segment time ≥ (output buffer delay) + (DCAN bus delay) + (input buffer delay).
Phase segment 1, Phase segment 2	This segment is for compensating the data bit time error. The larger the size, the larger the tolerable error, but the slower the communication speed.
SJW	Short for re-Synchronization Jump Width. This bit sets the bit synchronization range.

Table 15-16. Segment Name and Segment Length

Segment Name	Segment Length
Sync segment (Synchronization segment)	1
Prop segment (Propagation segment)	Programmable 1 to 8
Phase segment 1 (Phase buffer segment 1)	Programmable 1 to 8
Phase segment 2 (Phase buffer segment 2)	Maximum value of phase segment 1 + IPT[Note] (IPT = 0 to 2)
SJW	Programmable 1 to 4

Note IPT : Information processing time

(2) Adjusting data bit synchronization

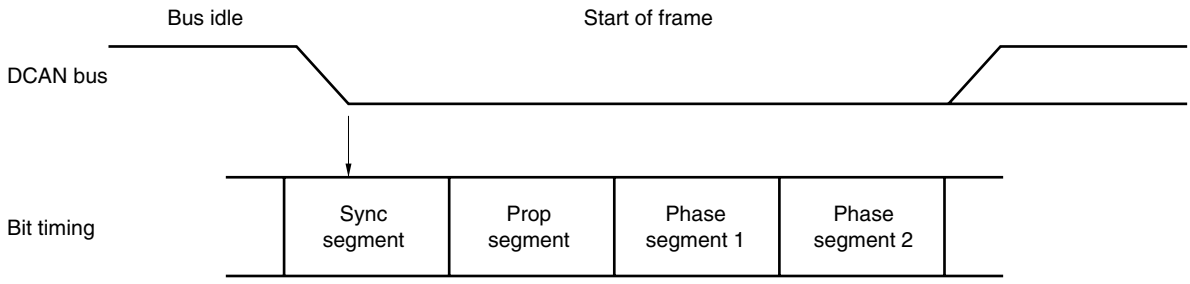
- The transmission node transmits data in synchronization with the transmission node bit timing.
- The reception node synchronizes with the changing level on the bus by means of hardware or software synchronization.

(a) Hardware synchronization

Bit synchronization when the reception node detects the start of frame in the bus idle state.

- When a falling edge is detected on the bus, that bit and the next bit become sync segment and prop segment respectively. In this case, the synchronization is not related to SJW.
- After reset or wake up, because it is necessary to synchronize the bits, only the first level change on the bus is taken for hardware synchronization (afterwards, following bit synchronization is performed).

Figure 15-16. Adjusting Synchronization of the Data Bit

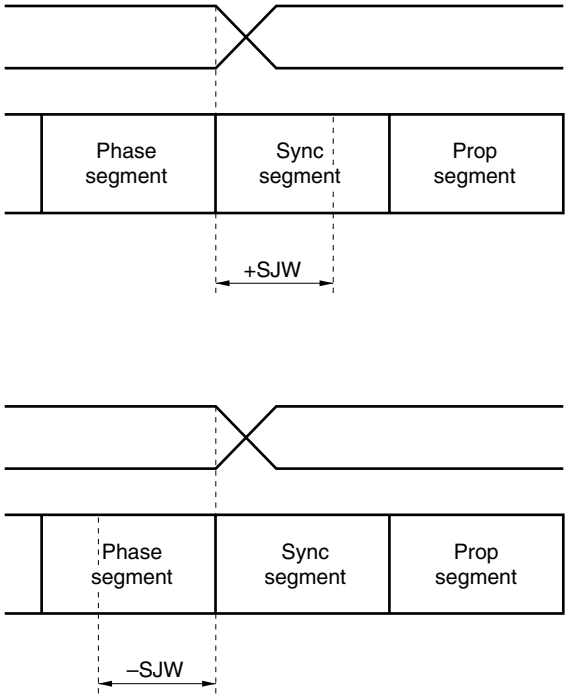


(b) Bit synchronization

When a level change on the bus is detected during reception, bit synchronization is performed.

- 2 types of synchronization can be performed.
  Normal operation: Level falling edge
  Low speed operation: Level falling and rising edges
- Synchronization is performed only when an edge is detected in the bit timing that is specified by SJW.
- The data sample point at the reception node shifts relatively according to a baud rate discrepancy between the transmission node and reception node.
- The allowable range of discrepancy is defined as 'SJW'. The range of SJW is set in front of and behind (+/– of baud rate) the sync segment. When an edge is generated in the SJW range, synchronization is achieved. When an edge is generated out of the SJW range, synchronization is not achieved.
- The bit detected at the edge becomes the mandatory sync segment, and the next one becomes the prop segment, whereupon the bit rate starts again.

Figure 15-17. Bit Synchronization



15.2.8 State shift chart

Figure 15-18. Transmission State Shift Chart

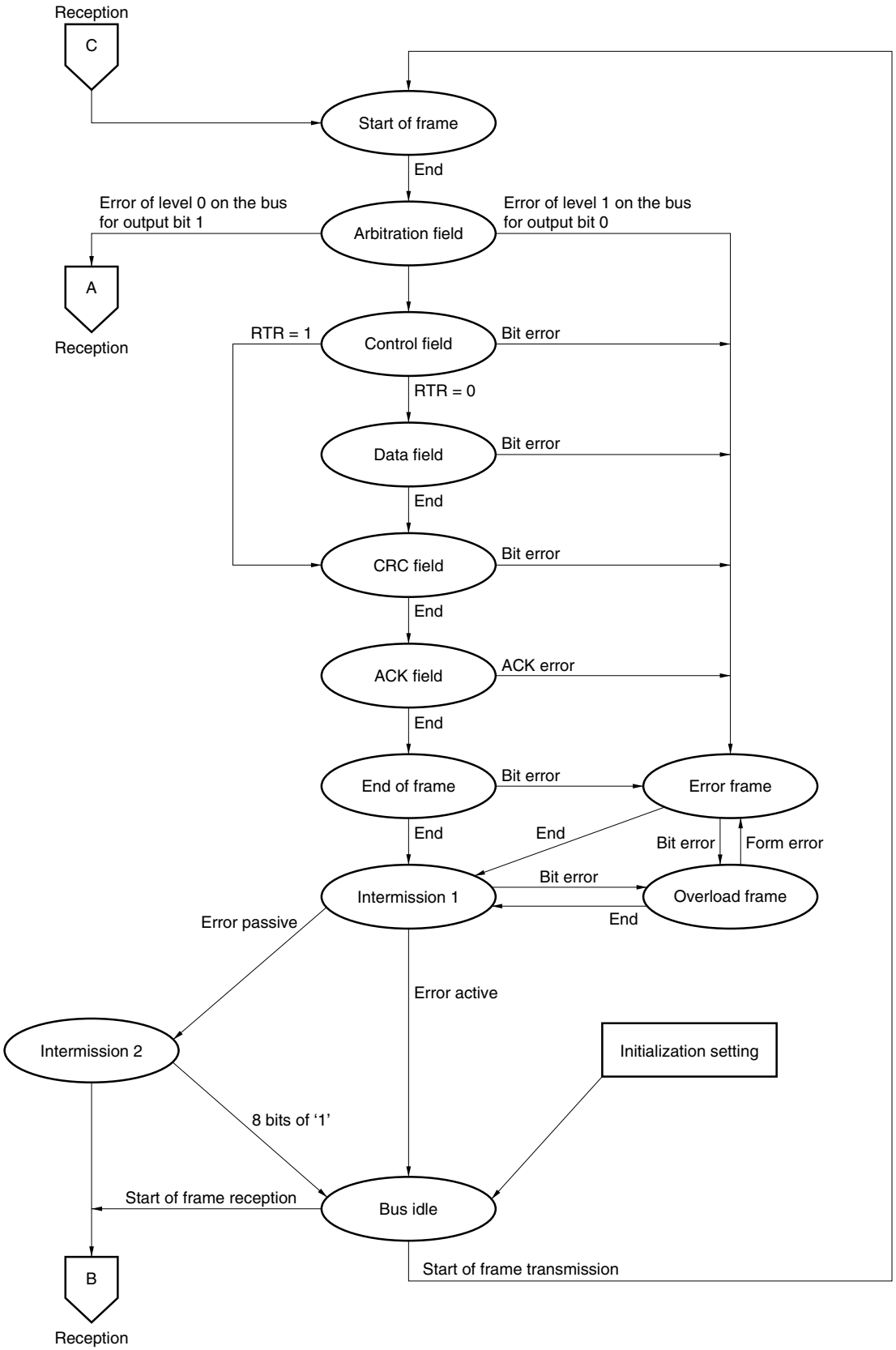


Figure 15-19. Reception State Shift Chart

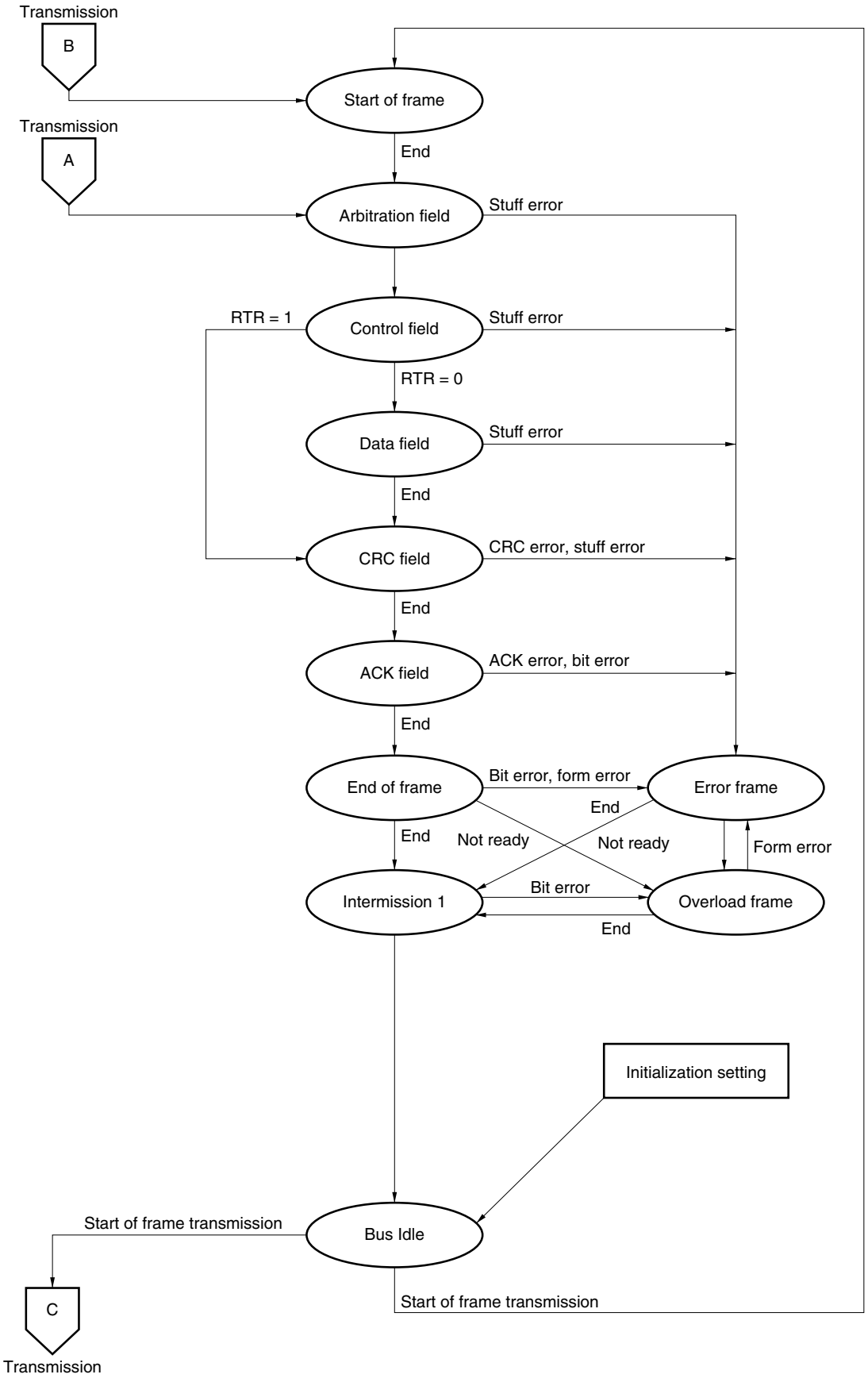
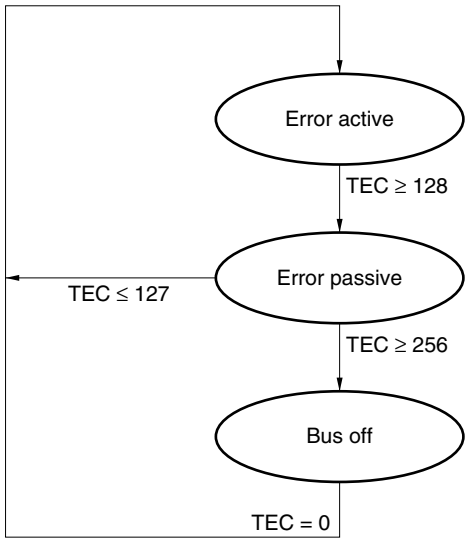


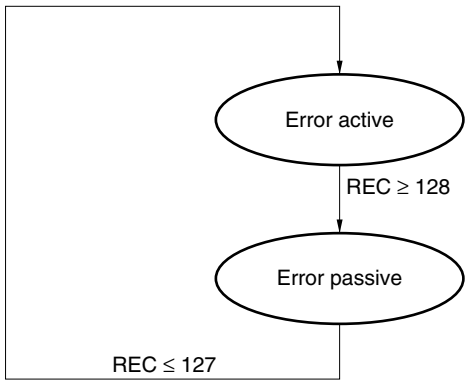
Figure 15-20. Error State Shift Chart

(a) Transmission



Remark TEC = Transmission error counter

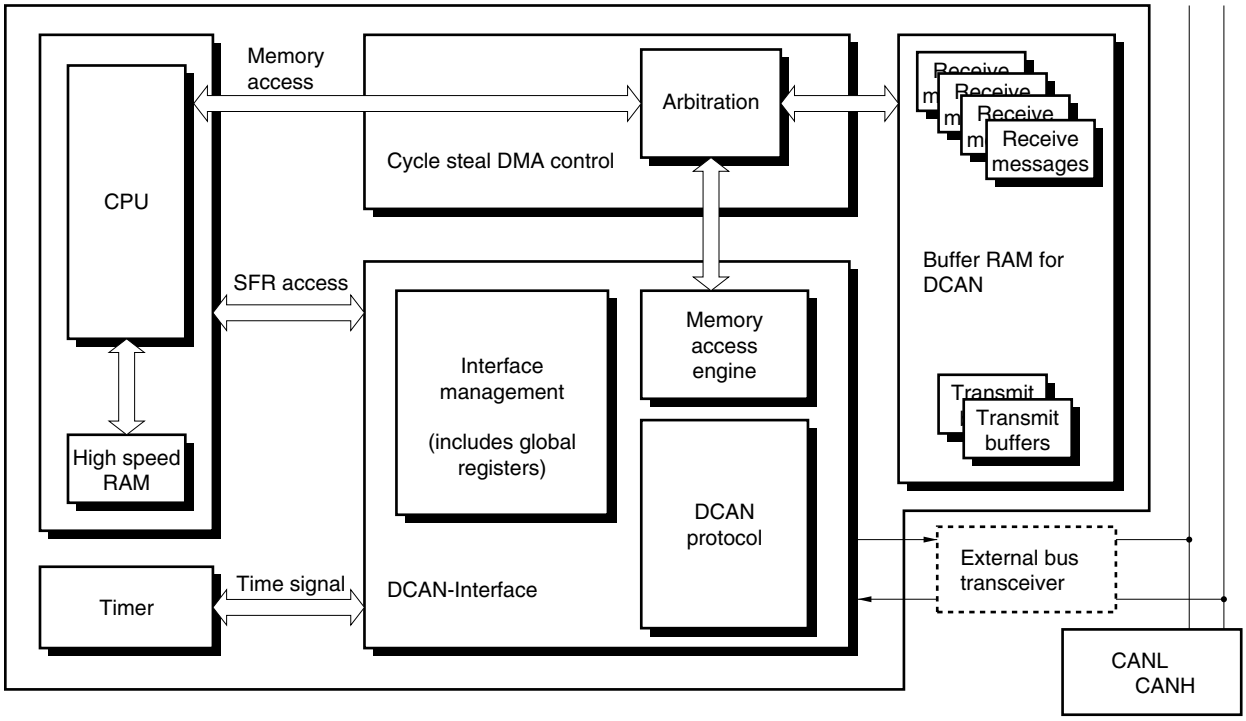
(b) Reception



Remark REC = Reception error counter

15.3 Outline

Figure 15-21. Block Diagram



The DCAN interface section processes all protocol operations by means of the DCAN protocol section hardware.

The memory access engine either fetches the DCAN protocol data transmitted from a specific RAM area and transfers it to the DCAN protocol section, or compares and sorts the fetched data and then stores it in a predefined RAM area.

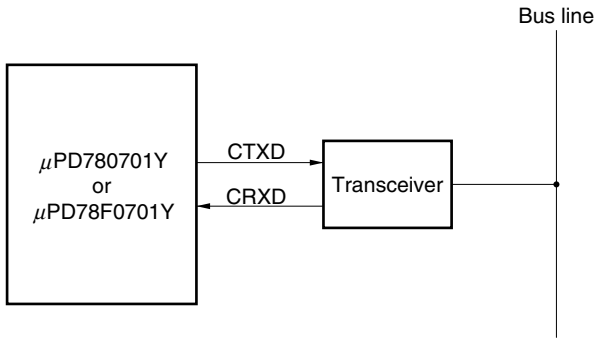
The DCAN directly accesses the CPU area, which can be accessed by both the DCAN and the CPU, without any effect on CPU operation.

The DCAN part operates with an external bus transceiver which converts the transmit data and receive data lines to the electrical characteristics of the DCAN bus itself.

15.4 Connection with Target System

The μPD780701Y, 78F0701Y has to be connected to the DCAN bus with an external transceiver.

Figure 15-22. Connection to the DCAN Bus



15.5 DCAN Controller Configuration

The DCAN controller consists of the following hardware.

Table 15-17. DCAN Controller Configuration

Item	Configuration
Message definition	In buffer RAM area for DCAN (288 bytes)
DCAN input/output	CTXD CRXD
Control register	CAN control register (CANC) Transmit control register (TRC) Received message register (RMES) Redefinition control register (REDEF) CAN error status register (CANES) Transmit error counter (TEC) Receive error counter (REC) Message count register (MCNT) Bit rate prescaler (BRPRS) Synchronous control register 0 (SYNC0) Synchronous control register 1 (SYNC1) Mask control register (MASKC)

15.6 Special Function Registers (SFR) for DCAN Controller

Table 15-18. SFR Definitions

Address	Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFC0H	CAN control register	CANC	R/W	√	√	–	01H
FFC1H	Transmit control register	TCR		–	√	–	00H
FFC2H	Received message register	RMES	R	–	√	–	
FFC3H	Redefinition control register	REDEF	R/W	√	√	–	
FFC4H	CAN error status register	CANES		–	√	–	
FFC5H	Transmit error counter	TEC	R	–	√	–	
FFC6H	Receive error counter	REC		–	√	–	
FFC7H	Message count register	MCNT		–	√	–	C0H
FFC8H	Bit rate prescaler	BRPRS	R/W	–	√	–	00H
FFC9H	Synchronous control register 0	SYNC0		–	√	–	18H
FFCAH	Synchronous control register 1	SYNC1		–	√	–	0EH
FFCBH	Mask control register	MASKC		–	√	–	00H

The following SFR-bits can be set with a 1-bit memory manipulation instruction. The other SFR-registers are set with an 8-bit memory manipulation instruction.

Table 15-19. SFR Bit Definitions

Name	Description	Register.Bit
SOFE	SOFOUT operation setting flag	CANC.4
SLEEP	DCAN sleep mode setting flag	CANC.2
INIT	Initialization mode setting flag	CANC.0
DEF	Redefinition enable flag	REDEF.7

15.7 Message and Buffer Configuration

Table 15-20. Message and Buffer Configuration

Address	Register Name	R/W	After Reset
F900H	Transmit buffer 0	R/W	Note
F910H	Transmit buffer 1	R/W	Note
F920H	Receive message 0 / Mask 0	R/W	Note
F930H	Receive message 1	R/W	Note
F940H	Receive message 2 / Mask 1	R/W	Note
F950H	Receive message 3	R/W	Note
F960H	Receive message 4	R/W	Note
F970H	Receive message 5	R/W	Note
F980H	Receive message 6	R/W	Note
F990H	Receive message 7	R/W	Note
F9A0H	Receive message 8	R/W	Note
F9B0H	Receive message 9	R/W	Note
F9C0H	Receive message 10	R/W	Note
F9D0H	Receive message 11	R/W	Note
F9E0H	Receive message 12	R/W	Note
F9F0H	Receive message 13	R/W	Note
FA00H	Receive message 14	R/W	Note
FA10H	Receive message 15	R/W	Note

Note The contents are undefined because the data resides in a normal RAM area.

15.8 Transmit Buffer Configuration

The DCAN has two independent transmit buffers. The two buffers have a 16-byte configuration, support standard and extended frames. Transmit message data (8 bytes) can be set in the transmit buffers. The configuration of the transmit buffers is similar to that of the receive buffers. Addresses, message data addresses, and transmit buffers that are unused by DCAN can be used as usual RAM. The control bits, identification data, and message data are stored in the message RAM area.

Transmission is controlled by the transmission control register (TCR). A transmit request is set by setting bits 0 and 1 (TXRQ0 and TXRQ1) of TCR to 1.

If there are two transmissions, it is necessary to set the priority order prior to transmission by setting bit 7 (TXP) of TCR.

15.9 Transmit Message

Table 15-21. Transmit Message Configuration

Name	Address[Note]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TCON	n0H	IDE	RTR	0	0	DLC3	DLC2	DLC1	DLC0
	n1H	Unused							
IDTX0	n2H	ID standard section							
IDTX1	n3H	ID standard section			0	0	0	0	0
IDTX2	n4H	ID extended section							
IDTX3	n5H	ID extended section							
IDTX4	n6H	ID extended section		0	0	0	0	0	0
	n7H	Unused							
DATA0	n8H	Message data byte 0							
DATA1	n9H	Message data byte 1							
DATA2	nAH	Message data byte 2							
DATA3	nBH	Message data byte 3							
DATA4	nCH	Message data byte 4							
DATA5	nDH	Message data byte 5							
DATA6	nEH	Message data byte 6							
DATA7	nFH	Message data byte 7							

Note This address is expressed as an offset relative to the starting address of the transfer buffer.

15.9.1 Transmit message specification section

The transmit message specification section (TCON) corresponds to the message specification bits of the control field of the DCAN protocol.

TCON is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TCON to undefined.

Figure 15-23. Format of Transmit Message Specification Section (TCON)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCON	IDE	RTR	0	0	DLC3	DLC2	DLC1	DLC0	xxx0H	Undefined	R/W

IDE	Identifier extension select
0	Transmit standard frame message (11-bit identifier)
1	Transmit extended frame message (29-bit identifier)

RTR	Remote frame transmission select
0	Transmit data frames
1	Transmit remote frames

DLC3	DLC2	DLC1	DLC0	Data length code selection of transmit message
0	0	0	0	0 data bytes
0	0	0	1	1 data byte
0	0	1	0	2 data bytes
0	0	1	1	3 data bytes
0	1	0	0	4 data bytes
0	1	0	1	5 data bytes
0	1	1	0	6 data bytes
0	1	1	1	7 data bytes
1	0	0	0	8 data bytes
Other than above				**Note**

Note The data length code selects number of bytes to be transmitted. Valid entries for the data length code are 0 to 8. If a value greater than 8 is selected, 8 bytes are transmitted in the data frame along with the data length code specified by DLC3 to DLC0.

Remark The control section describes the type of frame that is generated and its length. The reserved bits of the DCAN protocol are always transferred in the dominant state (0).

15.9.2 Transmit identifier section

The transmit identifier section (IDTX0 to IDTX4) is used to set the message identifier in the arbitration field of the DCAN protocol.

IDTX0 to IDTX4 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets IDTX0 to IDTX4 to undefined.

Figure 15-24. Format of Transmit Identifier Section (IDTX0 to IDTX4)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDTX0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxx2H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDTX1	ID20	ID19	ID18	0	0	0	0	0	xxx3H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDTX2	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	xxx4H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDTX3	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	xxx5H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDTX4	ID1	ID0	0	0	0	0	0	0	xxx6H	Undefined	R/W

15.9.3 Transmit data section

The transmit data section (DATA0 to DATA7) is used to set the transmit message data of the data field in the DCAN protocol.

DATA0 to DATA7 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets DATA0 to DATA7 to undefined.

Figure 15-25. Format of Transmit Data Section (DATA0 to DATA7)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA0									xxx8H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA1									xxx9H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA2									xxxAH	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA3									xxxBH	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA4									xxxCH	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA5									xxxDH	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA6									xxxEH	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA7									xxxFH	Undefined	R/W

15.10 Receive Message Buffer Configuration

The DCAN has 16 receive message buffers. The number of buffers to be used is specified by the MCNT register. Unused buffers can be used as usual RAM. The receive messages are stored directly in the RAM area for DCAN.

The 16 receive message buffers have a 16-byte configuration and support standard and extended frames. Eight bytes of receive message data can be stored in a receive message buffer. The configuration of the receive buffers is similar that of the transmit buffers.

Reception detection and data processing are reliably performed using bits 6 and 7 (MUC and DN) of the receive status register (DSTAT). When there are 8 initial receive message buffers, it is also possible to confirm the end of reception with the DNn flag of the receive message register (RMES).

The receive interrupt request can be enabled/disabled separately for each buffer used.

15.11 Receive Message

Table 15-22. Receive Message Configuration

Name	Address[Note]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IDCON	n0H	0	0	0	0	0	ENI	RTR	IDE
DSTAT	n1H	DN	MUC	R1	R0	DLC			
IDREC0	n2H	ID standard section							
IDREC1	n3H	ID standard section			0	0	0	0	0
IDREC2	n4H	ID extended section							
IDREC3	n5H	ID extended section							
IDREC4	n6H	ID extended section		0	0	0	0	0	0
	n7H	Unused							
DATA0	n8H	Message data byte 0							
DATA1	n9H	Message data byte 1							
DATA2	nAH	Message data byte 2							
DATA3	nBH	Message data byte 3							
DATA4	nCH	Message data byte 4							
DATA5	nDH	Message data byte 5							
DATA6	nEH	Message data byte 6							
DATA7	nFH	Message data byte 7							

Note This address is expressed as an offset relative to the starting address of the receive buffer.

15.11.1 Receive control specification section

The receive control specification section (IDCON) is used to set the receive control bits of the control field of the DCAN protocol.

IDCON can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets IDCON to undefined.

Figure 15-26. Format of Receive Control Specification Section (IDCON)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDCON	0	0	0	0	0	ENI	RTR	IDE	xxx0H	Undefined	R/W

ENI	Receive interrupt enable setting
0	No interrupt generated
1	Generate receive interrupt after reception of valid message

RTR	Remote frame receive setting
0	Receive data frames
1	Receive remote frames

IDE	Identifier extension setting
0	Receive standard frame message (11-bit identifier)
1	Receive extended frame message (29-bit identifier)

IDCON specifies the type of message received in the associated buffer.

Caution When new data is entered in IDCON, it is necessary to specify whether a receive interrupt is required or not.

15.11.2 Receive status section

The receive status section (DSTAT) is used to set the receive status bits of the arbitration field of the DCAN protocol.

The status of the receive message is reflected in DSTAT, which also signals whether new data has been stored in the receive confirmation flag message buffer, or if the DCAN is currently transferring data into this buffer.

DSTAT can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets DSTAT to undefined.

Figure 15-27. Format of Receive Status Section (DSTAT) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DSTAT	DN	MUC	R1	R0	DLC3	DLC2	DLC1	DLC0	xxx1H	Undefined	R/W

DN	Data update
0	No change in data (readout from the receive message buffer has finished)
1	Data changed (no readout from receive message buffer)
The DCAN controller sets DN to 1 when transfer of the message to the buffer begins. The DN bit must be cleared when the data is readout from the buffer.	

MUC	Memory update
0	DCAN is not accessing the data section
1	DCAN is transferring new data to the message buffer
The DCAN controller sets MUC to 1 when transfer of the message to the buffer begins, and clears it to 0 when transfer is complete.	

R1	Reserved bit 1
0	Reserved bit 1 of the receive message was 0
1	Reserved bit 1 of the receive message was 1

R0	Reserved bit 0
0	Reserved bit 0 of the receive message was 0
1	Reserved bit 0 of the receive message was 1

Figure 15-27. Format of Receive Status Section (DSTAT) (2/2)

DLC3	DLC2	DLC1	DLC0	Data length code of receive message
0	0	0	0	0 data bytes
0	0	0	1	1 data byte
0	0	1	0	2 data bytes
0	0	1	1	3 data bytes
0	1	0	0	4 data bytes
0	1	0	1	5 data bytes
0	1	1	0	6 data bytes
0	1	1	1	7 data bytes
1	0	0	0	8 data bytes
Other than above				**Note**

On initial access to the DSTAT buffer area, DN = 1, MUC = 1, and the reserved bits and DLC are written.
On final access to the DSTAT buffer area, DN = 1, MUC = 0, and the reserved bits and DLC are written.

Note Valid entries for the data length code are 0 to 8. If a value higher than 8 is received, 8 bytes are received in the data frame along with the data length code specified by DLC3 to DLC0.

15.11.3 Receive identifier specification section

The receive identifier specification section (IDREC0 to IDREC4) is used to set the receive identifier specification of the control field of the DCAN protocol.

IDREC0 to IDREC4 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets IDREC0 to IDREC4 to undefined .

Figure 15-28. Format of Receive Identifier Specification Section (IDREC0 to IDREC4)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDREC0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxx2H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDREC1	ID20	ID19	ID18	0	0	0	0	0	xxx3H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDREC2	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	xxx4H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDREC3	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	xxx5H	Undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IDREC4	ID1	ID0	0	0	0	0	0	0	xxx6H	Undefined	R/W

The identifier of the receive message buffer has to be specified during the initialization of the DCAN.

The DCAN uses the IDREC0 to IDREC4 section for comparison with the message identifier received on the DCAN bus.

The identifier of the receive message buffers should be changed in the initialization phase, otherwise the receive buffer is set to redefinition by the redefinition control register (REDEF). Moreover, if the identifier is changed while a message is being stored in the buffer following an identifier comparison match, the data will not be stored correctly.

Remark The unused parts of the identifier may be written to 0 by the DCAN.

15.11.4 Receive message data section

The receive message data section (DATA0 to DATA7) can be used to confirm the receive message data of the DCAN protocol.

DATA0 to DATA7 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets DATA0 to DATA7 to undefined.

Figure 15-29. Format of Receive Message Data Section (DATA0 to DATA7)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DATA0									xxx8H	Undefined	R/W
DATA1									xxx9H	Undefined	R/W
DATA2									xxxAH	Undefined	R/W
DATA3									xxxBH	Undefined	R/W
DATA4									xxxCH	Undefined	R/W
DATA5									xxxDH	Undefined	R/W
DATA6									xxxEH	Undefined	R/W
DATA7									xxxFH	Undefined	R/W

The DCAN stores receive data in the memory area. Only those message data bytes that match the identifier in the messages received from the DCAN bus are stored in the receive buffer memory area (DATA0 to DATA7).

If the data length code (DLC0 to DLC3) of the receive status (DSTAT) is less than eight, the DCAN will not write the additional bytes up to eight. The DCAN stores a maximum of 8 bytes (conforming to the DCAN protocol rules) even when the value of DLC0 to DLC3 in the control field set by the received frame is greater than eight.

15.12 Mask Function

Table 15-23. Mask Function

Name	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MCON	n0H	0	0	0	0	0	0	RTR	0
	n1H	Unused							
MREC0	n2H	ID standard section							
MREC1	n3H	ID standard section			0	0	0	0	0
MREC2	n4H	ID extended section							
MREC3	n5H	ID extended section							
MREC4	n6H	ID extended section		0	0	0	0	0	0
	n7H	Unused							
	n8H	Unused							
	n9H	Unused							
	nAH	Unused							
	nBH	Unused							
	nCH	Unused							
	nDH	Unused							
	nEH	Unused							
	nFH	Unused							

Message buffers 0 and 2 can operate as receive message identifier masks by means of a mask control register (MASKC) setting.

In this case, the message buffers do not operate as normal message buffers.

Message buffer 0 masks the identifier of message buffer 1 and message buffer 2 masks the identifier of message buffer 3.

When the global mask function is selected with the mask control register (MASKC), the message buffers operate as masks of all the message buffer identifiers that have been allocated to the higher addresses of the message buffers that operate as masks.

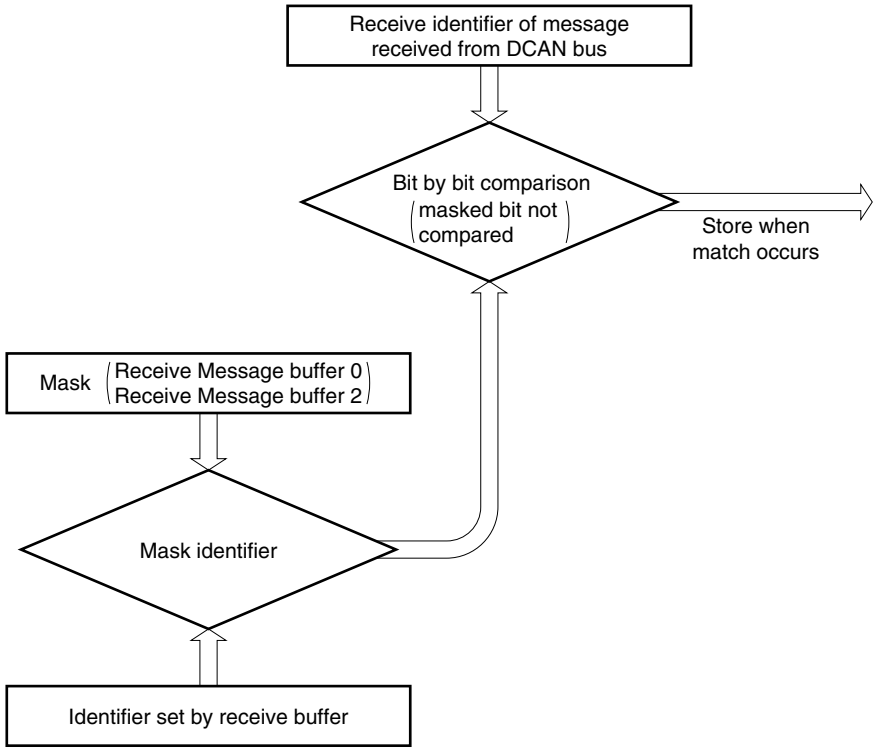
When the global mask function is selected, masks can be applied to both standard and extended frames.

15.12.1 Identifier mask

The DCAN compares the identifiers of the messages received from the DCAN bus with the identifiers of the receive message buffers, and when there is a match, stores the message in the message buffer.

When the mask function is operating, the bit that is set to 1 by the mask is ignored, and comparison is performed.

Figure 15-30. Identifier Mask



This function implements the so-called basic-DCAN operation.

The type of identifier in this case is either set to “standard” or “extended” by the setting of the IDE bit in the receive buffer. The comparison of the RTR bit can also be masked. It is possible to receive data and remote frames in the same masked receive buffer.

The following information is stored in RAM

- Identifier (11/29 bits as specified by IDE bit)
- Remote bit (RTR)
- Received bits
- Data length code (DLC)
- Data bytes as defined by DLC

Caution All writes into DCAN are byte accesses. Unused bits of the same byte will be written to zero.

15.12.2 Mask identifier control bit setting

The mask identifier control bit (MCON) is used to set the mask identifier control bits of the DCAN protocol.

MCON can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets MCON to undefined.

Figure 15-31. Control Bits for Mask Identifier

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MCON	0	0	0	0	0	0	RTR	0	xxx0H	Undefined	R/W

RTR	Remote transmission selection
0	Checks RTR bit of received message
1	Does not check RTR bit of received message

15.12.3 Mask identifier setting

The mask identifiers (MREC0 to MREC4) are used to set the mask identifier specification of the DCAN.

MREC0 to MREC4 can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets MREC0 to MREC4 to undefined.

Figure 15-32. Mask Identifier

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MREC0	MID28	MID27	MID26	MID25	MID24	MID23	MID22	MID21	xxx2H	Undefined	R/W
MREC1	MID20	MID19	MID18	0	0	0	0	0	xxx3H	Undefined	R/W
MREC2	MID17	MID16	MID15	MID14	MID13	MID12	MID11	MID10	xxx4H	Undefined	R/W
MREC3	MID9	MID8	MID7	MID6	MID5	MID4	MID3	MID2	xxx5H	Undefined	R/W
MREC4	MID1	MID0	0	0	0	0	0	0	xxx6H	Undefined	R/W

MIDn	Mask identifier bit (n = 0...28)
0	Checks IDn bit of received message
1	Does not check IDn bit of received message

15.13 DCAN Controller Control Registers

15.13.1 CAN control register (CANC)

This register sets the operation mode of the DCAN controller.

CANC can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CANC to 01H.

Figure 15-33. Format of CAN Control Register (CANC) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
CANC	RxF	TxF	0	SOFE	SOFSEL	SLEEP	STOP	INIT	FFC0H	01H	R/W

RxF	DCAN reception status flag
0	Receive operation stopped status
1	Receive operation status

TxF	DCAN transmission status flag
0	Transmit operation stopped status. Becomes TxF = 0 when transmission is complete.
1	Transmit operation status

SOFE	SOFOUT operation setting flag
0	SOFOUT unchanged
1	SOFOUT toggles according to selection mode

SOFSEL	SOFOUT output type selection flag
0	SOFOUT operates as time stamp function
1	SOFOUT operates as global time function. Indicated as SOF on the bus.

SLEEP	DCAN sleep mode selection flag
0	Standard operation
1	Sets the DCAN sleep/stop mode (sleep/stop mode is set by bit 1 (STOP) of CANC. Can be woken up from any DCAN bus operation.)

STOP	Sleep/stop mode selection flag
0	Sleep operation Sleep mode is woken up by detection of a transition on the DCAN bus.
1	Stop mode operation Stop mode can only be released by CPU access. It cannot be woken up from the DCAN bus.

Figure 15-33. Format of CAN Control Register (CANC) (2/2)

INIT	Initialization mode setting flag
0	Standard operation
1	Initialization mode

- This flag sets the DCAN controller to initialization mode.
- The DCAN controller stops all transmit/receive operations in initialization mode (following the setting of INIT to 1, after confirming that the system has transferred to initialization mode, use bit 3 (INISTATE) of the CAN error status register (CANES) to make the initialization setting).
- In initialization mode, the CTXD outputs a recessive level.
- The setting registers (MCNT, SYNC0, SYNC1, MASKC) can only be written to in initialization mode.

Caution Bits 6 and 7 (TxF and RxF) are read-only bits.

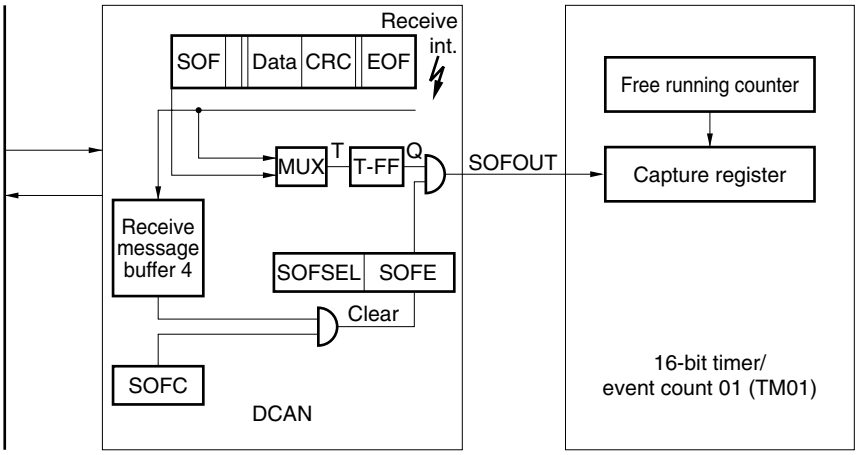
The clock supply to the DCAN is stopped in the DCAN's sleep and stop status.

The sleep status is released under following conditions:

a) The SLEEP bit is cleared.
b) There was a transition on DCAN Bus. Only when STOP = 0.
c) The sleep flag is set while the DCAN controller is transmitting/receiving.

Bit 1 (WAKE) of CANES is set to 1 under conditions b) and c). (An error interrupt (INTCE) is activated at the same time).

Figure 15-34. DCAN Support



The SOFOUT signal is used for time measurements and global time base synchronization in different DCAN nodes.

Table 15-24. SOFOUT Function Settings

SOFSEL	SOFC	SOFE	SOFOUT Function	Description of SOFOUT Operation
0	×	0	Setting prohibited	SOFOUT is unchanged
0	×	1	Time stamp mode	A toggle occurs at each receive interrupt
1	×	0	Global time mode	SOFOUT is unchanged
1	0	1		A toggle occurs at the start of each DCAN bus frame
1	1	1		A toggle occurs at the start of each DCAN bus frame When the recording of messages to receive message buffer 4 begins, SOFE is cleared.

Remarks 1. SOFC: Bit 6 (SYNC1) of synchronization control register 1

2. Aside from $\overline{\text{RESET}}$, SOFOUT is cleared to 0 by setting the INT bit.

Figure 15-35. Time Stamp Function

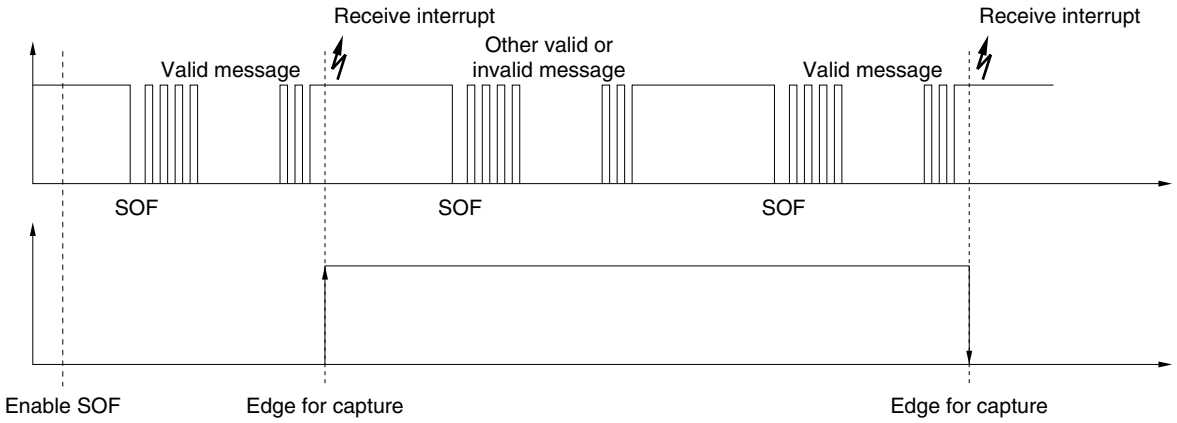


Figure 15-36. SOFOUT Toggle Function

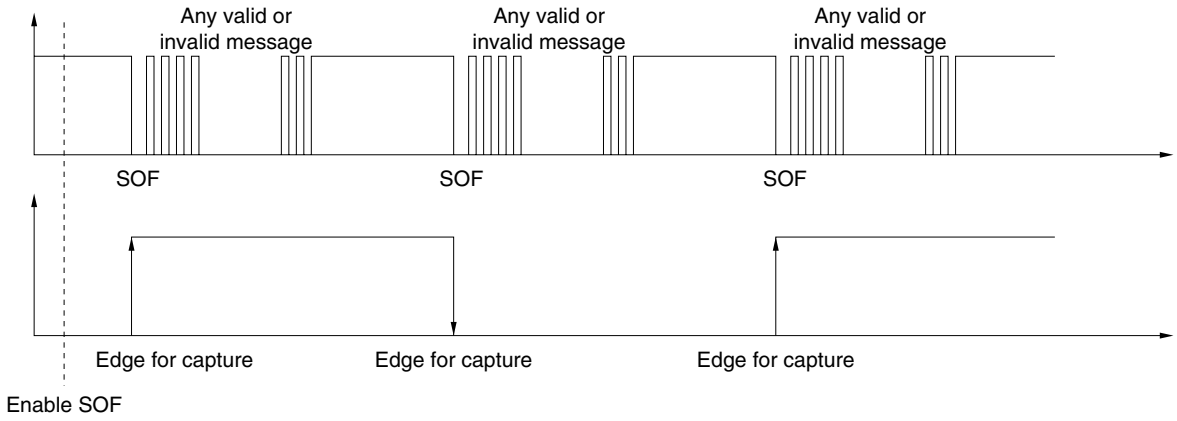
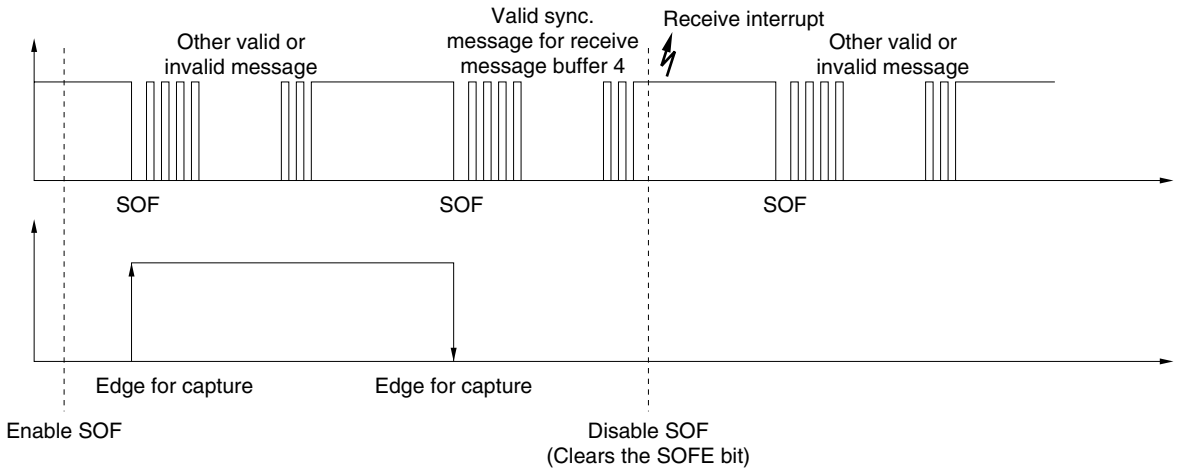


Figure 15-37. Global Time System Function



15.13.2 CAN error status register (CANES)

This register indicates the CAN error status of the transmission and reception operations.

CANES can be set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CANC to 00H.

Figure 15-38. Format of CAN Error Status Register (CANES) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
CANES	BOFF	RECS	TECS	0	INITSTATE	VALID	WAKE	OVER	FFC4H	00H	R/W

BOFF	Bus off flag
0	Transmit error counter (TEC) ≤ 255
1	TEC > 255

RECS	Receive error passive warning flag
0	Receive error counter (REC) < 96
1	REC ≥ 96 (error passive warning)

TECS	Transmit error passive warning flag
0	Transmit error counter (TEC) < 96
1	TEC ≥ 96 (error passive warning)

INITSTATE	Initialization mode status confirmation flag
0	Normal DCAN operation
1	DCAN stopped. Waiting for new configuration data (initialization data) to be set.
A change will occur if a delay is generated for INIT (bit 0 of the CAN control register (CANC)). The delay that occurs between the setting of INIT and INITSTATE changing to 1 is related to the status of the bus during operation and the time that is set when operation is stopped.	

VALID	Valid protocol detection confirmation flag
0	No detection of valid message by DCAN protocol
1	Error-free message received from DCAN bus
VALID indicates whether or not a valid protocol operation is in process on the connected DCAN bus. When a frame on the DCAN bus is correctly received in the DCAN controller, VALID is set to 1 by the completion part of the frame (the frame is not related to identifier or mask settings set in the receive message buffers of the DCAN controller).	

Figure 15-38. Format of CAN Error Status Register (CANES) (2/2)

WAKE	Wake up status confirmation flag
0	Normal operation
1	Sleep mode is released

The WAKE bit is set and an error interrupt occurs under the following conditions:

- A wake up is sent from the DCAN bus while the DCAN controller is in the sleep status.
- The SLEEP bit (bit 2 of the CAN controller register (CANC)) is set during transmit/receive operations.

Unless the WAKE bit is cleared, error interrupts will continue to occur.

OVER	Overrun flag
0	Normal operation
1	An overrun error occurred during RAM access

When the DCAN RAM access that is necessary for sorting and recording receive data or fetching transmit data is not possible, the overrun flag will be set and an error interrupt request will occur simultaneously.
An overrun will not occur in the following circumstances:

- There are too many specified messages
- Compared with the DCAN baud rate, DMA access of the RAM area is too slow.

The response of the DCAN when an overrun occurs differs depending on the situation. Make the DCAN clock that is set by bits 6 and 7 (PRM0 and PRM1) of the bit rate prescaler (BRPRS) at least 16 times the DCAN baud rate, and make the selected CPU clock (set by the DCC register) at least 8 times the baud rate.

Cautions
1. **Bits 3 and 5 to 7 (INISTATE, TECS, RECS, BOFF) are read-only bits.**
2. **The VALID, WAKE, and OVER bits are released either when written to 1, or in initialization mode (when bit 0 (INIT) of the CAN control register (CANC) is set to 1).**
3. **Writing 0 to the VALID, WAKE, and OVER bits will be ignored.**
4. **Do not bit-manipulate the CANES register.**

Remark The VALID, WAKE, and OVER bits operate in the following way during write operations.
- When they are written to 0, this operation is ignored
- When they are written to 1, they are cleared

This avoids any timing conflicts between write operations and internal operations. Because the internally set condition of the bit has priority, writing 0 to the bit by means of a bit clear request is invalid.

Table 15-25. Response of DCAN

Overrun Situation	When Detected	DCAN Operation
Cannot get transmit data.	When the DCAN controller is attempting to readout from transmit buffer the next byte data in the transmit message.	DCAN will retransmit the control frame after synchronization with the bus.
Cannot store receive data in receive message buffer	When the receive message buffer data is being stored in 6 bits of the next frame.	Data in the receive RAN does not match and no receive flags are set. The DN and MUC bits are set in the message.
Cannot receive data for ID comparison	During progression of ID comparison in 6 bits of the next frame.	Message is not received and its data is lost.

15.13.3 Transmit error counter (TEC)

The transmit error counter (TEC) is used to configure the transmit error counter of the data transmission operation.

TEC can be read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TEC to 00H.

Figure 15-39. Format of Transmit Error Counter (TEC)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TEC	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	FFC5H	00H	R

The transmit error counter reflects the error counter status of transmit errors specified in the DCAN Protocol.

15.13.4 Receive error counter (REC)

The receive error counter (REC) is used to configure the receive error counter of the data reception operation.

REC can be read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets REC to 00H.

Figure 15-40. Format of Receive Error Counter (REC)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
REC	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	FFC6H	00H	R

The receive error counter reflects the error counter status of receive errors specified in the DCAN Protocol.

15.13.5 Message count register (MCNT)

This register sets the number of receive message buffers and the RAM area of the receive message buffers handled by the DCAN controller.

MCNT can be read with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets MCNT to C0H.

Figure 15-41. Format of Message Count Register (MCNT)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MCNT	1	1	0	MCNT4	MCNT3	MCNT2	MCNT1	MCNT0	FFC7H	C0H	R/W

This register is readable at any time. Writing is only permitted when the DCAN is in initialization mode.

MCNT4	MCNT3	MCNT2	MCNT1	MCNT0	Receive message count
0	0	0	0	0	No receive message handling
0	0	0	0	1	1 receive buffer
0	0	0	1	0	2 receive buffers
0	0	0	1	1	3 receive buffers
0	0	1	0	0	4 receive buffers
0	0	1	0	1	5 receive buffers
0	0	1	1	0	6 receive buffers
0	0	1	1	1	7 receive buffers
0	1	0	0	0	8 receive buffers
0	1	0	0	1	9 receive buffers
0	1	0	1	0	10 receive buffers
0	1	0	1	1	11 receive buffers
0	1	1	0	0	12 receive buffers
0	1	1	0	1	13 receive buffers
0	1	1	1	0	14 receive buffers
0	1	1	1	1	15 receive buffers
1	0	0	0	0	16 receive buffers
Other than above					Setting prohibited; will be automatically changed to 16

Caution Be sure to set bit 5 to 0 and bits 6 and 7 to 1.

15.13.6 Bit rate prescaler (BRPRS)

The bit rate prescaler is used to set the output cycle for the DCAN.

BRPRS can be set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRPRS to 00H.

Figure 15-42. Format of Bit Rate Prescaler (BRPRS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRPRS	PRM1	PRM0	BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	FFC8H	00H	R/W

PRM1	PRM0	DCAN clock selection
0	0	f_X
0	1	$f_X/2$
1	0	$f_X/4$
1	1	$f_X/8$

BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	Output cycle
0	0	0	0	0	0	DCAN clock / 2
0	0	0	0	0	1	DCAN clock / 4
0	0	0	0	1	0	DCAN clock / 6
0	0	0	0	1	1	DCAN clock / 8
.	.	.	.	.	.	
.	.	.	.	.	.	
.	.	.	.	.	.	
1	1	1	0	1	0	DCAN clock / 118
1	1	1	0	1	1	DCAN clock / 120
1	1	1	1	0	0	DCAN clock / 122
1	1	1	1	0	1	DCAN clock / 124
1	1	1	1	1	0	DCAN clock / 126
1	1	1	1	1	1	DCAN clock / 128

Remarks 1. The clock selected by the PRM bits can be used for all DCAN operations.

2. The clock for DCAN operation is determined by BRPRS.

$$\text{Baud rate} = \frac{\text{DCAN clock (PRMn)}}{2 \times \text{BRPRS} + 2}$$

BRPRS selects the reference clock for the protocol section of the DCAN. The baud rate is determined by this setting in conjunction with the SYNC1 and SYNC0 registers.

Writing to the BRPRS register is only possible when the initialization mode is set (when bit 0 (INIT) of the CANC register is set to 1).

15.13.7 Synchronization control register n (SYNCn: n = 0, 1)

These registers specify the bit timing of the DCAN. They select the length of one data bit on the DCAN bus and the position of the bit timing sampling point.

SYNC0 and SYNC1 can be set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets SYNC0 to 18H.

$\overline{\text{RESET}}$ input sets SYNC1 to 0EH.

Figure 15-43. Format of Synchronization Control Register n (SYNCn: n = 0, 1) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SYNC0	SPT2	SPT1	SPT0	DBT4	DBT3	DBT2	DBT1	DBT0	FFC9H	18H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SYNC1	0	SOFC	SAMP	RXONLY	SJW1	SJW0	SPT4	SPT3	FFCAH	0EH	R/W

DBT4	DBT3	DBT2	DBT1	DBT0	Data bit time
0	0	1	1	1	(Output cycle of BRPRS) × 8
0	1	0	0	0	(Output cycle of BRPRS) × 9
0	1	0	0	1	(Output cycle of BRPRS) × 10
0	1	0	1	0	(Output cycle of BRPRS) × 11
0	1	0	1	1	(Output cycle of BRPRS) × 12
0	1	1	0	0	(Output cycle of BRPRS) × 13
0	1	1	0	1	(Output cycle of BRPRS) × 14
0	1	1	1	0	(Output cycle of BRPRS) × 15
0	1	1	1	1	(Output cycle of BRPRS) × 16
1	0	0	0	0	(Output cycle of BRPRS) × 17
1	0	0	0	1	(Output cycle of BRPRS) × 18
1	0	0	1	0	(Output cycle of BRPRS) × 19
1	0	0	1	1	(Output cycle of BRPRS) × 20
1	0	1	0	0	(Output cycle of BRPRS) × 21
1	0	1	0	1	(Output cycle of BRPRS) × 22
1	0	1	1	0	(Output cycle of BRPRS) × 23
1	0	1	1	1	(Output cycle of BRPRS) × 24
1	1	0	0	0	(Output cycle of BRPRS) × 25
Other than above					Setting prohibited

Remark DBT0 to DBT4 set the length of one data bit on the DCAN bus.

Figure 15-43. Format of Synchronization Control Register n (SYNCn: n = 0, 1) (2/2)

SPT4	SPT3	SPT2	SPT1	SPT0	Sampling point position
0	0	0	0	1	(Output cycle of BRPRS) × 2
0	0	0	1	0	(Output cycle of BRPRS) × 3
0	0	0	1	1	(Output cycle of BRPRS) × 4
0	0	1	0	0	(Output cycle of BRPRS) × 5
0	0	1	0	1	(Output cycle of BRPRS) × 6
0	0	1	1	0	(Output cycle of BRPRS) × 7
0	0	1	1	1	(Output cycle of BRPRS) × 8
0	1	0	0	0	(Output cycle of BRPRS) × 9
0	1	0	0	1	(Output cycle of BRPRS) × 10
0	1	0	1	0	(Output cycle of BRPRS) × 11
0	1	0	1	1	(Output cycle of BRPRS) × 12
0	1	1	0	0	(Output cycle of BRPRS) × 13
0	1	1	0	1	(Output cycle of BRPRS) × 14
0	1	1	1	0	(Output cycle of BRPRS) × 15
0	1	1	1	1	(Output cycle of BRPRS) × 16
1	0	0	0	0	(Output cycle of BRPRS) × 17
Other than above					Setting prohibited

Remark SPT0 to SPT4 select the sampling point during bit timing.

Synchronization control register 1 (SYNC1) sets the bit timing sampling point, and in addition defines the synchronization jump width. This determines the possible range of synchronization with different baud rate nodes.

SYNC1 can be set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets SYNC1 to 0EH.

Figure 15-44. Format of Synchronization Control Register 1 (SYNC1) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SYNC1	0	SOFC	SAMP	RXONLY	SJW1	SJW0	SPT4	SPT3	FFCAH	0EH	R/W

SJW1	SJW0	Synchronization jump width
0	0	(Output cycle of BRPRS) × 1
0	1	(Output cycle of BRPRS) × 2
1	0	(Output cycle of BRPRS) × 3
1	1	(Output cycle of BRPRS) × 4

RXONLY	Receive-only operation
0	Normal operation
1	Only receive operation, DCAN does not activate transmit line

Figure 15-44. Format of Synchronization Control Register 1 (SYNC1) (2/2)

SAMP	Bit sampling
0	Receive data is sampled once at the sampling point
1	Receive data is sampled 3 times at the sampling point and one is taken from these.

SOFC	Start of frame control
0	The SOFE bit is not related to the operation of the DCAN bus
1	A message for receive message buffer 4 is received, and when the SOF mode is selected, the SOFE bit is cleared.

Caution SYNC0 and SYNC1 can be read at any time. Writing to SYNC0 and SYNC1 is only possible in the initialization mode (when bit 0 of CANC is set to 1).

Remark SOFE: Bit 4 (SOFOUT operation setting flag) of the CAN control register (CANC)

(1) SJW0 and SJW1 bits (bits 2 and 3 of SYNC1)

SJW0 and SJW1 specify the synchronization jump width as prescribed in the Bosch CAN specification 2.0. This determines the possible range of synchronization with different baud rates nodes.
The limits of the DCAN protocol are as follows. Violating these limits will cause a violation of the DCAN protocol.

Limits on defining bit timing:

$2 \leq$ SPT (sampling point) ≤ 16
$7 \leq$ DBT (data bit timing) ≤ 24
$2 \leq$ (DBT − SPT) ≤ 8
SJW (allowable baud rate discrepancy range) $\leq$ (DBT − SPT − 1)

The following rules must be observed in order to conform to the DCAN protocol specification:

$1 \leq (2 \times \text{SPT} - \text{DBT}) \leq 8$

<Example> System clock: fx 6.29 MHz
CAN parameter: Baud rate 390 Kbps
Sampling point: 75%
SJW: 2BTL

BRPRS = 00H	(Clock selection = fx:	PRM = 00B)
	(DCAN Prescaler = 1/2:	BRPRS = 000000B)
SYNC0 = A7H	(CAN Bit in BTL = 8	
	[7 < (fx/DBT/Baud rate) < 26]:	DBT = 00111B)
SYNC1 = ×4H	(Sampling point 75% = 6:	SPT = 00101B)
	(SJW = 2:	SJW = 01B

× depends on the setting of:
- Number of sampling points
- Receive-only function
- Use of time stamp or global time system

(2) RXONLY bit (bit 4 of SYNC1)

The RXONLY bit is used to set the receive-only operation mode, which is used to detect the baud rate. This makes it possible to set a different baud rate without disturbing other DCAN nodes on the bus.

The receive-only operation mode has the following features.
- It never sends acknowledge, error frames, or transmit messages.
- Error counters do not count.

Bit 2 (VALID) of the CAN error status register (CANES) indicates whether the protocol has received any valid messages.

(3) SOFC bit (bit 6 of SYNC1)

SOFC is the frame control starting bit. SOFC works in conjunction with bits 3 and 4 (SOFSEL and SOFE) of the CAN control register (CANC). For detailed information refer to **15.13.1 CAN control register (CANC)**.

15.13.8 Transmit control register (TCR)

This register controls the transmission operations of the DCAN controller. It is possible to request and abort transmission of both buffers independently.

TCR can be set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TCR to 00H.

Figure 15-45. Format of Transmit Control Register (TCR)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCR	TXP	0	TXC1	TXC0	TXA1	TXA0	TXRQ1	TXRQ0	FFC1H	00H	R/W

TXP	Transmit buffer priority setting
0	Transmit buffer 0 has priority over transmit buffer 1
1	Transmit buffer 1 has priority over transmit buffer 0
When two transmit request flags (TXRQ0, TXRQ1) are set, set which transmit buffer transmits first. When there is only one transmit request, TXP is not involved. TXP is checked by the DCAN immediately before the frame is started. The order in which TXRQ0 and TXRQ1 are set before the first request frame has started on the bus does not affect the transmission priority.	

TXCn	Transmission complete flag
0	Transmission aborted/no data transmitted
1	Transmission complete/abort has no effect

TXAn	Transmit abort flag
0	Write : Standard operation Read : Abort not pending
1	Write : Abort current transmit request for transmit buffer n Read : Abort pending

TXRQn	Transmit request flag
0	Write : No transmit request for transmit buffer n Read : Transmit buffer n is released
1	Write : Transmit request for transmit buffer n Read : There is a prior transmit request in transmit buffer n

Cautions 1. Bits 4 and 5 (TXC0 and TXC1) are read-only bits.

2. The TXCn, TXAn, and TXRQn bits are cleared in initialization mode (when bit 0 (INIT) of the CAN control register (CANC) is set to 1).

3. The TXAn and TXRQn bits are not cleared even when written to 0.

4. Do not bit-manipulate the TCR register.

5. When TXRQn has been set, do not change the data in transmit buffer n.

Remark n = 0,1

(1) TXA0 and TXA1 (Bits 2 and 3 of TCR)

TXAn is the transmit abort flag. TXAn has the following functions.

- An abort can be requested by writing the TXAn bit to 1.
- The DCAN controller checks whether such request is still pending. The bit is cleared at the same time TXRQn is cleared.

This abort does not affect any rules of the DCAN protocol. A frame that has already started will continue to its end.

An abort operation can cause the DCAN controller to respond differently depending on the time it is set.

(a) If a transmit abort is set at the start of DCAN bus arbitration for transmission, the transmission complete flag (TXCn) is cleared.
(b) If a transmit abort is set in the middle of DCAN bus arbitration for transmission and arbitration is stopped, the transmission complete flag (TXCn) is cleared.
(c) If a transmit abort is set while a frame is being transmitted to the DCAN bus and transmission is erroneously completed, the transmission complete flag (TXCn) is cleared.
(d) If a transmit abort is set while a frame is being transmitted to the DCAN bus but transmission is completed error-free, the transmission complete flag (TXCn) is set.

In all of the above cases, the TXRQn and TXAn bits are cleared at the end of the transmit abort operation. The transmission complete flag is reset at the end of each transmitted frame or transmit abort.

(2) TXRQ0 and TXRQ1 (bits 1 and 2 of TRC)

TXRQn is the transmit request flag. TXRQn has the following functions.

- Requesting the transmit buffer to transmit
- Indicating whether the buffer is in a usable state or not

Remarks 1. TXRQn is cleared after transmission is complete.
2. Errors that occur during transmission do not affect the transmit request status.
3. When an error occurs, the DCAN controller automatically retransmits the data.

15.13.9 Receive message register (RMES)

This register reflects the status of the DN bits in receive message buffers 0 to 7.

RMES can be read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets RMES to 00H.

Figure 15-46. Format of Receive Message Register (RMES)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
RMES	DN7	DN6	DN5	DN4	DN3	DN2	DN1	DN0	FFC2H	00H	R

DNn	New data bit for message n (n = 0...7)
0	No message received regarding message n, CPU has cleared DN bit in message n
1	New data received in message n

This register is read only and it is cleared in the initialization mode (when bit 0 (INIT) of the CAN control register (CANC) is set to 1).

DN0 has no meaning when a mask identifier has been set in receive message buffer 0 by the mask control register (MASKC).

DN2 has no meaning when a mask identifier has been set in receive message buffer 2 by the mask control register (MASKC).

15.13.10 Mask control register (MASKC)

This register sets the mask function operation of the receive message buffers provided in the DCAN controller.

When a bit that has been mask-specified in the identifier is compared with a message received on the DCAN bus, the bit is ignored.

Message buffers that have been specified as mask identifiers cannot be used as usual message storage buffers.

MASKC can be set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets MASKC to 00H.

Figure 15-47. Format of Mask Control Register (MASKC)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MASKC	0	0	0	0	0	GLOBAL	MSK1	MSK0	FFCBH	00H	R/W

GLOBAL	Global mask setting flag
0	Normal mask operation
1	Global mask operation (Operates as a mask for identifiers of all message buffers allocated in higher addresses than the message buffers that operate as masks)

MSK1	Mask 1 setting flag
0	Receive message buffers 2 and 3 in normal operation
1	Receive message buffer 2 is mask for receive message buffer 3

MSK0	Mask 0 setting flag
0	Receive message buffers 0 and 1 in normal operation
1	Receive message buffer 0 is mask for receive message buffer 1

This register is readable at any time. Writing is only permitted when the DCAN is in initialization mode.

When global mask is operating, masks are applied to both standard and extended frames. Table 15-26 shows the mask operation statuses that can be set by the mask control register (MASKC).

Table 15-26. Mask Operation Buffers

GLOBAL	MSK1	MSK0	Receive Message Buffer					Operation
			0	1	2	3	4 to 15	
×	0	0	Compare ID	Compare ID	Compare ID	Compare ID	Compare ID	Standard
0	0	1	Mask 0	Compare ID & mask 0	Compare ID	Compare ID	Compare ID	Receive message buffer 0 masks receive message buffer 1
0	1	0	Compare ID	Compare ID	Mask 1	Compare ID & mask 1	Compare ID	Receive message buffer 2 masks receive message buffer 3
0	1	1	Mask 0	Compare ID & mask 0	Mask 1	Compare ID & mask 1	Compare ID	Receive message buffer 0 masks receive message buffer 1 Receive message buffer 2 masks receive message buffer 3
1	0	1	Mask 0	Compare ID & mask 0	Compare ID & mask 0	Compare ID & mask 0	Compare ID & mask 0	Receive message buffer 0 masks receive message buffers 1 to 15
1	1	0	Compare ID	Compare ID	Mask 1	Compare ID & mask 1	Compare ID & mask 1	Receive message buffer 2 masks receive message buffers 3 to 15
1	1	1	Mask 0	Compare ID & mask 0	Mask 1	Compare ID & mask 1	Compare ID & mask 1	Receive message buffer 0 masks receive message buffer 1 Receive message buffer 2 masks receive message buffers 3 to 15

Remark Compare ID: Compares the ID of the message received from on the bus with its own ID.

Mask 0,1: Operate as mask identifiers.

Compare ID and mask 0,1: When comparing the ID of the message received from on the bus with its own ID, the bit that is specified by the mask is ignored and comparison proceeds.

- Priority of receive message buffers during comparison
  It is possible that more than one receive message buffer is configured to receive an incoming message. In this case, the priority of the 16 receive buffers are as follows:

  - Receive message buffers with higher addresses have higher priority.
  - A masked receive message buffer has a lower priority than non-masked receive buffers.

15.13.11 Redefinition control register (REDEF)

This register is used to redefine the identifier of the DCAN controller's receive messages. REDEF is set when the identifier of a receive message buffer is changed while the DCAN controller is operating. This makes it possible to change the identifier of a certain receive message buffer without affecting other buffers.

REDEF can be set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets REDEF to 00H.

Figure 15-48. Format of Redefinition Control Register (REDEF)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
REDEF	DEF	0	0	0	SEL3	SEL2	SEL1	SEL0	FFC3H	00H	R/W

DEF	Redefine enable flag
0	Normal operation
1	The receive operation of the selected message buffer is invalid. This flag is automatically set to 1 when the identifier of a receive message buffer is changed at a time other than during the initialization phase.

SEL3	SEL2	SEL1	SEL0	Selection of receive message buffer to be redefined (n = 0...15)
0	0	0	0	Receive message buffer 0 is selected for redefinition
0	0	0	1	Receive message buffer 1 is selected for redefinition
0	0	1	0	Receive message buffer 2 is selected for redefinition
0	0	1	1	Receive message buffer 3 is selected for redefinition
0	1	0	0	Receive message buffer 4 is selected for redefinition
0	1	0	1	Receive message buffer 5 is selected for redefinition
0	1	1	0	Receive message buffer 6 is selected for redefinition
0	1	1	1	Receive message buffer 7 is selected for redefinition
1	0	0	0	Receive message buffer 8 is selected for redefinition
1	0	0	1	Receive message buffer 9 is selected for redefinition
1	0	1	0	Receive message buffer 10 is selected for redefinition
1	0	1	1	Receive message buffer 11 is selected for redefinition
1	1	0	0	Receive message buffer 12 is selected for redefinition
1	1	0	1	Receive message buffer 13 is selected for redefinition
1	1	1	0	Receive message buffer 14 is selected for redefinition
1	1	1	1	Receive message buffer 15 is selected for redefinition

Cautions 1. Do not change DEF and SEL at the same time. Change SEL after DEF is cleared. Then either write to SEL and DEF, or change them with a 1-bit manipulation instruction.

2. Keep SEL set to same value even if clearing DEF.

Remark The DEF bit is cleared to 0 in the initialization mode.

Receive operations are stopped for receive message buffers that have been selected for redefinition.

Caution When changing a receive identifier, be sure to check the data update confirmation bit (DN) of that buffer with the redefinition control register (REDEF) after the receive operation of that buffer has been stopped. When the DN bit is set, read out the data of the corresponding receive message buffer. Then, change the identifier, and restart the receive operation using REDEF. If a receive message buffer that is in the middle of a message recording operation has been selected for redefinition by REDEF, stop the receive operation after the message recording operation is complete.

At this time, if the identifier is changed and the receive operation of that buffer is restarted without reading out from the buffer, because the DN bit has been set even though correct data has not been recorded, it will be assumed the data is correct.

15.14 Interrupt Function

15.14.1 Interrupt vectors

The DCAN controller has the following interrupt sources.

Table 15-27. Interrupt Sources

Function	Source	Interrupt Request Flag	Interrupt Request Signal
Error	Error counter Overrun error Wake up	CEIF	INTCE
Receive	Received frame is valid	CRIF	INTCR
Transmit buffer n	TXRQn is cleared	CTIF	INTCT

Remark n = 0, 1

15.14.2 Transmit interrupt

When TXRQ0 and TXRQ1 (transmit request flags) are cleared, the DCAN controller generates a transmit interrupt request signal (INTCT).

If TXRQ0 and TXRQ1 have been cleared, it is possible to write new data to the transmit buffer. TXRQ0 and TXRQ1 are cleared following a normal transmit operation or a transmit abort.

15.14.3 Receive interrupt

The DCAN controller generates a receive interrupt request signal (INTCR) in the following cases.

- When the DCAN controller has received a receive message that conforms to the DCAN protocol.
- When the memory access engine has saved data for a receive message buffer that conforms to the receive message identifier.
- When the receive interrupt enable bit (ENI) of a receive message buffer has been set.

15.14.4 Error interrupt

When the following flags in the CAN control register (CANES) have been set to 1, the DCAN controller generates an error interrupt request signal (INTCE).

- BOFF (Bus off flag)
- TECS (Transmit error passive warning flag)
- RECS (Receive error passive warning flag)
- OVER (Overrun flag)
- WAKE (Wake up status confirmation flag)

Remark For the DCAN controller to restart operation when woken up, a certain number of clock cycles are required. In the period up until the DCAN controller restarts operation following a wake up, the error interrupt request signal is in the active state (INTCE).

15.15 Standby Function

15.15.1 HALT mode

The HALT mode can be used in conjunction with the DCAN sleep mode.

15.15.2 STOP mode

The DCAN controller stops operating when its clock supply stops. If the clock is not synchronized with the operation of the bus, DCAN bus operation may malfunction.

When the clock supply restarts, the DCAN will change to overrun the status.

The STOP mode can be used in conjunction with the DCAN sleep mode.

15.15.3 DCAN sleep mode

The DCAN supports the sleep mode prescribed in the Bosch Co.-determined CAN specifications.

Request the DCAN controller to enter the sleep mode by setting bit 1 (STOP) of the CAN control register (CANC) to 0 and bit 2 (SLEEP) of CANC to 1.

If the DCAN controller is in the middle of a transmit/receive operation when this request is set, the request is cancelled and WAKE (bit 1 of the CAN error status register (CANES)) becomes 1.

If a request has been cancelled, reset the DCAN sleep mode request after the WAKE bit is cleared. Be sure to put the CPU in either the HALT or STOP mode after the DCAN has moved into the sleep mode status.

The DCAN sleep mode can be released by either changing the DCAN bus line status, or by clearing the STOP and SLEEP bits.

★ **Caution The DCAN sleep mode is released when the DCAN sleep mode is set and the STOP instruction is executed.**

(a) If the DCAN error interrupt request (INTCE) is not masked, the DCAN sleep mode is released immediately after the STOP instruction is executed. The STOP mode is also released after the oscillation stabilization time has elapsed.

(b) If the DCAN error interrupt request (INTCE) is masked, the DCAN moves into the STOP mode. Even if the STOP mode was released by another source, the DCAN sleep mode was already released.

(c) Mask the DCAN error interrupt request (INTCE) when using the DCAN sleep mode or STOP mode.

Because the DCAN sleep mode is released immediately after the STOP instruction is executed, the WAKE bit of the CAN error status register (CANES) is also set at that time. Therefore, even if the DCAN bus becomes active in this status, a DCAN error interrupt request (INTCE) is not generated.

15.15.4 DCAN stop mode

Request the DCAN controller to enter the stop mode by setting bits 1 and 2 (STOP and SLEEP) of the CAN control register (CANC) to 1.

If the DCAN controller is in the middle of a transmit/receive operation when this request is set, the request is cancelled and WAKE (bit 1 of the CAN error status register (CANES)) becomes 1.

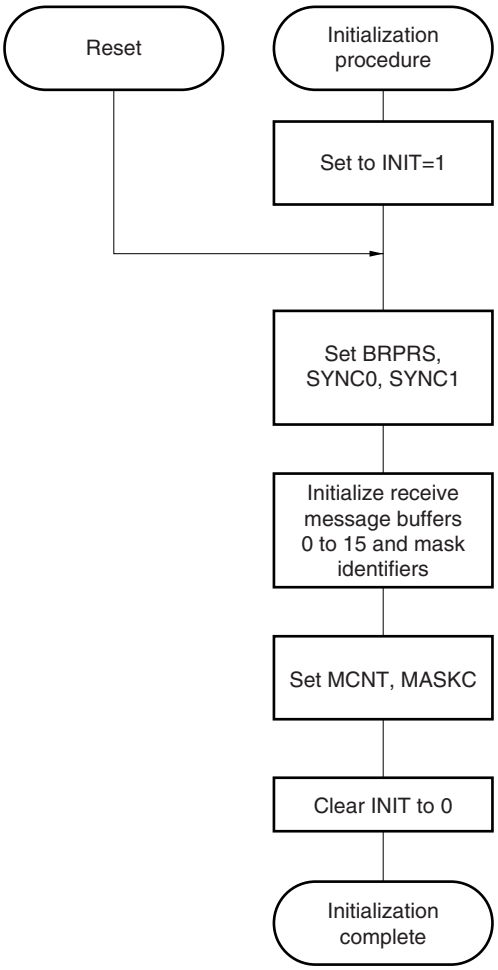
If a request has been cancelled, reset the DCAN stop mode request after the WAKE bit is cleared. Be sure to put the CPU in either the HALT or STOP mode after the DCAN has moved into the stop mode status.

The DCAN stop mode can be released by clearing the STOP and SLEEP bits.

15.16 Description of Functions by Flowcharts

15.16.1 Initialization procedure

Figure 15-49. Initialization Procedure

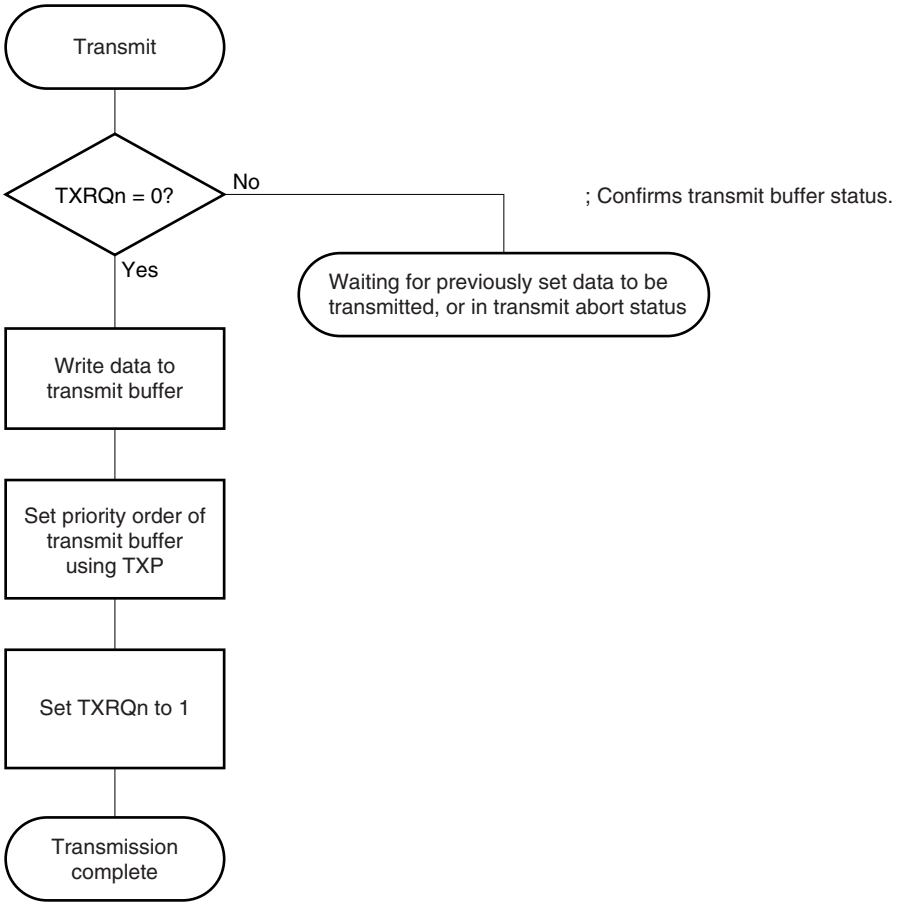


Caution BRPRS, SYNC0, SYNC1, MCNT, and MASKC can only be written to when INIT is set to 1.

Remark INIT: Bit 0 of the CAN control register (CANC)
BRPRS: Bit rate prescaler
SYNC0,
SYNC1: Initialization control register
MCNT: Message count register
MASKC: Mask control register

15.16.2 Transmit procedure

Figure 15-50. Transmit Procedure

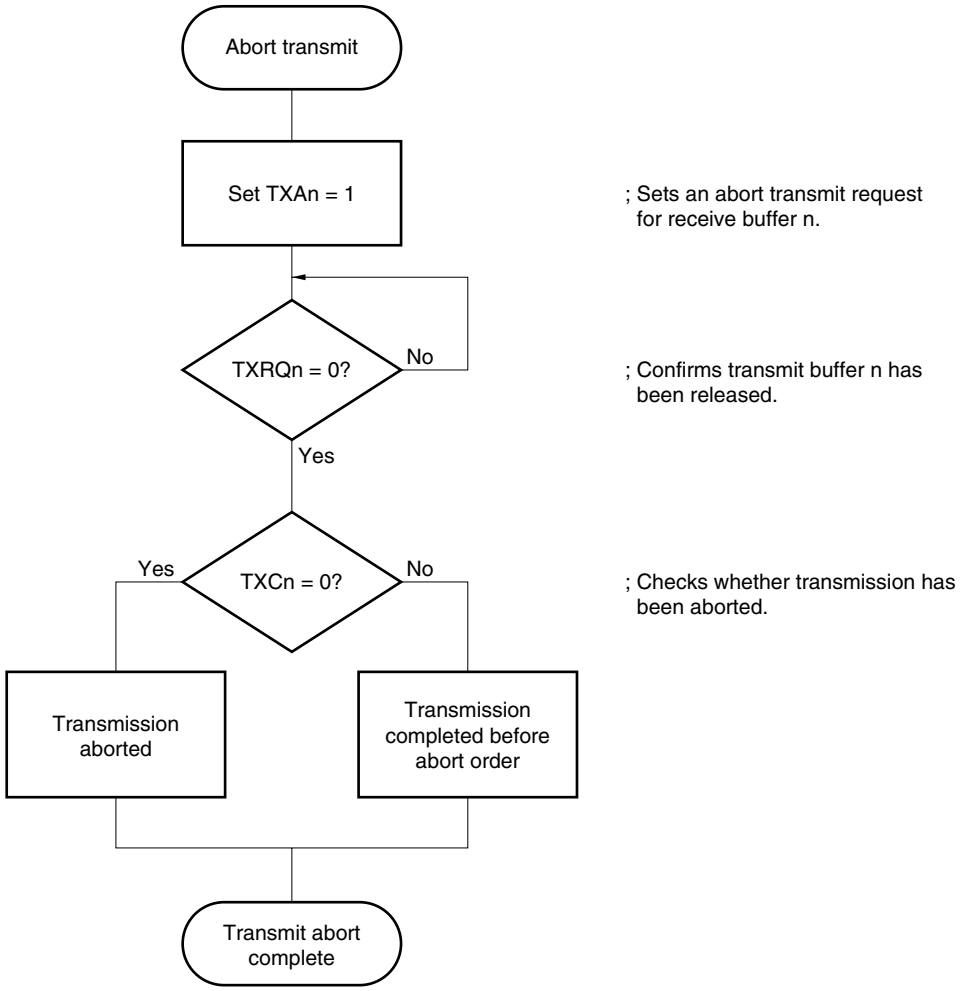


n = 0, 1

Remark TXRQn: Bits 0 and 1 of the transmit control register (TCR)
TXP: Bit 7 of the transmit control register (TCR)

15.16.3 Transmit abort procedure

Figure 15-51. Transmit Abort Procedure

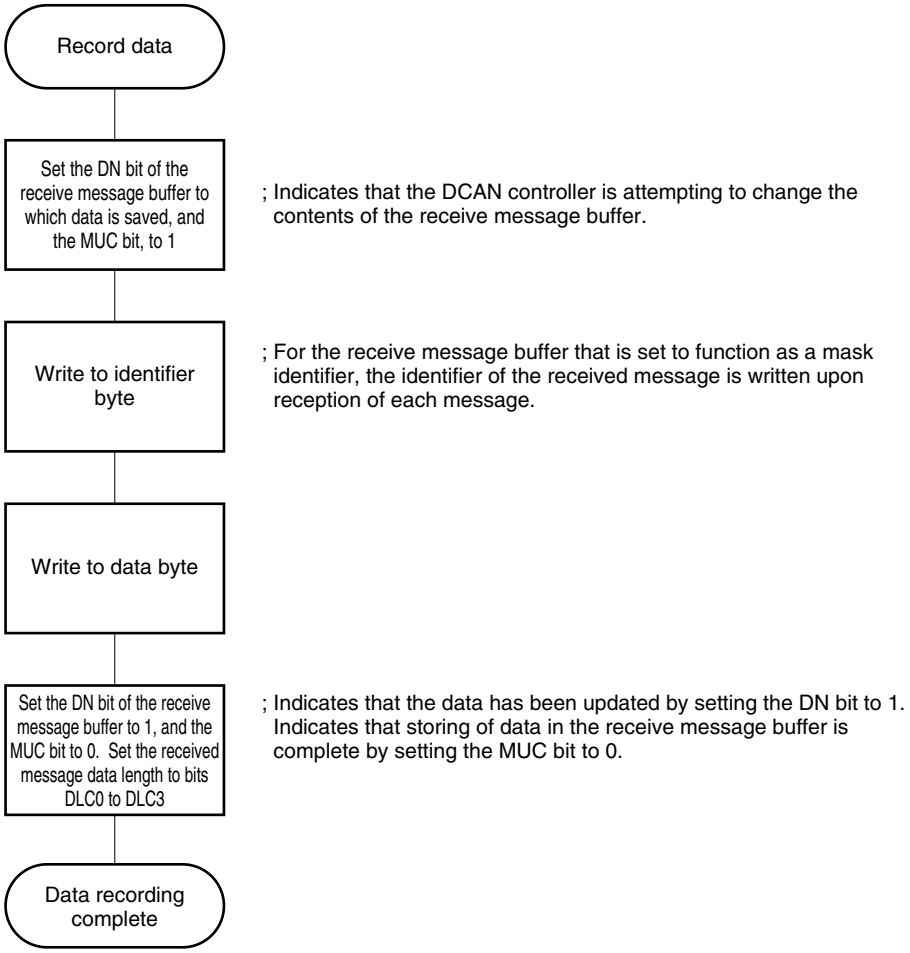


n = 0, 1

Remark TXAn: Bits 2 and 3 of the transmit control register (TCR)
TXRQn: Bits 0 and 1 of the transmit control register (TCR)
TXCn: Bits 4 and 5 of the transmit control register (TCR)

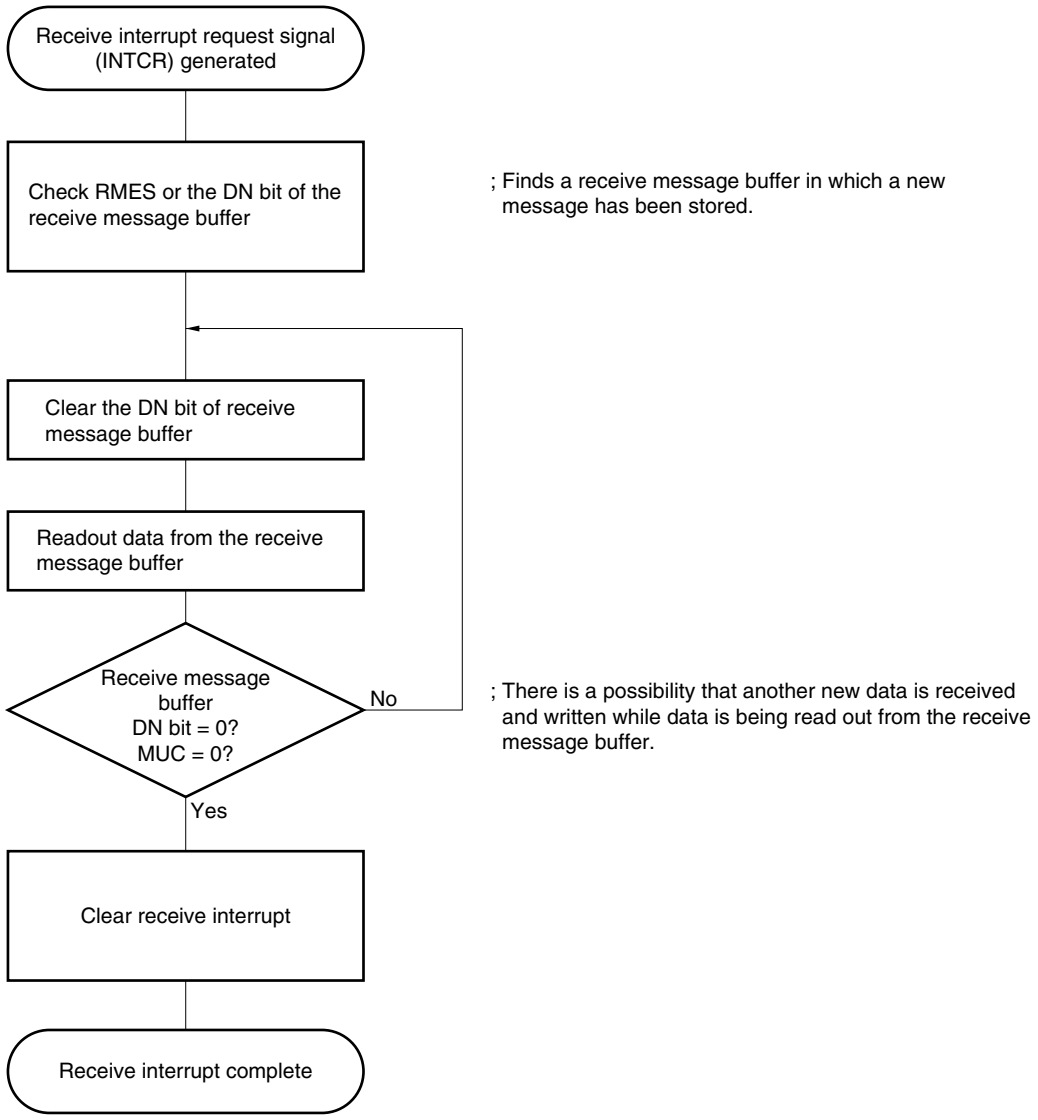
15.16.4 Receive processing by DCAN controller

Figure 15-52. Processing for DN and MUC Bits of Receive Message Buffer by DCAN Controller during Receive Operation



15.16.5 Receive procedure (when receive interrupt request signal (INTCR) used)

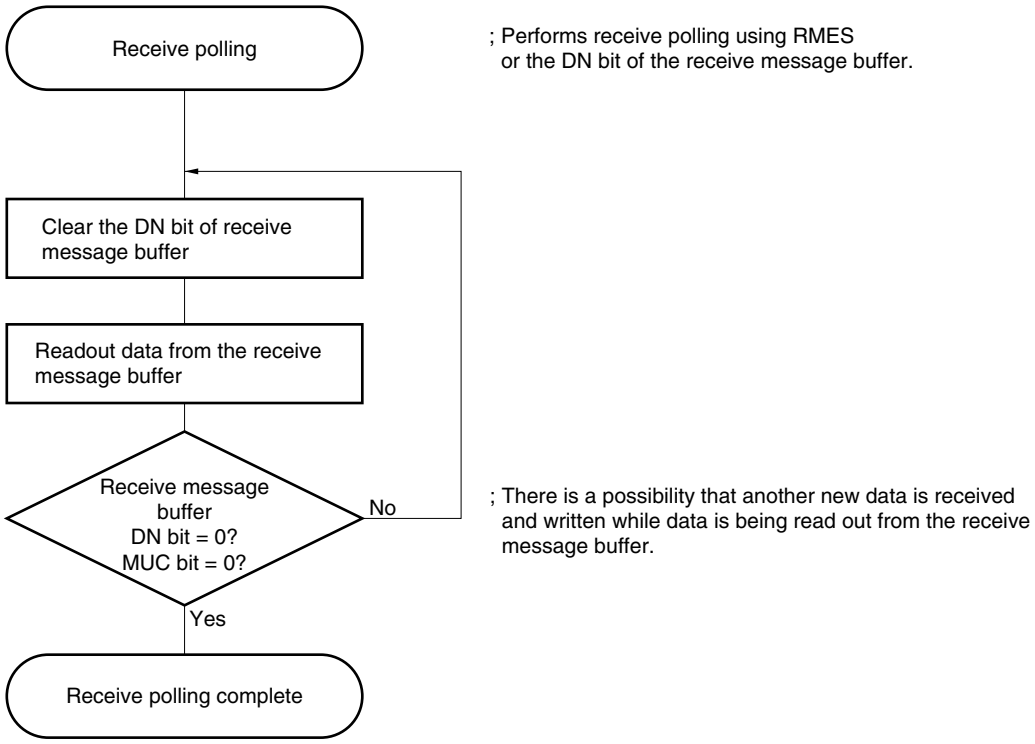
Figure 15-53. Receive Procedure (When Receive Interrupt Request Signal (INTCR) Used)



Remark RMES: Receive message register

15.16.6 Receive procedure (when receive interrupt request signal (INTCR) not used)

Figure 15-54. Receive Procedure (When Receive Interrupt Request Signal (INTCR) Not Used)



Remark RMES: Receive message register

★ CHAPTER 16 IEBus CONTROLLER (μPD780702Y, 78F0701Y ONLY)

The μPD780701Y Subseries (μPD780702Y, 78F0701Y) incorporates an IEBus controller.

IEBus (Inter Equipment Bus™) is a small-scale digital data transfer system that transfers data between units. To implement IEBus by using the μPD780701Y Subseries, an external IEBus driver and receiver are necessary because they are not provided.

The internal IEBus controller of the μPD780701Y Subseries is of negative logic.

Caution The μPD780701Y does not incorporate an IEBus controller. Moreover, because the IEBus controller pins in the μPD78F0701Y can also be used for the DCAN controller, it is not possible to use the IEBus controller at the same time the DCAN controller is being used.

16.1 IEBus Controller Functions

16.1.1 Communication protocol of IEBus

The communication protocol of the IEBus is as follows.

(1) Multitask mode

All the units connected to the IEBus can transfer data to the other units.

(2) Broadcast 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 (the μPD780701Y Subseries does not support modes 0 and 2 for the effective transfer rate).

- Mode 1: Approx. 18 Kbps (when fx = 6.29 MHz)

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 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

Although the IEBus has three communication modes each having a different transfer rate, the μPD780701Y Subseries supports only communication mode 1. The transfer rate and the maximum number of transfer bytes in one communication frame in communication mode 1 are as shown in Table 16-1.

Table 16-1. Transfer Rate and Maximum Number of Transfer Bytes in Communication Mode 1

Communication Mode	Maximum Number of Transfer Bytes (Bytes/Frame)	Effective Transfer Rate (Kbps)[Note]
1	32	Approx. 18

Note The effective transfer rate when the maximum number of transfer bytes is transmitted (when f_X = 6.29 MHz).

Select the communication mode (mode 1) 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 executed correctly.

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 multiple slave units exist, the slave units do not return an acknowledge signal during communication.

Whether broadcast communication or normal communication is to be executed is selected by the broadcast bit (for this bit, refer to **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, refer to **16.1.6 (4) Slave address field**).

(1) Group-unit broadcast communication

Broadcast communication is performed with 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 with all the units, regardless of the value of the group number.

16.1.6 Transfer format of IEBus

Caution The logic on the IEBus I/O pin of the μPD780701Y Subseries and the logic of the IEBus protocol (data on the IEBus) are inverted values. The following describes the case of the IEBus protocol.
<Example: Start bit>
• μPD780702Y: High level
• IEBus protocol: Low level

Figure 16-1 shows the transfer signal format of the IEBus.

Figure 16-1. IEBus Transfer Signal Format

	Header		Master address field		Slave address field			Control field			Telegraph length field			Data field					
Frame format	Start bit	Broad-cast bit	Master address bit	P	Slave address bit	P	A	Control bit	P	A	Tele-graph length bit	P	A	Data bit	P	A	Data bit	P	A

Remarks 1. P: Parity bit, A: $\overline{\text{ACK}}$/NACK bit

2. The master station 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 low-level signal (start bit) 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, refer to **16.1.6 (4) Slave address field**).

Because two or more slave units exist in the case of broadcast communication, the acknowledge bit in each field subsequent to the master address field is not returned.

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 (ALLRQ) of IEBus control register 0 (BCR0) 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 is used to make a judgment of arbitration.

The data output in the master address field is compared with the data on the bus each time one bit is output. If the master address output in 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.

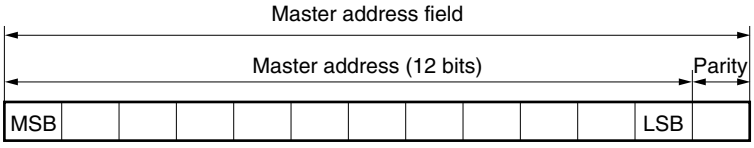
Because the IEBus is configured of wired AND, the unit having the minimum master address of the units participating in arbitration (arbitration masters) wins in arbitration.

After a 12-bit master address has been output, only one unit remains in the transmission status as the master unit.

Next, this master unit outputs a parity bit, determines the master address of other units, and starts outputting a slave address field.

If one unit occupies the bus as the master, the address set by the IEBus unit address register (UAR) 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 acknowledge signal from the slave unit to confirm that the slave unit exists on the bus. When the master has detected the acknowledge signal, it starts outputting the control field. During broadcast communication, however, the master does not detect the acknowledge bit but starts outputting the control field.

The slave unit outputs the acknowledge 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 does not output the acknowledge signal if the parities are odd. At this time, the master unit is in the standby (monitor) status, and communication ends.

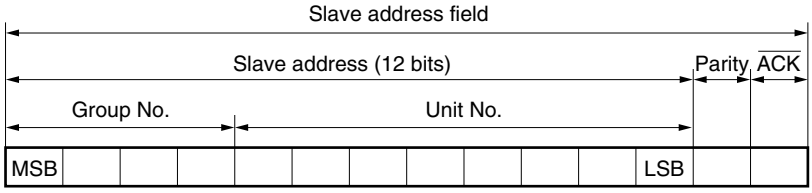
During broadcast communication, the slave address is used to identify group-unit broadcast or all-unit broadcast, as follows.

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 broadcast 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 IEBus slave address register (SAR) 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 acknowledge 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 does not output the acknowledge signal, and returns to the standby (monitor) status.

The master unit starts outputting the telegraph length field after confirming the acknowledge signal.

If the master cannot confirm the acknowledge signal, the master unit enters the standby status, and communication ends. During broadcast communication, however, the master unit does not confirm the acknowledge signal, and starts outputting the telegraph length field.

Table 16-2 shows the contents of the control bits.

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

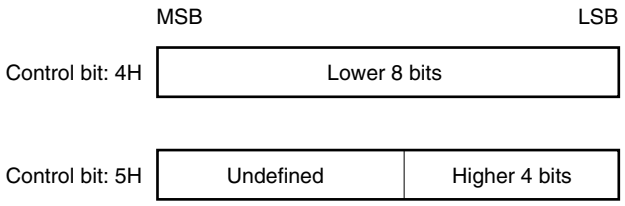
Notes 1. The telegraph length bit of the telegraph length field and data transfer direction of the data field change as follows depending on the value of bit 3 (MSB).

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 (refer to **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 does not output the acknowledge bit.

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	1	0	1	Read lock address (higher 4 bits)

Moreover, units for which lock is not set by the master unit reject acknowledgment and do not output an acknowledge bit when the control data shown in Table 16-4 is received.

Table 16-4. Control Field for Unlocked Slave Unit

Bit 3	Bit 2	Bit 1	Bit 0	Function
0	1	0	0	Read lock address (lower 8 bits)
0	1	0	1	Read lock address (higher 4 bits)

If one unit occupies the bus as the master, the value set to the IEBus control data register (CDR) is output.

Figure 16-4. Control Field

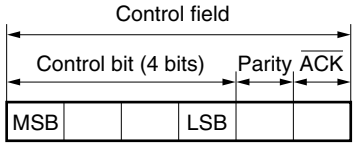


Table 16-5. Acknowledge Signal Output Conditions of Control Field

(a) If received control data is AH, BH, EH, or FH

Communication Target (SLVRQ) Slave Specification = 1 No Specification = 0	Lock Status (LOCK) Lock = 1 Unlock = 0	Master Unit Identification (Match with PAR) Lock Request Unit = 1 Other = 0	Slave Transmission Enable (ENSLVTX)	Slave Reception Enable (ENSLVRX)	Received Control Data			
					AH	BH	EH	FH
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 Target (SLVRQ) Slave Specification = 1 No Specification = 0	Lock Status (LOCK) Lock = 1 Unlock = 0	Master Unit Identification (Match with PAR) Lock Request Unit = 1 Other = 0	Slave Transmission Enable (ENSLVTX)	Slave Reception Enable (ENSLVRX)	Received Control Data					
					0H	3H	4H	5H	6H	7H
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, × ($\overline{\text{ACK}}$ is not returned) is unconditionally assumed.

Remarks 1. √: $\overline{\text{ACK}}$ is returned.
×: $\overline{\text{ACK}}$ is not returned.

2. ENSLVTX: Bit 4 of IEBus control register 0 (BCR0)
ENSLVRX: Bit 3 of IEBus control register 0 (BCR0)
LOCK: Bit 2 of the IEBus unit status register (USR)
SLVRQ: Bit 6 of the IEBus unit status register (USR)
PAR: IEBus partner address register

(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

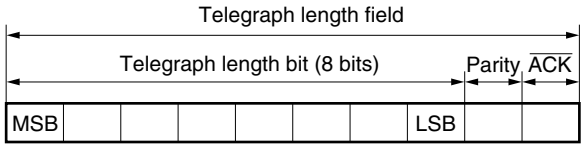


Table 16-6. Contents of Telegraph Length Bit

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. When the slave unit detects that the parity is even, it outputs the acknowledge signal, and starts outputting the data field. During broadcast communication, however, the slave unit does not output the acknowledge signal.

If the parity is odd, the slave unit judges that the telegraph length bit has not been correctly received, does not output the acknowledge 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 acknowledge signal.

If the parity bit is odd, the master unit judges that the telegraph length bit has not been correctly received, does not output the acknowledge signal, and returns to the standby (monitor) status. At this time, the slave unit also returns to the standby status, and communication ends.

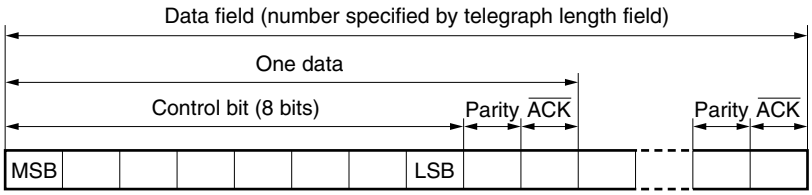
(7) Data field

This is data output by the transmission side.

The master unit transmits to or receives data 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 signal is ignored.

The operation differs as follows depending on whether the master transmits or receives data.

(a) When master transmits data

When the master unit 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 IEBus data register (DR) when the slave unit has received the data bit and parity bit, the slave unit outputs an acknowledge signal. If the parity is odd or if the receive data is stored in the IEBus data register (DR), the slave unit rejects receiving the data, and does not output the acknowledge signal.

If the slave unit does not output the acknowledge signal, the master unit transmits the same data again. This operation continues until the master detects the acknowledge 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 acknowledge signal, the master unit transmits the next data.

During broadcast communication, the slave unit does not output the acknowledge 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 does not output the acknowledge 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 returns the acknowledge 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 that 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 bit is 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 that 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 ($\overline{\text{ACK}}$).
- 1: Indicates that the transmit data is not recognized (NACK).

During broadcast communication, however, the contents of the acknowledge bit are ignored.

(a) Acknowledge bit at end of slave address field

The acknowledge bit at the end of the slave address field serves as NACK 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) Acknowledge bit at end of control field

The acknowledge bit at the end of the control field serves as NACK 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 (ENSLVRX) is not set (1) (refer to **16.4.2 (1) IEBus control register 0 (BCR0)**)
- If the control bit indicates reading of data (3H or 7H) when the slave transmission enable flag (ENSLVTX) is not set (1) (refer to **16.4.2 (1) IEBus control register 0 (BCR0)**)
- If a unit other than the one that set locking requests control bits 3H, 6H, 7H, AH, BH, EH, or FH when locking is set
- If the control bit indicates reading of a lock address (4H or 5H) even when locking is not set
- If a timing error occurs
- If the control bit is undefined

Cautions 1. Even when the slave transmission enable flag (ENSLVTX) is not set (1), $\overline{\text{ACK}}$ may be returned if control data is received (refer to Table 16-5).

2. Even when the slave reception enable flag (ENSLVRX) is not set (1), NACK is always returned by the acknowledge bit in the control field if data/command writing control data is received.

Slave reception can be disabled (communication stopped) by the ENSLVRX flag only in the case of individual communication. In the case of broadcast communication, communication is maintained and the data request interrupt (INTIE1) or IEBus end interrupt (INTIE2) is generated.

(c) Acknowledge bit at end of telegraph length field

The acknowledge bit at the end of the telegraph length field serves as NACK 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) Acknowledge bit at end of data field

The acknowledge bit at the end of the data field serves as NACK 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 receive data is stored in the IEBus data register (DR) 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 acknowledge bit ($\overline{ACK}$) 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]	Meaning
0	Transmit data is not written in IEBus data register (DR)
1	Transmit data is written in IEBus data register (DR)

Bit 1[Note 2]	Meaning
0	Receive data is not stored in IEBus data register (DR)
1	Receive data is stored in IEBus data register (DR)

Bit 2	Meaning
0	Unit is not locked
1	Unit is locked

Bit 3	Meaning
0	Fixed to 0

Bit 4[Note 3]	Meaning
0	Slave transmission is stopped
1	Slave transmission is ready

Bit 5	Meaning
0	Fixed to 0

Bit 7	Bit 6	Meaning	
0	0	Mode 0	Indicates the highest mode supported by 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.

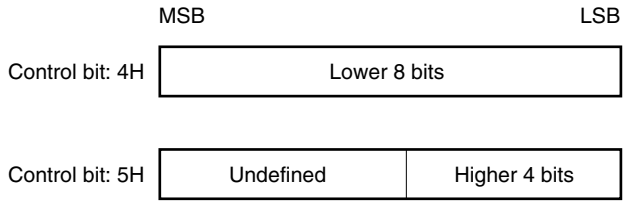
3. When the μPD780701Y Subseries serves as a slave unit, this bit corresponds to the status indicated by bit 4 (ENSLVTX) of IEBus control register 0 (BCR0).

4. When the μPD780701Y Subseries serves as a slave unit, bits 7 and 6 are fixed to '0' and '1' (mode 1), respectively.

(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 (either individual or 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 related to locking (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.

The locking and unlocking conditions are shown below.

(c) Lock setting conditions

Control Data	Broadcast Communication		Individual Communication	
	Communication End	Frame End	Communication End	Frame End
3H, 6H[Note]			Not locked	Lock set
AH, BH	Not locked	Not locked	Not locked	Lock set
0H, 4H, 5H, EH, FH	Not locked	Not locked	Not locked	Not locked

(d) Lock 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

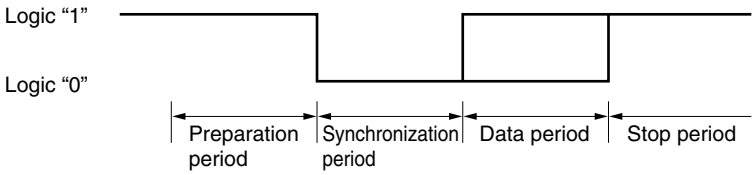
Note The frame end of control data 6H (slave status read/unlock) occurs when the parity in the data field is odd, and when the acknowledge signal from the IEBus unit is repeated up to the maximum number of transfer bytes without being output.

16.1.8 Bit format

Caution The logic on the IEBus I/O pin of the μPD780701Y Subseries and the logic of the IEBus protocol (data on the IEBus) are inverted values. The following describes the case of the IEBus protocol.
<Example: Start bit>
• μPD780702Y: High level
• IEBus protocol: Low level

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 communicating. 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 Simple IEBus Controller

The μPD780701Y Subseries has a newly developed IEBus controller. The functions of this IEBus controller are limited compared with the conventional IEBus interface functions of the existing models (provided in the 78K/0 Series).

Table 16-7 compares the conventional IEBus interface functions of the existing models with the simple IEBus interface functions of the μPD780701Y Subseries.

Table 16-7. Comparison of Existing and Simple IEBus Interface Functions

Item	Conventional Functions (IEBus of 78K/0)	Simple Version (IEBus Incorporated in μPD780701Y Subseries)
Communication mode	Modes 0, 1, and 2	Fixed to mode 1
Internal system clock	f_X = 6.0 (6.29) MHz	f_X = 6.29 MHz[Note 1]
Internal buffer size	Transmit buffer: 33 bytes (FIFO) Receive buffer: 40 bytes (FIFO) Up to 4 frames can be received.	Transmit/receive data register
CPU processing	Communication start preprocessing (data setting) Setting and management of each communication status Writing data to transmit buffer Reading data from receive buffer	Communication start preprocessing (data setting) Setting and management of each communication status 1-byte data write processing 1-byte data read processing Management of transmission such as slave status Management of multiple frames, master request reprocessing
Hardware processing	Bit processing (modulation/demodulation, error detection) Field processing (generation/management) Arbitration result detection Parity processing (generation/error detection) Automatic return of $\overline{ACK}$/NACK Automatic data transmission reprocessing Automatic master reprocessing[Note 2] Transmission processing such as automatic slave status transmission Multiple-frame reception processing	Bit processing (modulation/demodulation, error detection) Field processing (generation/management) Arbitration result detection Parity processing (generation/error detection) Automatic return of $\overline{ACK}$/NACK Automatic data transmission reprocessing

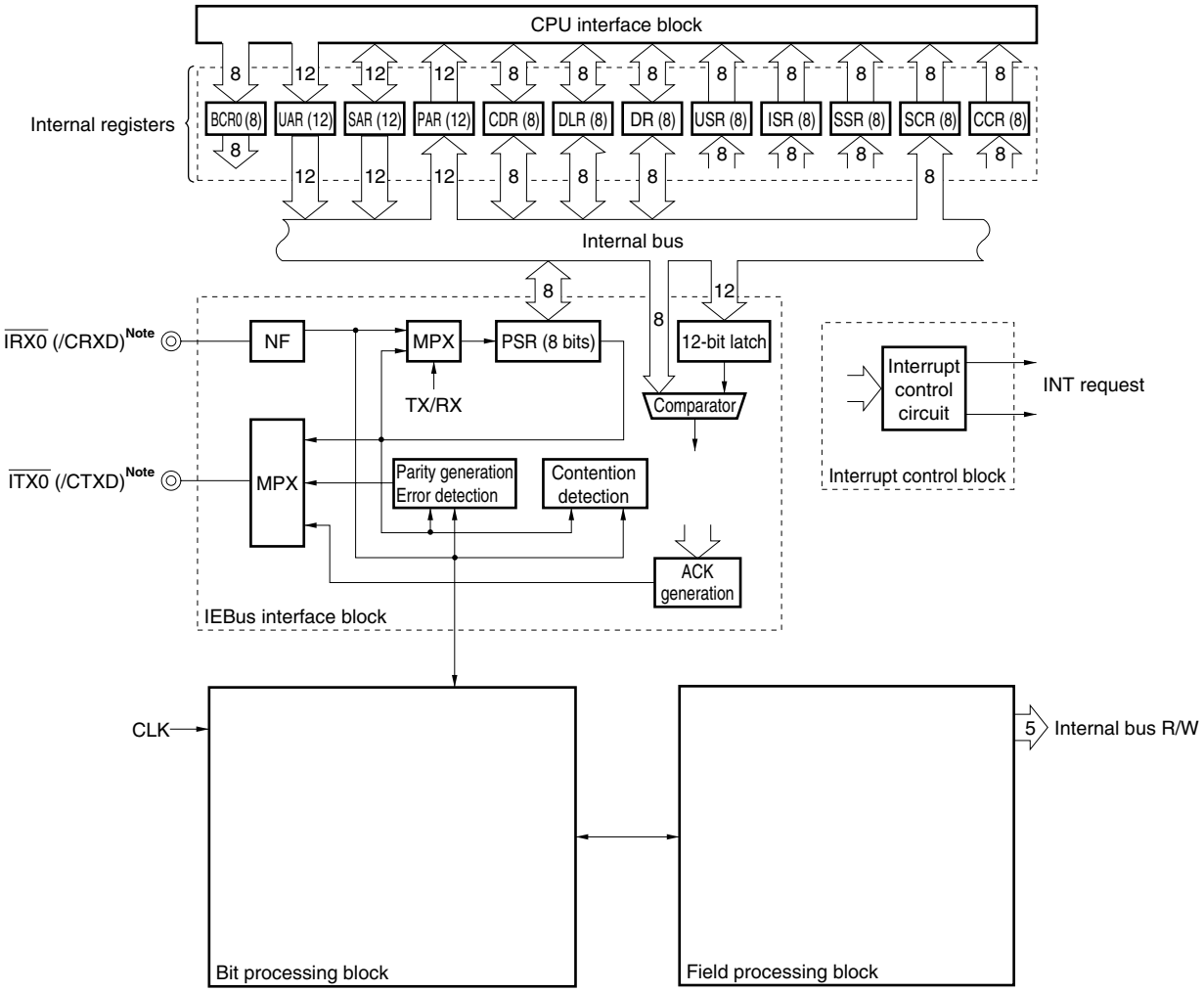
Notes 1. The μPD780701Y Subseries supports the IEBus controller when f_X = 6.29 MHz. Frequencies other than 6.29 MHz are not supported.

2. Automatic master reprocessing: After generating the master request, if the master request is cancelled by arbitration, etc., the bus is released and the master automatically re-issues the master request.

16.3 Configuration of IEBus Controller

Figure 16-10 shows the block diagram of the IEBus controller.

Figure 16-10. Block Diagram of IEBus Controller



Note Values in parentheses apply to the μPD78F0701Y only.

(1) Hardware configuration and functions

The IEBus mainly consists of the following six internal blocks.

- CPU interface block
- Interrupt control block
- Internal registers
- Bit processing block
- Field processing block
- IEBus interface block

(a) CPU interface block

This is a control block that interfaces between the CPU and the IEBus.

(b) Interrupt control block

This control block transfers interrupt request signals from the IEBus to the CPU.

(c) Internal registers

These registers set data to the control registers and fields that control the IEBus (for the internal registers, refer to **16.4 Internal Registers of IEBus Controller**).

(d) Bit processing block

This block generates and disassembles bit timing, and mainly consists of a bit sequence ROM, 8-bit preset timer, and comparator.

(e) 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.

(f) IEBus interface block

This is the interface block for an external driver/receiver, and mainly consists of a noise filter, shift register, conflict detector, parity detector, parity generator, and $\overline{\text{ACK}}$/NACK generator.

16.4 Internal Registers of IEBus Controller

The IEBus controller consists of the following registers.

16.4.1 Internal register list

Table 16-8. Internal Registers of IEBus Controller

Address	Special Function Register (SFR) Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFAFH	IEBus control register	BCR0	R/W	√	√	—	00H
FFB1H	IEBus control data register	CDR		—	√	—	01H
FFB2H	IEBus unit address register	UAR		—	—	√	0000H
FFB4H	IEBus slave address register	SAR		—	—	√	
FFB6H	IEBus partner address register	PAR	R	—	—	√	
FFB9H	IEBus telegraph length register	DLR	R/W	—	√	—	01H
FFBAH	IEBus data register	DR		—	√	—	00H
FFBBH	IEBus unit status register	USR	R	√	√	—	
FFBCH	IEBus interrupt status register	ISR	R/W	√	√	—	
FFBDH	IEBus slave status register	SSR	R	√	√	—	41H
FFBEH	IEBus communication success counter	SCR		—	√	—	01H
FFBFH	IEBus transmit counter	CCR		—	√	—	20H

Cautions 1. The above registers are mapped to the SFR space.

2. UAR, SAR, and PAR must be manipulated in 1-word units.

3. The Read-Modify-Write instructions (such as XCH and ROL4) cannot be used for DR, CDR, and DLR.

16.4.2 Description of internal registers

The internal registers incorporated in the IEBus controller are described below.

(1) IEBus control register 0 (BCR0)

Figure 16-11. Format of IEBus Control Register 0 (BCR0)

Address: FFAFH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
BCR0	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 the IEBus is operating as the master, writing to the BCR0 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 nested. However, if the IEBus is specified as a slave while a master request is being held pending, the BCR0 register can be written to at the end of communication to clear the communication end/frame end flag. This is also the case when communication has been forcibly stopped (ENIEBUS flag = 0).

2. If a bit manipulation instruction for the BCR0 register conflicts with a hardware reset of the MSTRQ flag, the BCR0 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 BCR0 register.**

(a) Communication enable flag (ENIEBUS)...Bit 7

<Set/reset conditions>

Set: By software

Reset: By software

Caution Before setting the ENIEBUS flag, make the following setting.

- **Set the interrupt enabled (EI) status and enable the interrupt servicing of INTIE2 (IEMK2 = 0).**
- **Set the IEBus unit address register (UAR)**

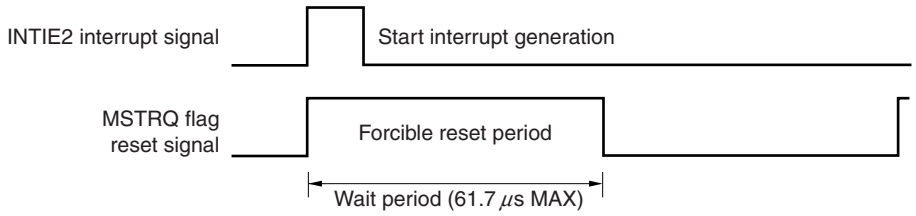
(b) Master request flag (MSTRQ)...Bit 6

<Set/reset conditions>

Set: By software

Reset: By hardware, at the end of the arbitration period. Because the reset signal is generated in the ACK period of the slave address field, if a MSTRQ flag setting instruction is sent in this period, it will be invalid.

Cautions 1. The master request should be resent by software following a loss in arbitration. When resending the master request in this case, set (1) the MSTRQ flag after securing the required wait period. This flag is unable to be set (1) before the end of this wait period.



2. When a master request has been sent and bus mastership acquired, do not set the MSTRQ, ENSLVTX, or ENSLVRX flag until the end of communication (i.e. the ISR register's communication end/frame end flag is set (1)) as setting these flags disables interrupt request generation. However, these flags can be set if communication has been aborted.

(c) Broadcast request flag (ALLRQ)...Bit 5

<Set/reset conditions>

Set: By software

Reset: By software

Caution When requesting broadcast communication, always set the ALLRQ flag, then the MSTRQ flag.

(d) Slave transmission enable flag (ENSLVTX)...Bit 4

<Set/reset conditions>

Set: By software

Reset: By software

Cautions 1. Clear the ENSLVTX flag before setting the MSTRQ flag when making a master request. If a slave transmission request is sent in slave mode when the ENSLVTX flag is unset, NACK in the control field will be returned. Moreover, when returning to an enabled state from a disabled state, transmission becomes valid from the next frame.

2. If the controller receives control data for data/control writing (3H, 7H) when the ENSLVTX flag is unset, NACK will be returned via the acknowledge bit of the control field.

3. The ENSLVTX flag will be set and the status interrupt (INTIE2) will be generated when the control data (0H, 4H, 5H, 6H) of a slave status request is returned, even if the ENSLVTX flag is in the reset status. At this time, the data returned via the acknowledge bit of the control field ($\overline{\text{ACK}}$ or NACK) depends on the status of the local unit and the received control data.

(e) Slave reception enable flag (ENSLVRX)...Bit 3

<Set/reset conditions>

Set: By software

Reset: By software

Caution If the ENSLVRX flag is reset when the IEBus is busy with other CPU processing, NACK will be returned via the acknowledge bit of the control field, making it possible to disable slave reception. Note that resetting this flag only disables individual communication, not broadcast communication. If the received slave address matches the unit address during individual communication, however, the start interrupt (INTIE2) is generated. If CPU processing has priority (neither reception nor transmission occurs), be sure to stop the IEBus unit by resetting the ENIEBUS flag. Note also that when returning to an enabled state from a disabled state, transmission becomes valid from the next frame.

(2) IEBus unit address register (UAR)

This register sets the local address of an IEBus unit. This register must be always set before starting communication.

The unit address (12 bits) is set to bits 11 to 0.

Figure 16-12. Format of IEBus Unit Address Register (UAR)

Address: FFB2H After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UAR	0	0	0	0												

(3) IEBus slave address register (SAR)

When a master request is issued, the value of this register is reflected in the value of the transmit data in the slave address field. This register must always be set before starting communication.

The slave address (12 bits) is set to bits 11 to 0.

Figure 16-13. Format of IEBus Slave Address Register (SAR)

Address: FFB4H After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAR	0	0	0	0												

(4) IEBus partner address register (PAR)

(a) Slave unit

The value of the receive data in the master address field (address of the master unit) is written to this register.

If a request “4H” to read the lock address (lower 8 bits) is received from the master, the CPU must read the value of this register, and write the data of the lower 8 bits to the IEBus data register (DR).

If a request “5H” to read the lock address (higher 4 bits) is received from the master, the CPU must read the value of this register and write the data of the higher 4 bits to the IEBus data register (DR).

The partner address (12 bits) is set to bits 11 to 0.

Figure 16-14. Format of IEBus Partner Address Register (PAR)

Address: FFB6H After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAR	0	0	0	0												

(5) IEBus control data register (CDR)

(a) Master unit

The data of the lower 4 bits is reflected in the data transmitted in the control field. When a master request is issued, this register must be set in advance before starting communication.

(b) Slave unit

The data received in the control field is written to the lower 4 bits.

When the status transmission flag (STATUSF) of the IEBus interrupt status register (ISR) is set, an interrupt (INTIE2) is issued, and each processing should be performed by software, according to the value of the lower 4 bits of CDR.

Figure 16-15. Format of IEBus Control Data Register (CDR)

Address: FFB1H After reset: 01H R/W

Symbol	7	6	5	4	3	2	1	0
CDR	0	0	0	0	CDR3	CDR2	CDR1	CDR0

CDR3	CDR2	CDR1	CDR0	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 (higher 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", it must read the value of this register after completing communication.**
2. **Instructions in Read Modify Write mode (such as XCH and ROL4) cannot be used for CDR.**
3. **If the master unit sets an undefined value, NACK is returned from the slave unit, and communication is aborted. During broadcast communication, however, the master unit continues communication without recognizing $\overline{\text{ACK}}$/NACK; therefore, make sure not to set an undefined value to this register during broadcast communication.**
4. **In the case of defeat in a bus conflict and a slave status request is received from the unit that won, the IEBus telegraph length register (DLR) is fixed to "01H". Therefore, when a re-request of the master follows, the appointed telegraph length must be set to DLR.**

(c) Slave status return operation

When the IEBus receives a request to transfer from master to slave status (control data: 0H, 6H) or a lock address request (4H, 5H), whether $\overline{ACK}$ 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 → $\overline{ACK}$ returned
(2) If 4H or 5H control data was received in the unlocked state → $\overline{ACK}$ not returned
(3) If 0H, 4H, 5H, or 6H control data was received in the locked state from the unit that sent the lock request → $\overline{ACK}$ 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 → $\overline{ACK}$ returned
(5) If 6H control data was received in the locked state from other than the unit that sent the lock request → $\overline{ACK}$ not returned

In all of the above cases, the acknowledgment of a slave status or lock address request will cause the STATUSF flag (bit 4 of the ISR register) to be set and the status interrupt request (INTIE2) to be generated. The generation timing is at the end of the control field parity bit (at the start of the $\overline{ACK}$ bit). However, if $\overline{ACK}$ is not returned, a NACK error is generated after the $\overline{ACK}$ bit, and communication is terminated.

Figure 16-16. Interrupt Generation Timing (for (1), (3), and (4))

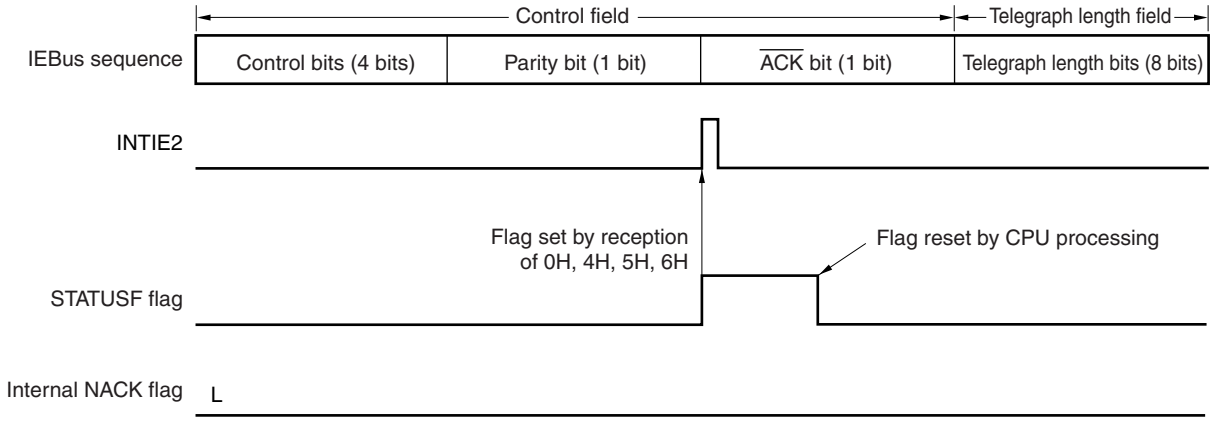
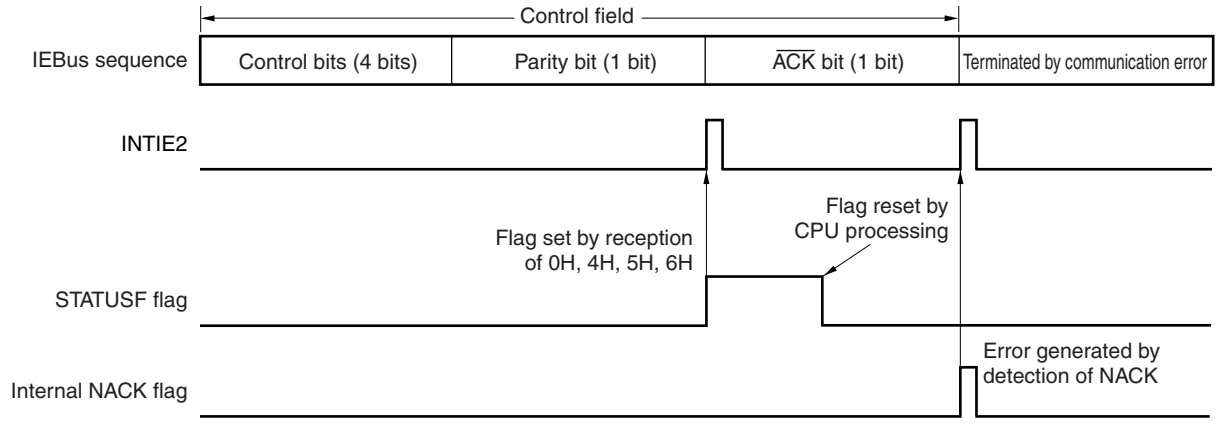
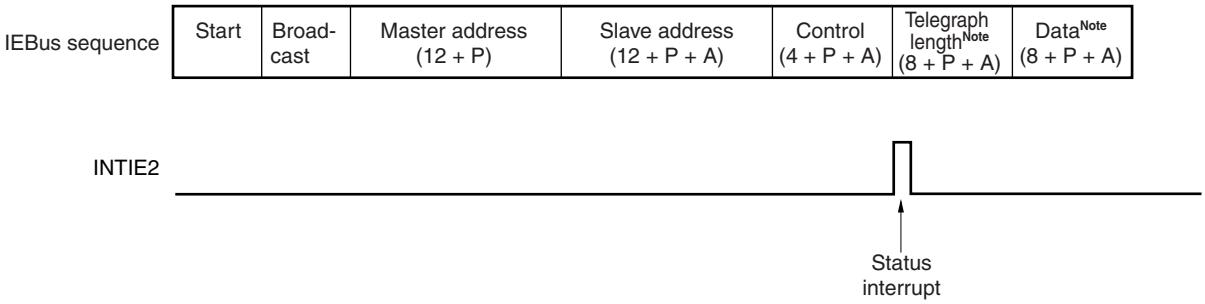


Figure 16-17. Interrupt Generation Timing (for (2) and (5))



Because in (4) and (5) the communication was from other than the unit that sent the lock request while the IEBus was in the locked state, the start or communication complete interrupt (INTIE2) is not generated, even if the IEBus unit is the communication target. The STATUSF flag (bit 4 of the IEBus interrupt status register (ISR)) is set and the status interrupt request (INTIE2) is generated, however, if a slave status or lock address request is acknowledged. Note that even if the same control data is received while the IEBus is in the locked state, the interrupt generation timing for INTIE2 differs depending on whether the master unit (3) or another unit (4) is requesting the locked state.

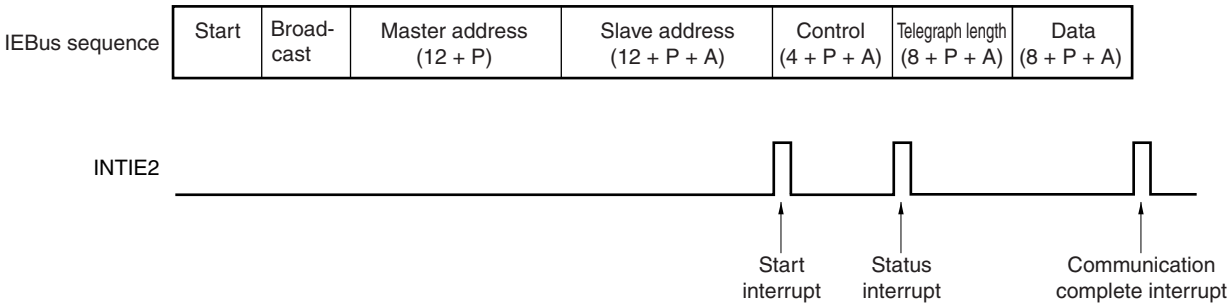
Figure 16-18. Timing of INTIE2 Interrupt Generation in Locked State (for (4) and (5))



Note The telegraph length and data modes are not set in the case of (5) because $\overline{ACK}$ is not returned.

Remark P: Parity bit, A: $\overline{ACK}$/NACK bit

Figure 16-19. Timing of INTIE2 Interrupt Generation in Locked State (for (3))



Remark P: Parity bit, A: $\overline{ACK}$/NACK bit

(6) IEBus telegraph length register (DLR)

(a) 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. This register must be set in advance before transmission.

(b) 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 IEBus telegraph length 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.

Figure 16-20. Format of IEBus Telegraph Length Register (DLR)

Address: FFB9H After reset: 01H R/W

Symbol	7	6	5	4	3	2	1	0
DLR								

Bit								Setting value	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. **If the master issues a request "0H, 4H, 5H, or 6H" to transmit a slave status and lock address (higher 4 bits, lower 8 bits), the contents of this register are set to "01H" by hardware; therefore, the CPU does not have to set this register.**
2. **In the case of defeat in a bus conflict and a slave status request is received from the unit that won, the IEBus telegraph length register (DLR) is fixed to "01H". Therefore, when a re-request of the master follows, the appointed telegraph length must be set to DLR.**
3. **Instructions in Read Modify Write mode (such as XCH and ROL4) cannot be used for DLR.**

(7) IEBus data register (DR)

The IEBus data register (DR) sets the communication data. The communication data (8 bits) is set to bits 7 to 0.

Remark The IEBus data 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) Transmission unit

The data (1 byte) written to the IEBus data register (DR) is stored in the shift register of the IEBus. It is then output from the most significant bit, and an interrupt (INTIE1) is issued to the CPU each time 1 byte has been transmitted. If NACK is received after 1-byte data has been transmitted during individual transfer, the next data is not transferred from DR to the shift register, and the same data is retransmitted. At this time, INTIE1 is not generated.

INTIE1 is issued when the IEBus interface shift register stores the IEBus data register (DR) value. However, when the last byte and 32nd byte (the last byte of 1 communication frame) are stored in the shift register, INTIE1 is not issued.

(b) Reception unit

One byte of the data received by the shift register of the IEBus interface block is stored to this register. Each time 1 byte has been correctly received, an interrupt (INTIE1) is issued.

Figure 16-21. Format of IEBus Data Register (DR)

Address: FFBAH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
DR								

Cautions

1. **If the next data is not set in time while the transmission unit is set, an underrun occurs, and a communication error interrupt (INTIE2) occurs, stopping transmission.**
2. **When the IEBus is a receiving unit, if the reading of the data is too late for the next data reception timing, the unit will enter the overrun state. At this time, during individual communication reception, NACK will be returned at the acknowledge bit of the data field, and the master unit will be requested to retransmit the data. If an overrun error occurs during broadcast communication reception, the communication error interrupt (INTIE2) is generated.**
3. **Instructions in Read Modify Write mode (such as XCH and ROL4) cannot be used for DR.**

(8) IEBus unit status register (USR)

Figure 16-22. Format of IEBus Unit Status Register (USR)

Address: FFBBH After reset: 00H R

Symbol	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 win
1	Arbitration loss

ALLTRNS	Broadcast communication flag
0	Individual communication status
1	Broadcast communication status

ACK	$\overline{\text{ACK}}$ transmission flag
0	NACK transmitted
1	$\overline{\text{ACK}}$ 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/reset conditions>

Set: When the unit is requested as a slave (if the received slave address and unit UAR match during individual communication reception, or if the higher 4 bits of the received slave address match or if the received slave address is FFFH during broadcast communication reception), this flag is set by hardware when the acknowledge period of the slave address field starts.

Reset: This flag is reset by hardware when the unit is not requested as a slave. 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 is set), and if a parity error occurs in the slave address field for that communication, the flag is not reset.

(b) Arbitration result flag (ARBIT)...Bit 5

A flag that indicates the result of arbitration.

<Set/reset conditions>

Set: When the data output by the IEBus unit during the arbitration period does not match the bus line data after the master request.

Reset: By the start bit timing.

Cautions 1. The timing at which the arbitration result flag (ARBIT) is reset differs depending on whether the unit outputs a start bit.

- **If start bit is output: The flag is reset at the output start timing.**
- **If start bit is not output: The flag is reset at the detection timing of the start bit (approx. 160 μs after output)**

2. The flag is reset at the detection timing of the start bit if the other unit outputs the start bit earlier and the local unit does not output the start bit after the master request.

(c) Broadcast communication flag (ALLTRNS)...Bit 4

A flag that indicates 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/reset conditions vary depending on the receive data of the broadcast field bit.

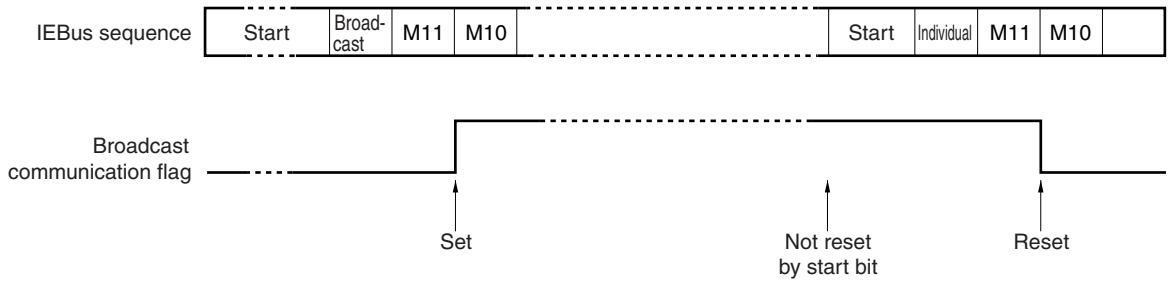
<Set/reset conditions>

Set: When “Broadcast” is received by the broadcast field

Reset: When “Individual” is received by the broadcast field, or upon the input of a system reset.

Caution The broadcast communication flag is updated regardless of whether the IEBus is the communication target or not.

Figure 16-23. Example of Broadcast Communication Flag Operation



(d) $\overline{\text{ACK}}$ transmission flag ($\overline{\text{ACK}}$)...Bit 3

A flag that indicates whether $\overline{\text{ACK}}$ has been transmitted in the $\overline{\text{ACK}}$ period of the $\overline{\text{ACK}}$ field when the IEBus is a receiving unit. The contents of the flag are updated in the $\overline{\text{ACK}}$ 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 $\overline{\text{ACK}}$ period of that field.

(e) Lock status flag (LOCK)...Bit 2

A flag that indicates whether the unit is locked.

<Set/reset conditions>

Set: When the communication end flag (ENDTRNS) goes low level and the frame end flag (ENDFRAM) goes high level after receipt of a lock specification (3H, 6H, AH, BH) in the control field.

Reset: When the communication enable flag (ENIEBUS) is cleared.
When the communication end flag is set 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.

Remark ENDTRNS: Bit 3 of the IEBus interrupt status register (ISR)
ENDFRAM: Bit 2 of the IEBus interrupt status register (ISR)
ENIEBUS: Bit 7 of IEBus control register 0 (BCR0)

(9) IEBus interrupt status register (ISR)

This register indicates the status when the IEBus issues an interrupt. ISR is read to generate an interrupt, after which the specified interrupt servicing is carried out.

Reset the ISR register after reading it. Until it is reset, the INTIE2 interrupt signal is not generated (nor held pending).

To reset the ISR register, reset each flag, satisfying the reset conditions in Table 16-9.

Table 16-9. Reset Conditions of Flags in ISR Register

Flag Name	Reset Condition	Processing Example
IEERR, STARTF, STATUSF	Byte write operation of ISR register. Any value can be written.	ISR = 00H, etc.
ENDTRNS, ENDFRAM	Set MSTRQ, ENSLVTX, or ENSLVRX flag.	BCR0 register = 88H or ENSLVTX = 1, etc.

Caution Even if 0 is written to the ENDTRNS or ENDFRAM flag by accessing the ISR register, these flags are not reset. Reset them as described above.

Remark MSTRQ: Bit 6 of IEBus control register 0 (BCR0)
ENSLVTX: Bit 4 of IEBus control register 0 (BCR0)
ENSLVRX: Bit 3 of IEBus control register 0 (BCR0)

Figure 16-24. Format of IEBus Interrupt Status Register (ISR)

Address: FFBCH After reset: 00H R/W

Symbol	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 does not occur
1	Start interrupt occurs

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 (32 bytes) prescribed by mode 1) did not end
1	The frame (transfer of the maximum number of bytes (32 bytes) prescribed by mode 1) ended

Caution Each of IEERR, STARTF, STATUSF, ENDTRNS, and ENDFRAM are generation triggers for the interrupt request signal (INTIE2) (see Figure 16-28 Configuration of Interrupt Control Block). Because of this, if any one of these interrupt triggers has been set, no new interrupt will be generated by a subsequent trigger. Clear the flag of the interrupt source by the interrupt servicing program before the next interrupt occurs.

(a) Communication error flag (IEERR)...Bit 6

A flag that indicates the detection of an error during communication.

<Set/reset conditions>

Set: The flag is set if a timing error, parity error (except in the data field), NACK reception (except in the data field), underrun error, or overrun error (that occurs during broadcast communication reception) occurs.

Reset: By software

(b) Start interrupt flag (STARTF)...Bit 5

A flag that indicates whether an interrupt was in the ACK period of the slave address field.

<Set/reset conditions>

Set: In the slave address field, upon a master request. When the IEBus is a slave unit, this flag is set upon a request from the master (only if it was a slave request in the locked state from the unit requesting a lock).

Reset: By software

(c) Status transmission flag (STATUSF)...Bit 4

A flag indicating that the transmission status is either the master to slave status, or the lock address (higher 4 bits, lower 8 bits), when the IEBus is a slave unit.

<Set/reset conditions>

Set: When 0H, 4H, 5H, or 6H is received in the control field from the master when the IEBus is a slave unit.

Reset: By software

(d) Communication end flag (ENDTRNS)...Bit 3

A flag that indicates whether communication ends after the number of bytes set in the telegraph length field have been transferred.

<Set/reset conditions>

Set: When the value of the IEBus communication success counter (SCR) is 0.

Reset: When the MSTRQ, ENSLVTX, or ENSLVRX flag of IEBus control register 0 (BCR0) is set.

(e) Frame end flag (ENDFRAM)...Bit 2

A flag that indicates whether communication ends after the maximum number of bytes (32 bytes) prescribed by mode 1 have been transferred.

<Set/reset conditions>

Set: When the value of the IEBus transmit counter (CCR) is 0.

Reset: When the MSTRQ, ENSLVTX, or ENSLVRX flag of IEBus control register 0 (BCR0) is set.

(f) Communication error triggers

- **Timing error**

  Occurrence conditions: Occurs if the high/low level width of the communication bit has shifted from the prescribed value.

  Remark: The respective prescribed values are set in the bit processing block and monitored by the internal 8-bit timer. An interrupt is generated when a timing error occurs.

- **Parity error**

  Occurrence conditions: Occurs if the generated parity and the received parity in each field do not match when the IEBus is a receiving unit.

  Remark: During individual communication, an interrupt is generated if a parity error occurs in a field other than the data field.
  During broadcast communication, an interrupt is generated even if a parity error occurs in the data field.

  Restriction: If there is a slave request that has lost in arbitration to a broadcast request, no interrupt is generated, even if a parity error occurs.

- **NACK reception**

  Occurrence conditions: This error occurs when NACK is received during the $\overline{\text{ACK}}$ period in each of the slave address, control, and telegraph length fields during individual communication, regardless of whether the unit is the master or a slave unit.
  A NACK reception only occurs in individual communication. $\overline{\text{ACK}}$ and NACK are not discriminated in broadcast communication.

  Remark: An interrupt is generated if NACK is received in a field other than the data field.

- **Underrun error**

  Occurrence conditions: Occurs during data transmission if there was insufficient time to write the next transmit data to the IEBus data register (DR) before $\overline{\text{ACK}}$ reception.

  Remark: An interrupt is generated if an underrun occurs.

- **Overrun error**

  Occurrence conditions: The data interrupt request (INTIE1) that stores each byte of data in the IEBus data register (DR) is generated, and the DR register is read by software. An overrun error occurs if this reading processing is late and its timing becomes that of the next data reception.

  Remark: In individual communication reception, an acknowledgment is not returned in the ACK period of this data, resulting in the retransmission of the data by the transmitting unit. Consequently, the IEBus transfer counter (CCR) is decremented, whereas the IEBus communication success counter (SCR) is not. In broadcast communication reception, reception is stopped by the occurrence of a communication error interrupt request (INTIE2), at which time the DR register is not updated. The STATRX flag (bit 1 of the SSR register) also remains set (1) without generating INTIE1. The overrun state is released at the timing of the next data reception following the reading of DR.

(g) Overrun error - supplementary details

(i) When the frame ends in the overrun state during individual communication reception

If the DR register is not read after entering the overrun state and the retransmitted data reaches the maximum number of bytes (32 bytes), the frame end interrupt (INTIE2) is generated. The overrun state is maintained until the DR register is read after the end of the frame.

(ii) If the next reception is started in the case of (i) above, or if the next reception is started without the DR register being read after the final data has been received, regardless of whether the communication is broadcast or individual

Even if communication to the IEBus unit starts in the overrun state, the cause of the overrun, NACK, is not returned in the ACK period of the slave address, control, or telegraph length field (the DR register is not updated). If the next communication is not to the IEBus unit, the DR register is not updated until it is read. Because the IEBus unit is not a communication target, the data interrupt (INTIE1) and communication error interrupt (INTIE2) are not generated.

(iii) If the next transmission occurs in the overrun state

The data to be transmitted next in the overrun state can be no more than 2 bytes long.

Because the data request interrupt (INTIE1) is not generated, the transmit data cannot be set, resulting in an underrun error. Therefore, clear the overrun status before starting transmission.

(iv) Overrun state release

The overrun state can only be released by reading the DR register or by a system reset. Therefore, be sure to read DR in the communication error interrupt servicing program.

(10) IEBus slave status register (SSR)

This register indicates the communication status of the slave unit. After receiving a slave status transmission request from the master, the CPU reads this register, and writes the slave status to the IEBus data register (DR) to transmit the slave status. At this time, the telegraph length is automatically set to “01H”, so setting of the IEBus telegraph length register (DLR) 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 “01H” (mode 1).

Figure 16-25. Format of IEBus Slave Status Register (SSR)

Address: FFBDH After reset: 41H R

Symbol	7	6	5	4	3	2	1	0
SSR	0	1	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 receive status
0	Receive data not stored in DR
1	Receive data stored in DR

STATTX	DR transmit status
0	Transmit data not stored in DR
1	Transmit data stored in DR

(a) Slave transmission status flag (STATSLV)...Bit 4

Reflects the contents of the slave transmission enable flag.

(b) Lock status flag (STATLOCK)...Bit 2

Reflects the contents of the locked flag.

(c) DR reception status (STATRX)...Bit 1

This flag indicates the DR reception state.

(d) DR transmission status (STATTX)...Bit 0

This flag indicates the DR transmission state.

(11) IEBus communication success counter (SCR)

The IEBus communication success counter (SCR) indicates the number of remaining communication bytes.

This register reads the count value of the counter in the value set by the IEBus telegraph length register (DLR) is decremented by $\overline{ACK}$ in the data field. When the count value has reached "00H", the communication end flag (ENDTRNS) of the IEBus interrupt status register (ISR) is set.

Figure 16-26. Format of IEBus Communication Success Counter (SCR)

Address: FFBEH After reset: 01H R

Symbol	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 hard 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 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.

(12) IEBus transmit counter (CCR)

The IEBus transmit counter (CCR) indicates the number of remaining bytes of the communication byte number specified in the communication mode.

Bits 7 to 0 of the IEBus transmit counter (CCR) indicate the number of transfer bytes.

This register reads the count value of the counter that is preset to the maximum number of transmitted bytes (32 bytes) per frame specified in mode 1. Whereas SCR (IEBus communication success counter) is decremented during normal communication ($\overline{ACK}$), CCR is decremented when 1 byte has been communicated, regardless of whether $\overline{ACK}$ or NACK. When the count value has reached "00H", the frame end flag (ENDFRAM) of the IEBus interrupt status register (ISR) is set.

The maximum number of transfer bytes of the preset value of mode 1 per frame is 20H (32 bytes).

Figure 16-27. Format of IEBus Transmit Counter (CCR)

Address: FFBFH After reset: 20H R

Symbol	7	6	5	4	3	2	1	0
CCR								

16.5 Interrupt Operations of IEBus Controller

16.5.1 Interrupt control block

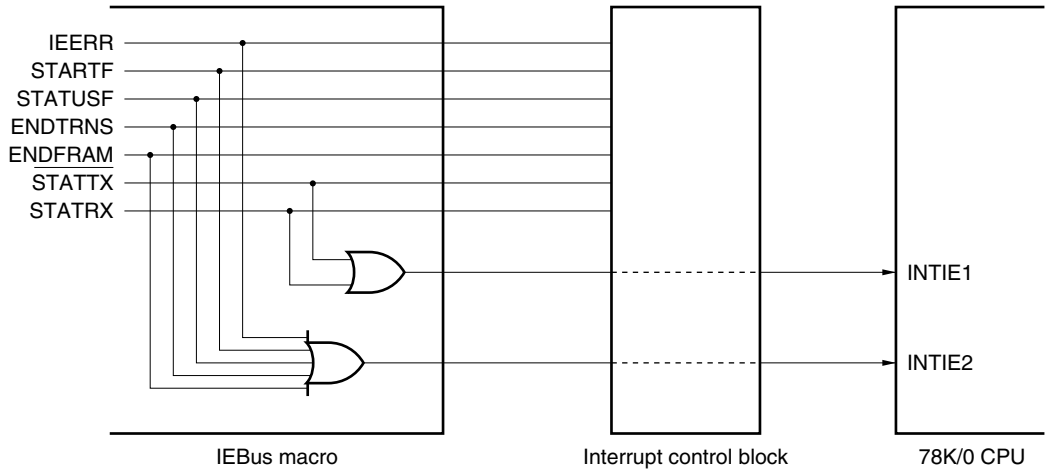
The interrupt request signals are shown below.

<1>	Communication error	IEERR
<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

<1> to <5> of the above interrupt requests are assigned to the IEBus interrupt status register (ISR). For details, refer to **Table 16-10 Interrupt Source List**.

The configuration of the interrupt control block is illustrated below.

Figure 16-28. Configuration of Interrupt Control Block



Caution OR output of IEERR, STARTF, STATUSF, ENDTRNS, and ENDFRAM is treated as a vectored interrupt request signal (INTIE2).

16.5.2 Interrupt source list

The interrupt sources are listed below.

Table 16-10. Interrupt Source List

Interrupt Source		Condition of Generation		CPU Processing After Generation of Interrupt	Remark
		Unit	Field		
Communication error	Timing error	Master/slave	All fields	Undo communication processing	Communication error is OR output of timing error, parity error, NACK reception, underrun error, and overrun 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)		
Start interrupt		Master	Slave/address	Slave request judgment Conflict judgment (If lost, remaster processing) Communication preparation processing	Interrupt always occurs if conflict lost when master request is issued.
		Slave	Slave/address	Slave request judgment Communication preparation processing	Generated only when slave request is issued.
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.
End of communication		Transmission	Data	End processing by software	Set if SCR is cleared to 0
		Reception	Data	End processing by software Receive data processing	
End of frame		Transmission	Data	Retransmission preparation processing	Set if CCR is cleared to 0
		Reception	Data	Re-reception preparation processing	
Transmit data write		Transmission	Data	Transmit data write processing by software	Set after transfer of transmission data to internal shift register. This does not occur when the last data is transferred.
Receive data read		Reception	Data	Receive data read processing by software	Set after normal data reception

16.5.3 Communication error source processing list

The following table shows the occurrence conditions of the communication errors (timing error, NACK reception, overrun error, underrun error, and parity error), error processing by the internal IEBus controller, and examples of processing by software.

Table 16-11. Communication Error Source Processing List (1/3)

		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 occurs • To start bit waiting status **Remark** Communication between other units does not end.		• Transmission stops. • INTIE2 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 occurs. • NACK is returned. • To start bit waiting status		• Transmission stops. • INTIE2 occurs. • To start bit waiting status	
	Software processing	• Error processing (such as retransmission request)		• Error processing (such as retransmission request)	

		NACK Reception				
Occurrence condition	Unit status	Reception		Transmission		
	Occurrence condition	Unit NACK transmission		NACK reception		
	Location of occurrence	Other than data field	Data field	Other than data field	Data field	NACK reception of data of 32nd byte
Broadcast communication	Hardware processing	–	–	–	–	–
	Software processing	–	–	–	–	–
Individual communication	Hardware processing	• Reception stops. • INTIE2 occurs. • To start bit waiting status	• INTIE2 does not occur. • Data retransmitted by other unit is received.	• Transmission stops. • INTIE2 occurs. • To start bit waiting status	• INTIE2 does not occur. • Retrans-mission processing	• INTIE2 occurs[Note]. • 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)

Note Both ISR.6 (IEERR) and ISR.2 (ENDFRAM) are set to 1.
To reset them, satisfy the conditions in Table 16-9.

Table 16-11. Communication Error Source Processing List (2/3)

		Overrun Error		Underrun Error	
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	DR cannot be read in time before the next data is received.		DR 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 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 occurs. • To start bit waiting status
	Software processing	–	• DR is read and overrun status is cleared. • Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)
Individual communication	Hardware processing	–	• INTIE2 does not occur. • NACK is returned. • Data is retransmitted from other unit. **Remark** Data cannot be received until overrun status is cleared.	–	• Transmission stops. • INTIE2 occurs. • To start bit waiting status
	Software processing	–	• DR is read and overrun status is cleared. • Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)

Table 16-11. Communication Error Source Processing List (3/3)

		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 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 occurs. • To start bit waiting status	• Reception does not stop. • INTIE2 does not occur. • NACK is returned. • Data retransmitted by other unit is received.	–	–
	Software processing	• Error processing (such as retransmission request)	–	–	–

16.6 Interrupt Generation Timing and Main CPU Processing

16.6.1 Master transmission

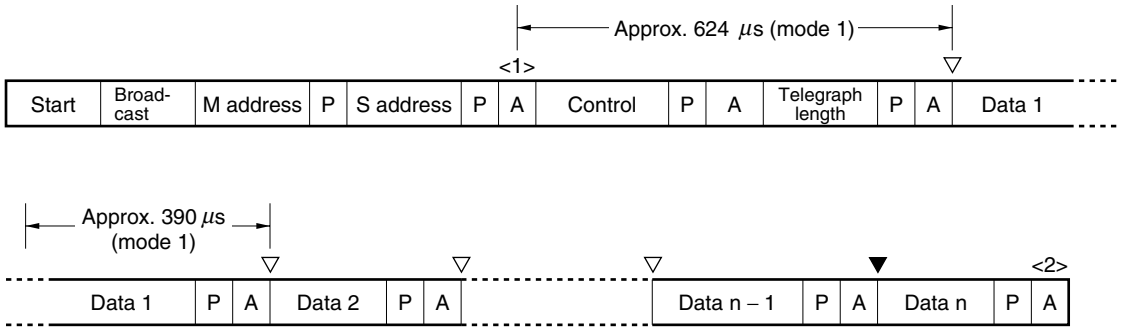
Initial preparation processing:

Set a unit address, slave address, control data, telegraph length, and the first byte of the transmit data.

Communication start processing:

Set the bus control register (enable communication, master request, and slave reception).

Figure 16-29. Master Transmission



<1> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of slave request → Slave reception processing
(See **16.6.1 (1) Slave reception processing**)

↓

Judgment of conflict result → Remaster request processing

<2> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Recommunication processing
(See **16.6.1 (3) Recommunication processing**)

Remarks 1. ▽: Interrupt (INTIE1) occurrence (See **16.6.1 (2) Interrupt (INTIE1) occurrence**)
The transmit data of the second and subsequent bytes are written to the IEBus data register (DR) by software. At this time, the data transfer direction is RAM (memory) → SFR (peripheral).

2. ▼: An interrupt (INTIE1) does not occur.

3. n = Final number of data bytes

(1) Slave reception processing

If a slave reception request is confirmed during vectored interrupt servicing, change the data transfer direction from RAM (memory) → SFR (peripheral) to SFR (peripheral) → RAM (memory) by software until the first data is received. The maximum pending period of this data transfer direction changing processing is about 1040 μs in communication mode 1.

(2) Interrupt (INTIE1) occurrence

If NACK is received from the slave in the data field, an interrupt (INTIE1) is not issued to the CPU, 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 occurs due to occurrence of underrun, and communication ends midway.

(3) Recommunication processing

The vectored interrupt servicing in <2> of Figure 16-29 judges 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.

16.6.2 Master reception

Before performing master reception, it is necessary to notify the slave unit 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 (ENSLVTX), and waits.

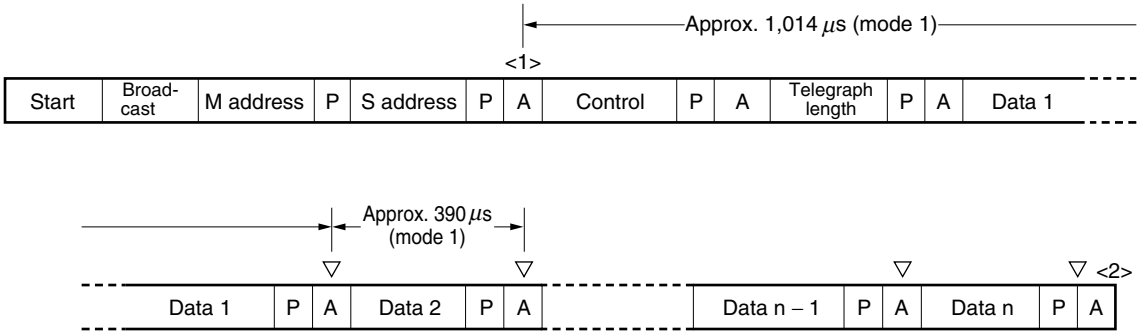
Initial preparation processing:

Set a unit address, slave address, and control data.

Communication start processing:

Set the bus control register (enable communication and master request).

Figure 16-30. Master Reception



<1> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing
↓
Judgment of slave request → Slave reception processing
↓
Judgment of conflict result → Remaster request processing

<2> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing
↓
Judgment of end of communication → End of communication processing
↓
Judgment of end of frame → Frame end processing (See **16.6.2 (2) Frame end processing**)

Remarks 1. ▽: Interrupt (INTIE1) occurrence (See **16.6.2 (1) Interrupt (INTIE1) occurrence**)
The receive data stored in the IEBus data register (DR) is read by software.
At this time, the data transfer direction is SFR (peripheral) → RAM (memory).

2. n = Final number of data bytes

(1) Interrupt (INTIE1) occurrence

If NACK is transmitted (hardware processing) in the data field, an interrupt (INTIE1) is not issued to the CPU, 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 NACK.

(2) Frame end processing

The vectored interrupt servicing in <2> of Figure 16-30 judges 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.

16.6.3 Slave transmission

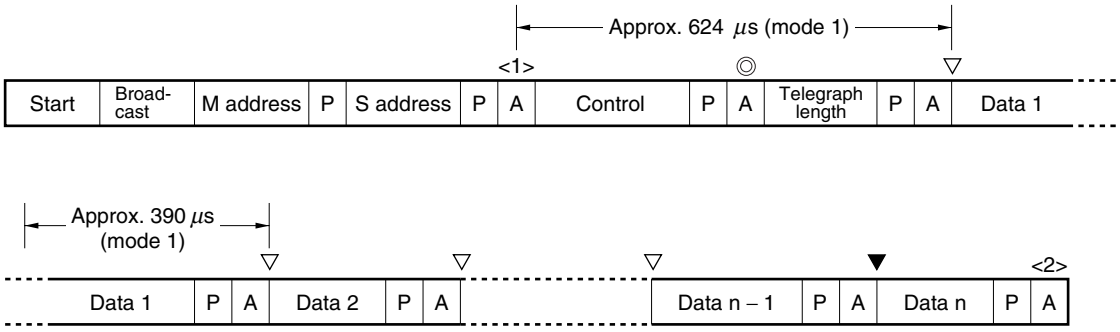
Initial preparation processing:

Set a unit address, telegraph length, and the first byte of the transmit data.

Communication start processing:

Set the bus control register (enable communication, slave transmission, and slave reception).

Figure 16-31. Slave Transmission



<1> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of slave request

<2> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Frame end processing (See **16.6.3 (2) Frame end processing**)

Remarks 1. ▽: Interrupt (INTIE1) occurrence (See **16.6.3 (1) Interrupt (INTIE1) occurrence**). The transmit data of the second and subsequent bytes are written to the IEBus data register (DR) by software. At this time, the data transfer direction is RAM (memory) → SFR (peripheral).

2. ▼: An interrupt (INTIE1) does not occur.

3. ◎: An interrupt (INTIE2) occurs only when 0H, 4H, 5H, or 6H is received in the control field in the slave status (for the slave status return operation in the locked state, refer to **16.4.2 (5) IEBus control data register (CDR)**).

4. n = Final number of data bytes

(1) Interrupt (INTIE1) occurrence

If NACK is received from the master in the data field, an interrupt (INTIE1) is not issued to the CPU, 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 occurs due to occurrence of underrun, and communication is abnormally ended.

(2) Frame end processing

The vectored interrupt servicing in <2> of Figure 16-31 judges 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.

16.6.4 Slave reception

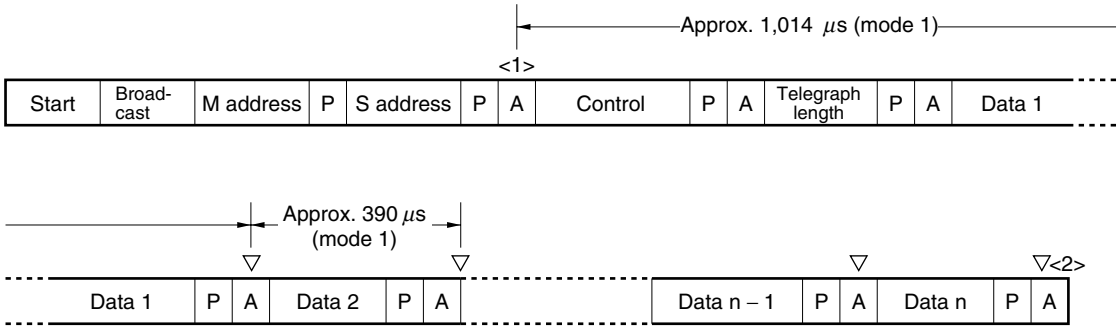
Initial preparation processing:

Set a unit address.

Communication start processing:

Set the bus control register (enable communication, disable slave transmission, and enable slave reception).

Figure 16-32. Slave Reception



<1> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of slave request → Slave processing

<2> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Frame end processing (See **16.6.4 (2) Frame end processing**)

Remarks 1. ▽: Interrupt (INTIE1) occurrence (See **16.6.4 (1) Interrupt (INTIE1) occurrence**).
The receive data stored in the IEBus data register (DR) is read by software.
At this time, the data transfer direction is SFR (peripheral) → RAM (memory).

2. n = Final number of data bytes

(1) Interrupt (INTIE1) occurrence

If NACK is transmitted in the data field, an interrupt (INTIE1) is not issued to the CPU, and the same data is retransmitted from the master.

If the receive data is not read by the time the next data is received, NACK is automatically transmitted.

(2) Frame end processing

The vectored interrupt servicing in <2> of Figure 16-32 judges whether the data has been correctly received within one frame.

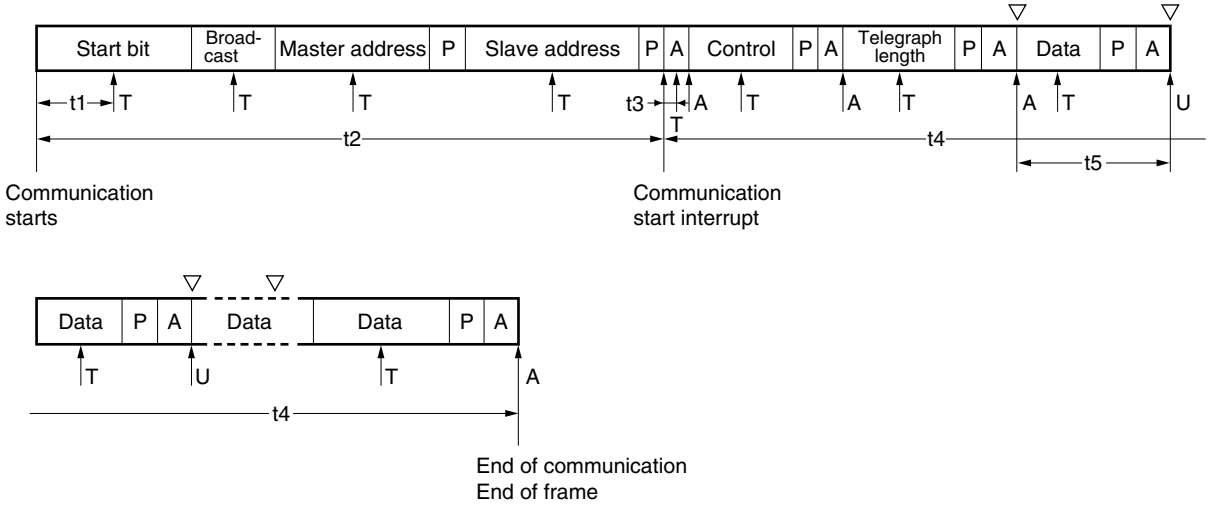
16.6.5 Interval of occurrence of interrupt for IEBus control

Each control interrupt must occur at each point of communication and perform the necessary processing until the next interrupt occurs. Therefore, the CPU must control the IEBus control block, taking the shortest time of this interrupt into consideration.

The locations at which the following error interrupts may occur are indicated by ↑ in the field where it may occur. ↑ does not mean that the interrupt occurs at each of the points indicated by ↑. If an error interrupt (timing error, parity error, NACK reception, underrun error, or overrun error) occurs, the IEBus internal circuit is initialized. As a result, subsequent interrupts do not occur in that communication frame.

(1) Master transmission

Figure 16-33. Master Transmission (Interval of Interrupt Occurrence)



Remarks 1. T: Timing error
A: NACK reception
U: Underrun error
▽: Data set interrupt (INTIE1)

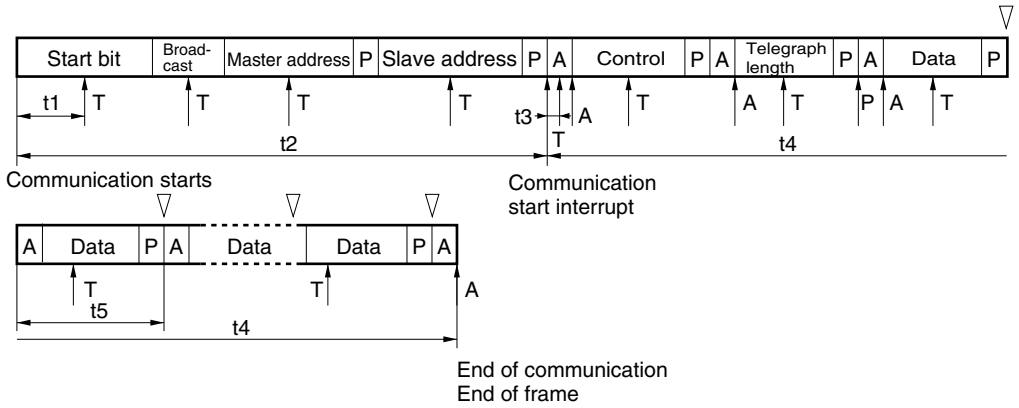
2. End of frame occurs at the end of 32-byte data.

(IEBus: 6.29 MHz operation)

Item	Symbol	MIN.	Unit
Communication starts - timing error	t1	Approx. 93	μs
Communication starts - communication start interrupt	t2	Approx. 1,282	μs
Communication start interrupt - timing error	t3	Approx. 15	μs
Communication start interrupt - end of communication	t4	Approx. 1,012	μs
Transmission data request interrupt interval	t5	Approx. 375	μs

(2) Master reception

Figure 16-34. Master Reception (Interval of Interrupt Occurrence)



Remarks 1. T: Timing error

P: Parity error

A: NACK reception

▽: Data set interrupt (INTIE1)

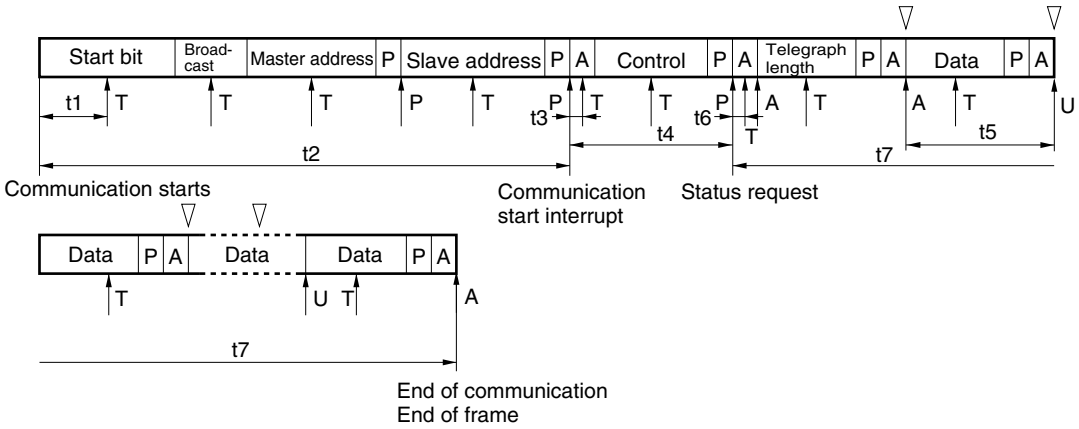
2. End of frame occurs at the end of 32-byte data.

(IEBus: 6.29 MHz operation)

Item	Symbol	MIN.	Unit
Communication starts - timing error	t1	Approx. 93	μs
Communication starts - communication start interrupt	t2	Approx. 1,282	μs
Communication start interrupt - timing error	t3	Approx. 15	μs
Communication start interrupt - end of communication	t4	Approx. 1,012	μs
Receive data read interval	t5	Approx. 375	μs

(3) Slave transmission

Figure 16-35. Slave Transmission (Interval of Interrupt Occurrence)



Remarks 1. T: Timing error
P: Parity error
A: NACK reception
U: Underrun error
▽: Data set interrupt (INTIE1)

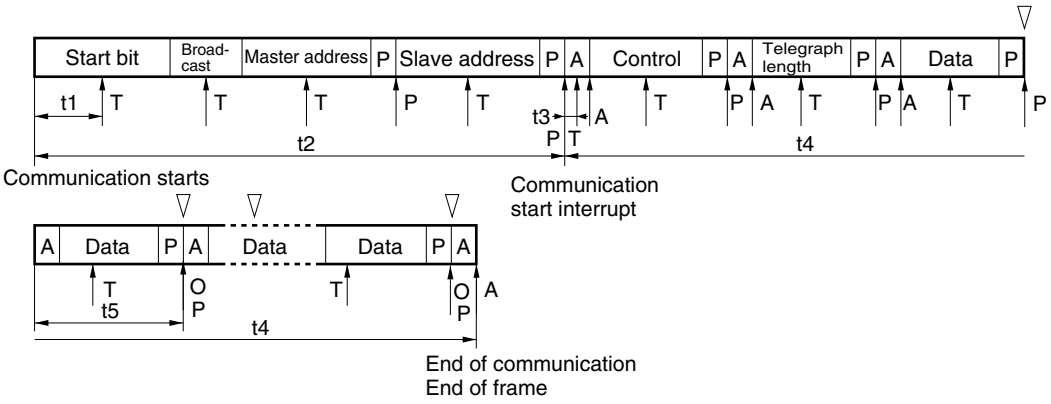
2. End of frame occurs at the end of 32-byte data.

(IEBus: 6.29 MHz operation)

Item	Symbol	MIN.	Unit
Communication starts - timing error	t1	Approx. 96	μs
Communication starts - communication start interrupt	t2	Approx. 1,192	μs
Communication start interrupt - timing error	t3	Approx. 15	μs
Communication start interrupt - status request	t4	Approx. 225	μs
Transmission data request interrupt interval	t5	Approx. 375	μs
Status request - timing error	t6	Approx. 15	μs
Status request - end of communication	t7	Approx. 787	μs

(4) Slave reception

Figure 16-36. Slave Reception (Interval of Interrupt Occurrence)



Remarks 1. T: Timing error
P: Parity error
A: NACK reception
O: Overrun error
▽: Data set interrupt (INTIE1)

2. End of frame occurs at the end of 32-byte data.

(IEBus: 6.29 MHz operation)

Item	Symbol	MIN.	Unit
Communication starts - timing error	t1	Approx. 96	μs
Communication starts - communication start interrupt	t2	Approx. 1,192	μs
Communication start interrupt - timing error	t3	Approx. 15	μs
Communication start interrupt - end of communication	t4	Approx. 1,012	μs
Receive data read interval	t5	Approx. 375	μs

CHAPTER 17 INTERRUPT FUNCTIONS

17.1 Interrupt Function Types

The following three types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged unconditionally. It does not undergo priority control and is given top priority over all other interrupt requests.

It generates a standby release signal.

One interrupt from the watchdog timer is incorporated as a non-maskable interrupt.

(2) Maskable interrupts

These interrupts undergo mask control. Maskable interrupts can be divided into a high interrupt priority group and a low interrupt priority group by setting the priority specify flag registers (PR0L, PR0H, PR1L, PR1H).

Multiple high priority interrupts can be applied to low priority interrupts. If two or more interrupts with the same priority are simultaneously generated, each interrupt has a predetermined priority (see **Table 17-1**).

A standby release signal is generated.

Nine external interrupt request sources and 20 (19 for the μPD780702Y) internal interrupt request source are incorporated as maskable interrupts.

(3) Software interrupt

This is a vectored interrupt to be generated by executing the BRK instruction. It is acknowledged even in an interrupt disabled stated. The software interrupt is not subject to interrupt priority control.

17.2 Interrupt Sources and Configuration

A total of 30 (29 for the μPD780702Y) interrupt sources exist among non-maskable, maskable, and software interrupts (see **Table 17-1**).

Table 17-1. Interrupt Source List (1/2)

Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	—	INTWDT	Watchdog timer overflow (with non-maskable interrupt selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (with interval timer selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTP4			000EH	
	6	INTP5			0010H	
	7	INTP6			0012H	
	8	INTP7			0014H	
	9	INTSER0	UART0 reception error generation	Internal	0016H	(B)
	10	INTSR0	End of UART0 reception		0018H	
	11	INTST0	End of UART0 transmission		001AH	
	12	INTCSI30	End of SIO30 transfer		001CH	
	13	INTCSI31	End of SIO31 transfer		001EH	
	14	INTIIC0	End of IIC0 transfer		0020H	
	15	INTCE[Note 3]	DCAN error		0022H	
	16	INTCR[Note 4] INTIE1[Note 5]	DCAN receive/ IEBus data access request		0024H	
	17	INTCT[Note 4] INTIE2[Note 5]	DCAN transmit/IEBus communication error and communication start/end		0026H	
	18	INTWTNI0	Reference time interval signal from watch timer		0028H	
	19	INTTM000	Generation of match signal between TM00 and CR000 (when compare register is specified). Detection of TI000 valid edge (when capture register is specified).		002AH	
	20	INTTM010	Generation of match signal between TM00 and CR010 (when compare register is specified). Detection of TI010 valid edge (when capture register is specified).		002CH	

Notes 1. The default priority is a priority applicable when two or more maskable interrupts are generated simultaneously. 0 is the highest and 28 is the lowest.

2. Basic configuration types (A) to (E) correspond to (A) to (E) in Figure 17-1.

3. Applies to the μPD780701Y and 78F0701Y only.

4. Applies to the μPD780701Y and 78F0701Y only. Selected when the DCAN controller is used in the μPD78F0701Y.

5. Applies to the μPD780702Y and 78F0701Y only. Selected when the IEBus controller is used in the μPD78F0701Y.

Table 17-1. Interrupt Source List (2/2)

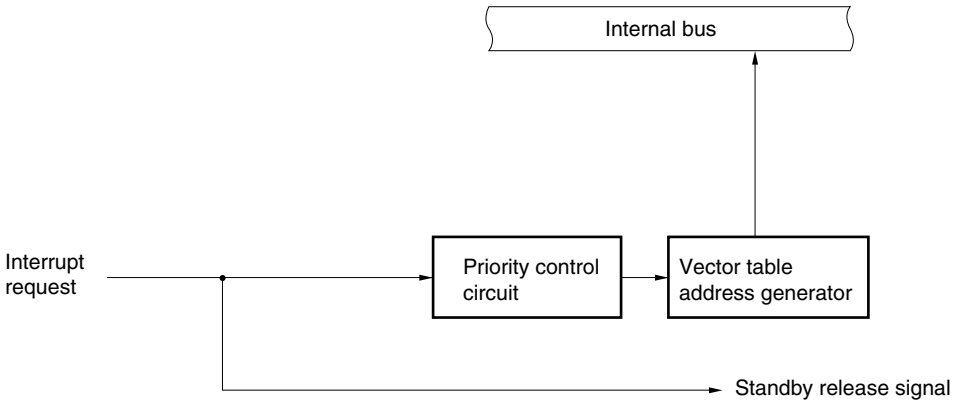
Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Maskable	21	INTTM001	Generation of match signal between TM01 and CR001 (when compare register is specified). Detection of TI001 valid edge (when capture register is specified.).	Internal	002EH	(B)
	22	INTTM011	Generation of match signal between TM01 and CR011 (when compare register is specified). Detection of TI011 valid edge (when capture register is specified.).		0030H	
	23	INTTM50	Generation of match signal between TM50 and CR50		0032H	
	24	INTTM51	Generation of match signal between TM51 and CR51		0034H	
	25	INTTM52	Generation of match signal between TM52 and CR52		0036H	
	26	INTAD	End of A/D converter conversion		0038H	
	27	INTWTN0	Watch timer overflow		003AH	
	28	INTKR	Detection of port 4 falling edge	External	003CH	(D)
Software	—	BRK	Execution of BRK Instruction	—	003EH	(E)

Notes 1. The default priority is the priority applicable when two or more maskable interrupts are generated simultaneously. 0 is the highest priority, and 28 is the lowest.

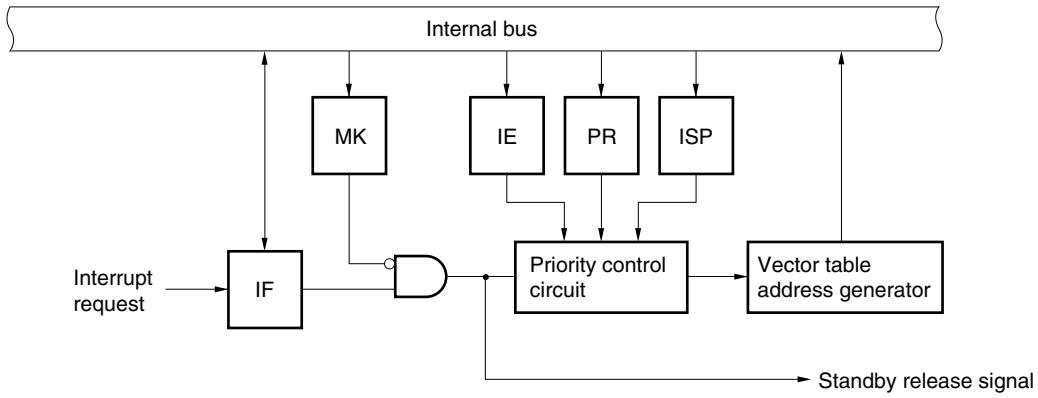
2. Basic configuration types (A) to (E) correspond to (A) to (E) in Figure 17-1.

Figure 17-1. Basic Configuration of Interrupt Function (1/2)

(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt (INTP0 to INTP7)

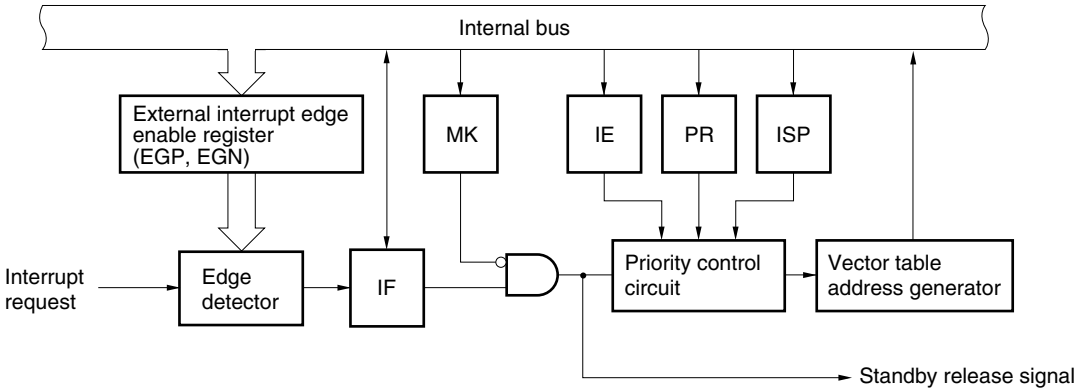
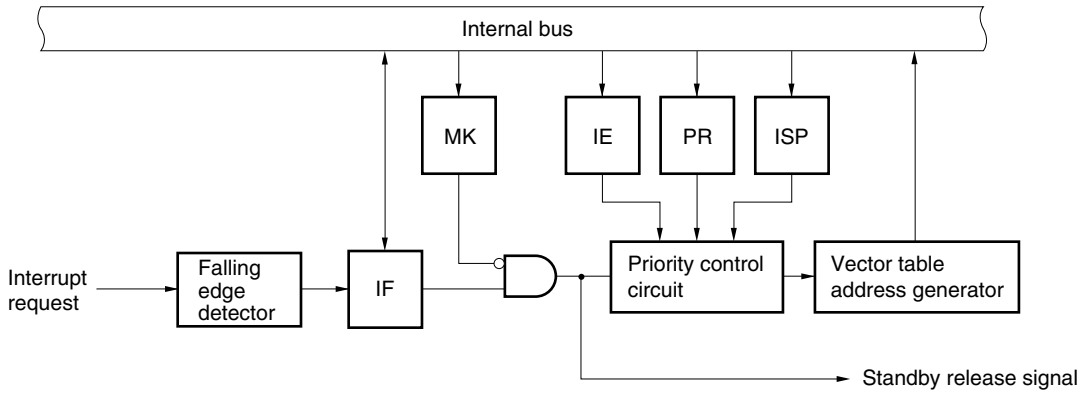
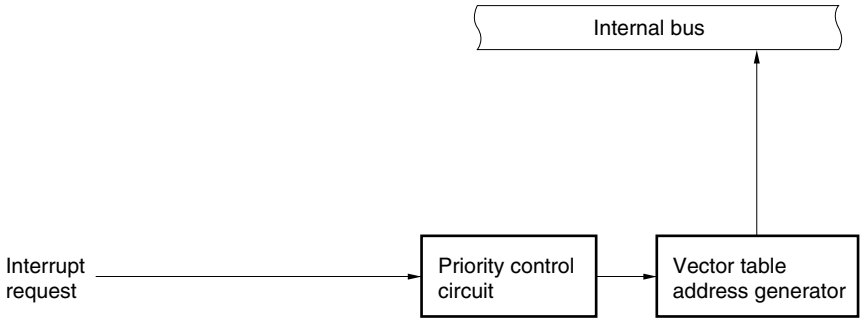


Figure 17-1. Basic Configuration of Interrupt Function (2/2)

(D) External maskable interrupt (INTKR)



(E) Software interrupt



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

17.3 Interrupt Function Control Registers

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

- Interrupt request flag register (IF0L, IF0H, IF1L, IF1H)
- Interrupt mask flag register (MK0L, MK0H, MK1L, MK1H)
- Priority specify flag register (PR0L, PR0H, PR1L, PR1H)
- External interrupt rising edge enable flag (EGP)
- External interrupt falling edge enable flag (EGN)
- Program status word (PSW)

Table 17-2 gives a list of interrupt request flags, interrupt mask flags, and priority specify flags corresponding to interrupt request sources.

Table 17-2. Flags Corresponding to Interrupt Request Sources

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specify Flag	
		Register		Register		Register
INTWDT	WDTIF[Note 1]	IF0L	WDTMK[Note 1]	MK0L	WDTPR[Note 1]	PR0L
INTP0	PIF0		PMK0		PPR0	
INTP1	PIF1		PMK1		PPR1	
INTP2	PIF2		PMK2		PPR2	
INTP3	PIF3		PMK3		PPR3	
INTP4	PIF4		PMK4		PPR4	
INTP5	PIF5		PMK5		PPR5	
INTP6	PIF6		PMK6		PPR6	
INTP7	PIF7	IF0H	PMK7	MK0H	PPR7	PR0H
INTSER0	SERIF0		SERMK0		SERPR0	
INTSR0	SRIF0		SRMK0		SRPR0	
INTST0	STIF0		STMK0		STPR0	
INTCSI30	CSIIF30		CSIMK30		CSIPR30	
INTCSI31	CSIIF31		CSIMK31		CSIPR31	
INTIIC0	IICIF0		IICMK0		IICPR0	
INTCE[Note 2]	CEIF		CEMK		CEPR	
INTCR[Note 3]	CRIF	IF1L	CRMK	MK1L	CRPR	PR1L
INTIE1[Note 4]	IEIF1		IEMK1		IEPR1	
INTCT[Note 3]	CTIF		CTMK		CTPR	
INTIE2[Note 4]	IEIF2		IEMK2		IEPR2	
INTWTNI0	WTNIIF0		WTNIMK0		WTNIPR0	
INTTM000	TMIF000		TMMK000		TMPR000	
INTTM010	TMIF010		TMMK010		TMPR010	
INTTM001	TMIF001		TMMK001		TMPR001	
INTTM011	TMIF011		TMMK011		TMPR011	
INTTM50	TMIF50		TMMK50		TMPR50	
INTTM51	TMIF51	IF1H	TMMK51	MK1H	TMPR51	PR1H
INTTM52	TMIF52		TMMK52		TMPR52	
INTAD	ADIF		ADMK		ADPR	
INTWTN0	WTNIF0		WTNMK0		WTNPR0	
INTKR	KRIF		KRMK		KRPR	

Notes 1. Interrupt control flag when the watchdog timer is used as an interval timer.

2. Applies to the μPD780701Y and 78F0701Y only. Not supported in the μPD780702Y.

3. Applies to the μPD780701Y and 78F0701Y only. Connected to another signal (shown in **Note 4**) in the μPD780702Y. Selected when the DCAN controller is used in the μPD78F0701Y.

4. Applies to the μPD780702Y and 78F0701Y only. Connected to another signal (shown in **Note 3**) in the μPD780701Y. Selected when the IEBus controller is used in the μPD78F0701Y.

(1) Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H)

The interrupt request flags are set to 1 when the corresponding interrupt request is generated or an instruction is executed. They are cleared to 0 when an instruction is executed upon acknowledgment of an interrupt request or upon application of $\overline{\text{RESET}}$ input.

IF0L, IF0H, IF1L, and IF1H are set with a 1-bit or 8-bit memory manipulation instruction. When IF0L/IF0H and IF1L/IF1H are combined to form 16-bit registers IF0 and IF1, they are read with a 16-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets these registers to 00H.

Figure 17-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IF0L	PIF6	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0	WDTIF	FFE0H	00H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IF0H	CEIF[Note]	IICIF0	CSIIF31	CSIIF30	STIF0	SRIF0	SERIF0	PIF7	FFE1H	00H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IF1L	TMIF50	TMIF011	TMIF001	TMIF010	TMIF000	WTNIIF0	CTIF (IEIF2)	CRIF (IEIF1)	FFE2H	00H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IF1H	0	0	0	KRIF	WTNIF0	ADIF	TMIF52	TMIF51	FFE3H	00H	R/W

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

Note Applies to the μPD780701Y and μPD78F0701Y only. Always set this bit to 0 in the μPD780702Y.

Cautions 1. The WDTIF flag is R/W enabled only when the watchdog timer is used as an interval timer. If watchdog timer mode 1 is used, set the WDTIF flag to 0.

2. Be sure to set bits 5 to 7 of IF1H to 0.

Remark Values in parentheses apply to the μPD780702Y or 78F0701Y when the IEBus controller is used.

(2) Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H)

The interrupt mask flags are used to enable/disable the corresponding maskable interrupt service and to set standby clear enable/disable.

MK0L, MK0H, MK1L, and MK1H are set with a 1-bit or 8-bit memory manipulation instruction. When MK0L/MK0H and MK1L/MK1H are combined to form 16-bit registers MK0 and MK1, they are set with a 16-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets these registers to FFH.

Figure 17-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MK0L	PMK6	PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	WDTMK	FFE4H	FFH	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MK0H	CEMK Note	IICMK0	CSIMK31	CSIMK30	STMK0	SRMK0	SERMK0	PMK7	FFE5H	FFH	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MK1L	TMMK50	TMMK011	TMMK001	TMMK010	TMMK000	WTNIMK0	CTMK (IEMK2)	CRMK (IEMK1)	FFE6H	FFH	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MK1H	1	1	1	KRMK	WTNMK0	ADMK	TMMK52	TMMK51	FFE7H	FFH	R/W

××MK×	Interrupt servicing control
0	Interrupt service enable
1	Interrupt service disable

Note Applies to the μPD780701Y and μPD78F0701Y only. Always set this bit to 1 in the μPD780702Y.

Cautions 1. If the watchdog timer is used in watchdog timer mode 1, the value of the WDTMK flag become undefined when read.

2. Because port 0 pins have an alternate function as external interrupt request input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set. Therefore, the interrupt mask flag should be set to 1 before using the output mode.

3. Be sure to set bits 5 to 7 of MK1H to 1.

Remark Values in parentheses apply to the μPD780702Y or 78F0701Y when the IEBus controller is used.

(3) Priority specify flag registers (PR0L, PR0H, PR1L, PR1H)

The priority specify flags are used to set the corresponding maskable interrupt priority orders.

PR0L, PR0H, PR1L, and PR1H are set with a 1-bit or 8-bit memory manipulation instruction. If PR0L/PR0H and PR1L/PR1H are combined to form 16-bit registers PR0 and PR1, they are set with a 16-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets these registers to FFH.

Figure 17-4. Format of Interrupt Priority Specify Flag Registers (PR0L, PR0H, PR1L, PR1H)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PR0L	PPR6	PPR5	PPR4	PPR3	PPR2	PPR1	PPR0	WDTPR	FFE8H	FFH	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PR0H	CEPR[Note]	IICPR0	CSIPR31	CSIPR30	STPR0	SRPR0	SERPR0	PPR7	FFE9H	FFH	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PR1L	TMPR50	TMPR011	TMPR001	TMPR010	TMPR000	WTNIPR0	CTPR (IEPR2)	CRPR (IEPR1)	FFEAH	FFH	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PR1H	1	1	1	KRPR	WTNPR0	ADPR	TMPR52	TMPR51	FFEBH	FFH	R/W

××PR×	Interrupt priority level selection
0	High priority level
1	Low priority level

Note Applies to the μPD780701Y and μPD78F0701Y only. Always set this bit to 1 in the μPD780702Y.

Cautions 1. When the watchdog timer is used in the watchdog timer 1 mode, set the WDTPR flag to 1.

2. Be sure to set bits 5 to 7 of PR1H to 1.

Remark Values in parentheses apply to the μPD780702Y or 78F0701Y when the IEBus controller is used.

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

These registers specify the valid edge for INTP0 to INTP7.

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

$\overline{\text{RESET}}$ input sets these registers to 00H.

Figure 17-5. Format of External Interrupt Rising Edge Enable Register (EGP), External Interrupt Falling Edge Enable Register (EGN)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
EGP	EGP7	EGP6	EGP5	EGP4	EGP3	EGP2	EGP1	EGP0	FF48H	00H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
EGN	EGN7	EGN6	EGN5	EGN4	EGN3	EGN2	EGN1	EGN0	FF49H	00H	R/W

EGPn	EGNn	INTPn pin valid edge selection (n = 0 to 7)
0	0	Interrupt disable
0	1	Falling edge
1	0	Rising edge
1	1	Both rising and falling edges

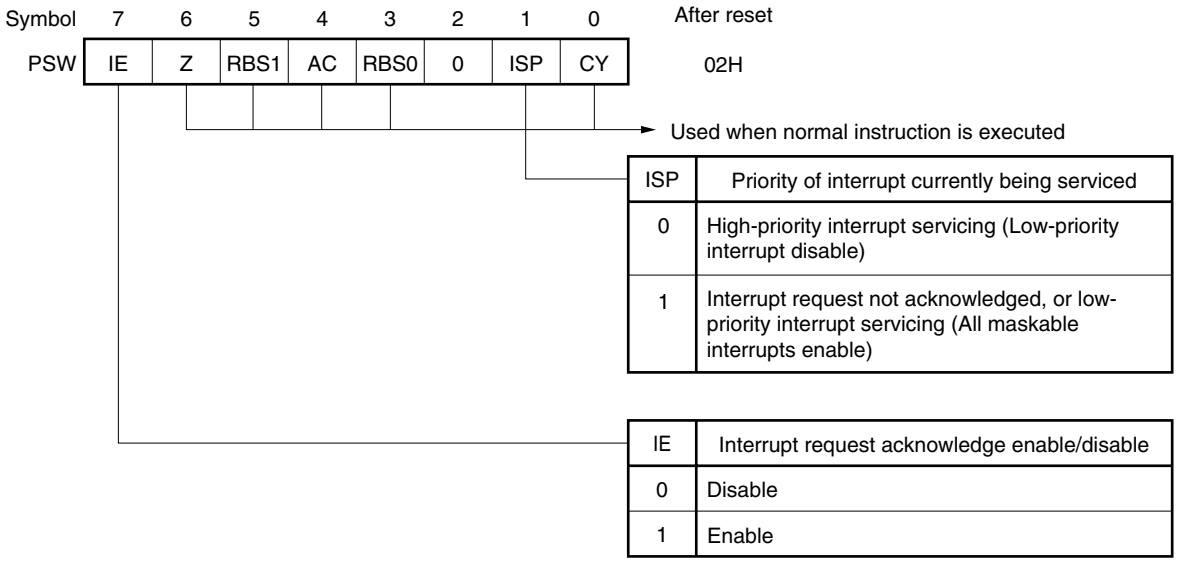
(5) Program status word (PSW)

Program status word is a register to hold the instruction execution result and the current status for an interrupt request. The IE flag to set maskable interrupt enable/disable and the ISP flag to control multiple processing are mapped.

Besides 8-bit read/write, this register can carry out operations with bit manipulation instructions and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged or when the BRK instruction is executed, the contents of PSW are automatically saved into a stack and the IE flag is reset to 0. If a maskable interrupt request is acknowledged, the contents of the priority specify flag of the acknowledged interrupt are transferred to the ISP flag. The PSW contents are also saved into the stack with the PUSH PSW instruction. They are reset from the stack with the RETI, RETB, and POP PSW instructions.

$\overline{\text{RESET}}$ input sets PSW to 02H.

Figure 17-6. Format of Program Status Word



ISP	Priority of interrupt currently being serviced
0	High-priority interrupt servicing (Low-priority interrupt disable)
1	Interrupt request not acknowledged, or low-priority interrupt servicing (All maskable interrupts enable)

IE	Interrupt request acknowledge enable/disable
0	Disable
1	Enable

17.4 Interrupt Servicing Operations

17.4.1 Non-maskable interrupt request acknowledge operation

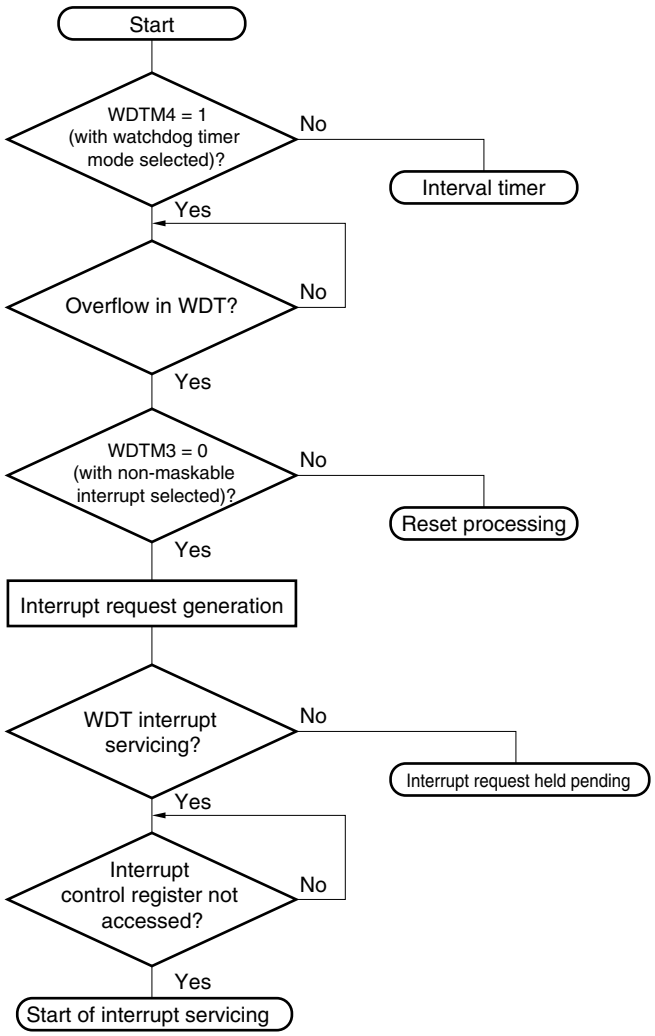
A non-maskable interrupt request is unconditionally acknowledged even if in an interrupt acknowledge disable state. It is not subject to interrupt priority control and has highest priority over all other interrupts.

If a non-maskable interrupt request is acknowledged, the contents of program status word (PSW) and program counter (PC) are saved into the stacks in the order of PSW, program counter (PC). The IE and ISP flags are then reset to 0, and the contents of the vector table are loaded into PC to let the program branch.

A new non-maskable interrupt request generated during execution of a non-maskable interrupt servicing program is acknowledged after the current execution of the non-maskable interrupt servicing program is terminated (following RETI instruction execution) and one main routine instruction is executed. However, if a new non-maskable interrupt request is generated twice or more during non-maskable interrupt servicing program execution, only one non-maskable interrupt request is acknowledged after termination of the non-maskable interrupt servicing program execution.

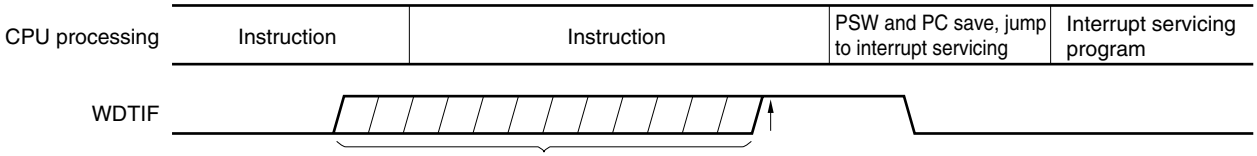
Figures 17-7, 17-8, and 17-9 show the flowchart of the non-maskable interrupt request generation through acknowledge, acknowledge timing of non-maskable interrupt request, and acknowledge operation at multiple non-maskable interrupt request generation, respectively.

Figure 17-7. Flowchart of Non-Maskable Interrupt Request Generation to Acknowledge



WDTM: Watchdog timer mode register
WDT: Watchdog timer

Figure 17-8. Timing of Non-Maskable Interrupt Request Acknowledgement

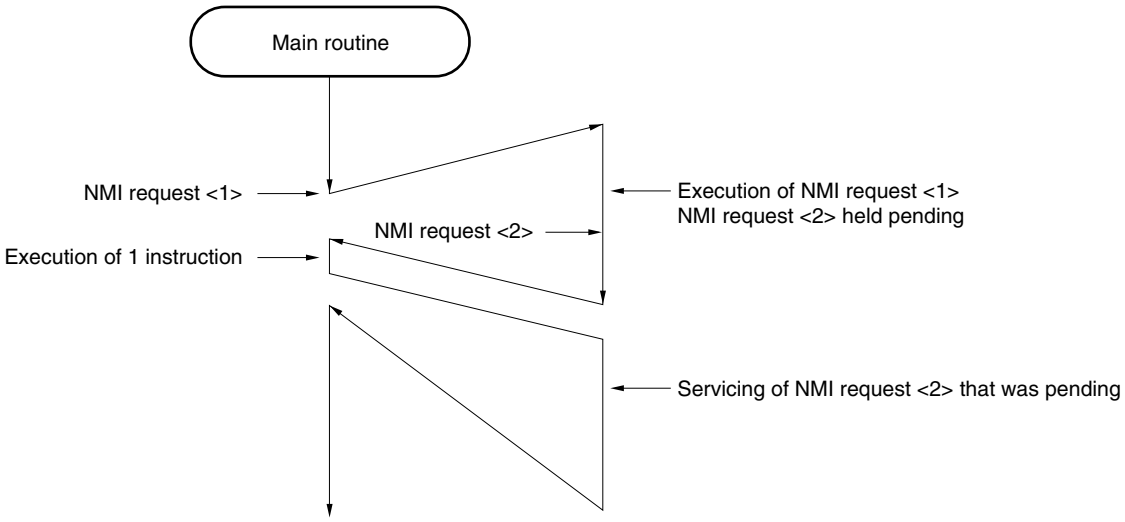


Interrupt request generated during this interval is acknowledged at ↑.

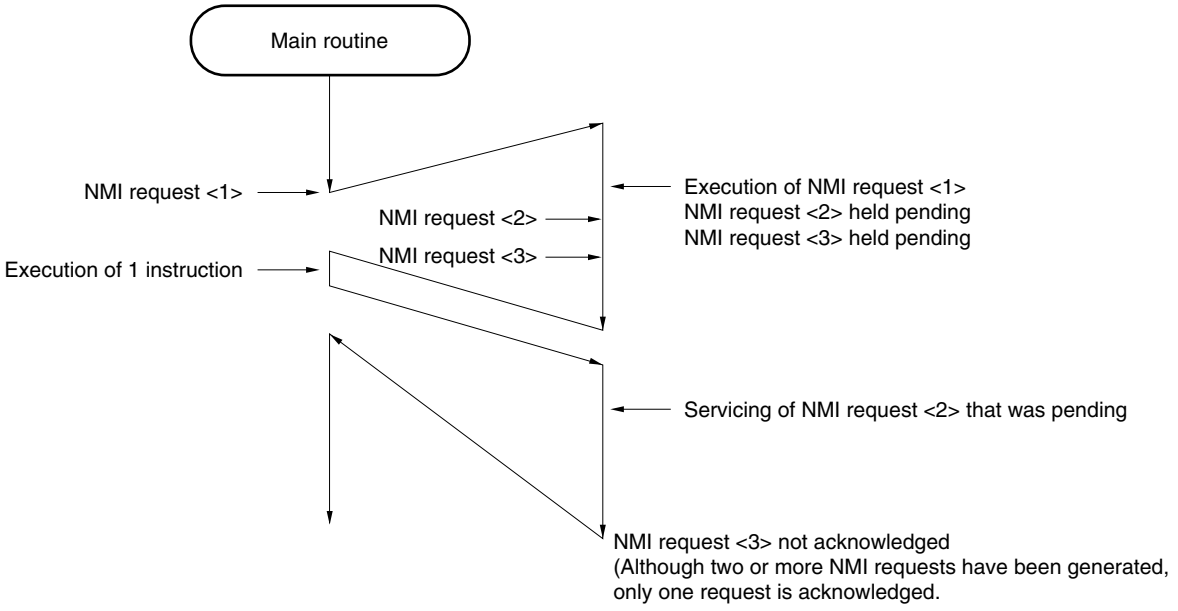
WDTIF: Watchdog timer interrupt request flag

Figure 17-9. Operation of Non-Maskable Interrupt Request Acknowledgement

(a) If another non-maskable interrupt request is generated during non-maskable interrupt servicing program execution



(b) If two non-maskable interrupt requests are generated during non-maskable interrupt servicing program execution



17.4.2 Maskable interrupt acknowledge operation

A maskable interrupt becomes acknowledgeable when the interrupt request flag is set to 1 and the mask (MK) flag corresponding to that interrupt request is cleared to 0. A vectored interrupt request is acknowledged if in the interrupt enable state (when IE flag is set to 1). However, a low-priority interrupt request is not acknowledged during servicing of a higher priority interrupt request (when the ISP flag is reset to 0).

The times from generation of a maskable interrupt request until interrupt servicing is performed are listed in Table 17-3 below.

For the interrupt request acknowledge timing, see Figures 17-11 and 17-12.

Table 17-3. Times from Generation of Maskable Interrupt until Servicing

	Minimum Time	Maximum Time[Note]
When ××PR = 0	7 clocks	32 clocks
When ××PR = 1	8 clocks	33 clocks

Note If an interrupt request is generated just before a divide instruction, the wait time becomes longer.

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

If two or more interrupt requests are generated simultaneously, the request with a higher priority level specified by the priority specify flag is acknowledged first. If two or more interrupt requests have the same priority level, the request with the highest default priority is acknowledged first.

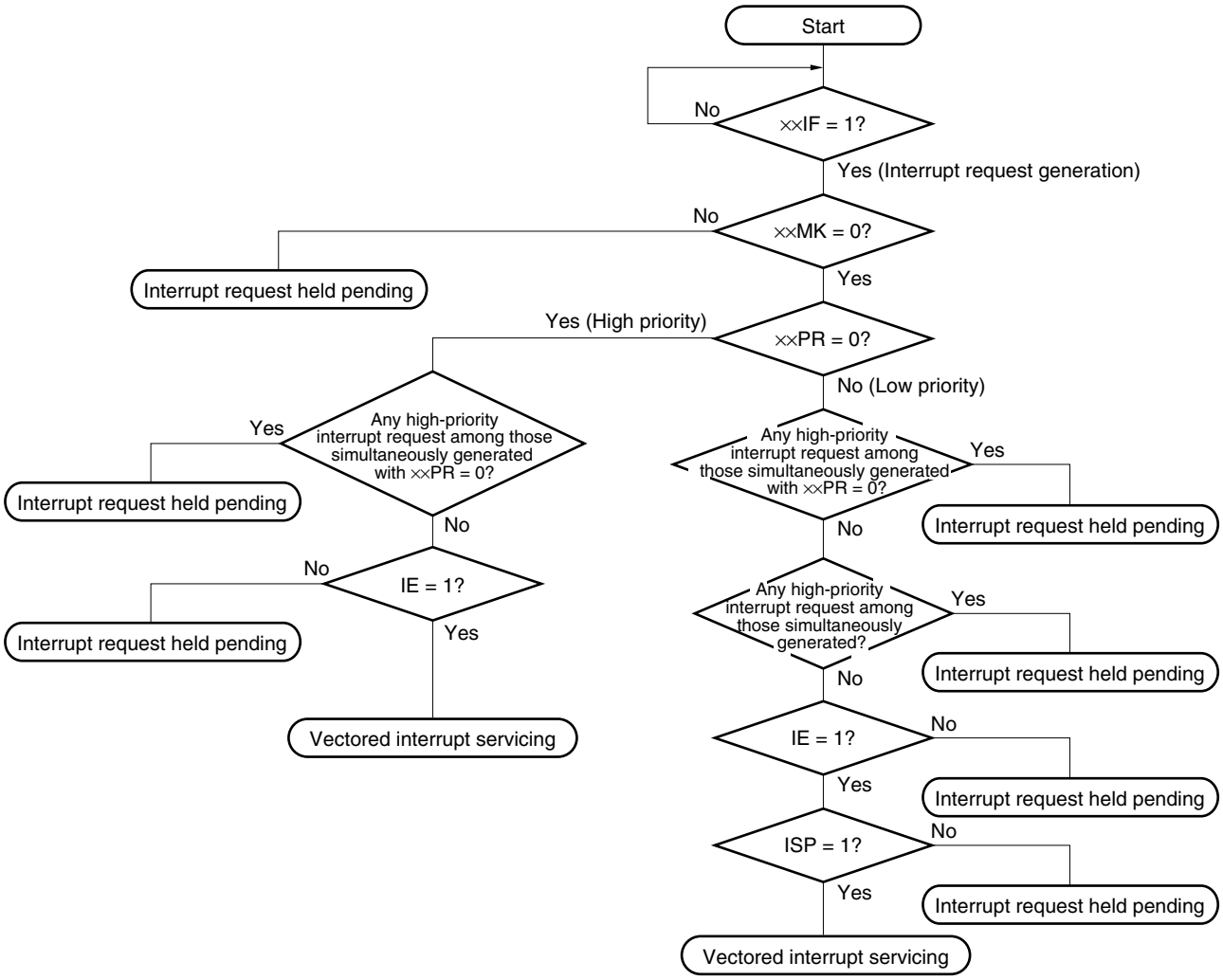
An interrupt request that is held pending is acknowledged when it becomes acknowledgeable.

Figure 17-10 shows the interrupt request acknowledge algorithm.

If a maskable interrupt request is acknowledged, the contents of PSW and PC are saved into the stacks in the order of PSW, PC. The IE flag is then reset to 0, and the contents of the priority specify flag corresponding to the acknowledged interrupt request are transferred to the ISP flag. Further, the vector table data determined for each interrupt request is loaded into PC to let the program branch.

Return from an interrupt is possible with the RETI instruction.

Figure 17-10. Interrupt Request Acknowledge Processing Algorithm



××IF: Interrupt request flag

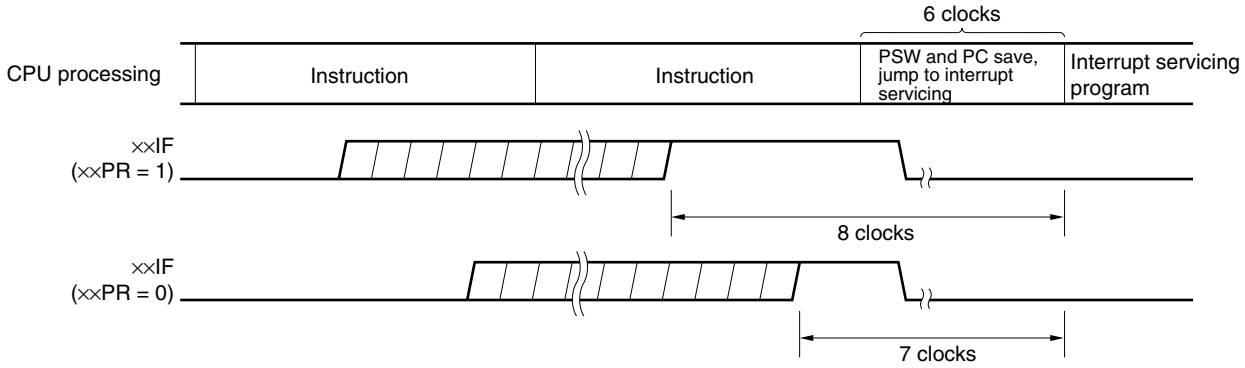
××MK: Interrupt mask flag

××PR: Priority specify flag

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

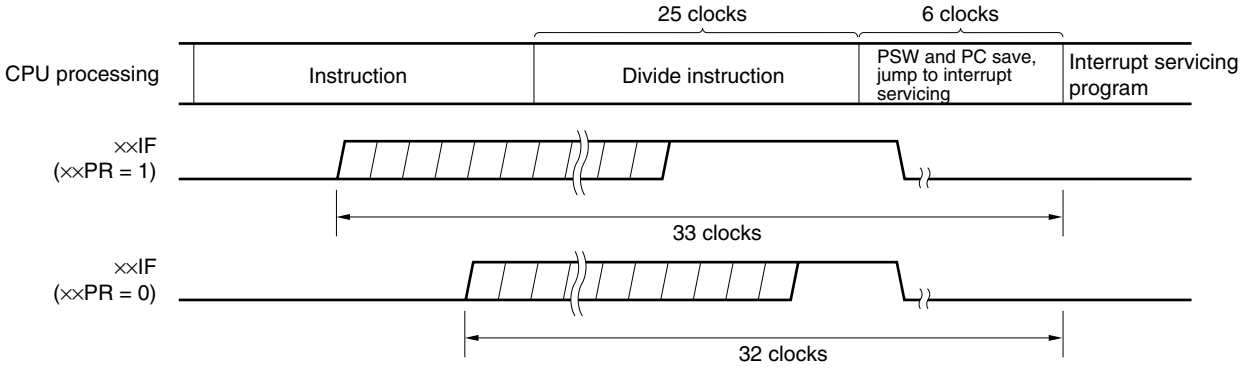
ISP: Flag that indicates the priority level of the interrupt currently being serviced (0 = high-priority interrupt servicing, 1 = No interrupt request received, or low-priority interrupt servicing)

Figure 17-11. Timing of Interrupt Request Acknowledgement (Minimum Time)



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

Figure 17-12. Timing of Interrupt Request Acknowledgement (Maximum Time)



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

17.4.3 Software interrupt request acknowledge operation

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

If a software interrupt request is acknowledged, the contents of PSW and PC are saved into the stacks in the order of PSW, PC. The IE flag is then reset to 0, and the contents of the vector table (003EH, 003FH) are loaded into PC to let the program branch.

Return from a software interrupt is possible with the RETB instruction.

Caution Do not use the RETI instruction for returning from the software interrupt.

17.4.4 Multiple interrupt servicing

Multiple interrupts occur when another interrupt request is acknowledged during execution of an interrupt.

Multiple interrupts do not occur unless the interrupt request acknowledge enable state is selected (IE = 1) (except non-maskable interrupts). Also, when an interrupt request is received, interrupt request acknowledge becomes disabled (IE = 0). Therefore, to enable multiple interrupts, it is necessary to set the IE flag to 1 with the EI instruction during interrupt servicing to enable interrupt acknowledge.

Moreover, even in the interrupt enable status, multiple interrupts may not be enabled, this being subject to interrupt priority control. Two types of priority control are available: default priority control and programmable priority control. Programmable priority control is used for multiple interrupts.

In the interrupt enable status, if an interrupt request with a priority equal to or higher than that of the interrupt currently being serviced is generated, it is acknowledged for multiple interrupt servicing. If an interrupt with a priority lower than that of the interrupt currently being serviced is generated during interrupt servicing, it is not acknowledged.

Interrupt requests that are not enabled because of the interrupt disable state or they have a lower priority are held pending. When servicing of the current interrupt ends, the ended interrupt request is acknowledged following execution of one main processing instruction execution.

Multiple interrupt servicing is not possible during non-maskable interrupt servicing.

Table 17-4 shows interrupt requests enabled for multiple interrupt servicing, and Figure 17-13 shows multiple interrupt examples.

Table 17-4. Interrupt Request Capable of Multiple Interrupt during Interrupt Servicing

Multiple Interrupt Request / Interrupt being Serviced		Non-Maskable Interrupt Request	Maskable Interrupt Request			
			××PR = 0		××PR = 1	
			IE = 1	IE = 0	IE = 1	IE = 0
Non-maskable interrupt		N/A	N/A	N/A	N/A	N/A
Maskable interrupt	ISP = 0	A	A	N/A	N/A	N/A
	ISP = 1	A	A	N/A	A	N/A
Software interrupt		A	A	N/A	A	N/A

Remarks 1. A: Multiple interrupt enable (available)

2. N/A: Multiple interrupt disable (not available)

3. ISP and IE are flags contained in PSW.

ISP = 0: An interrupt with higher priority is being serviced.

ISP = 1: No interrupt request has been acknowledged, or an interrupt with a lower priority is being serviced.

IE = 0: Interrupt request acknowledge is disabled.

IE = 1: Interrupt request acknowledge is enabled.

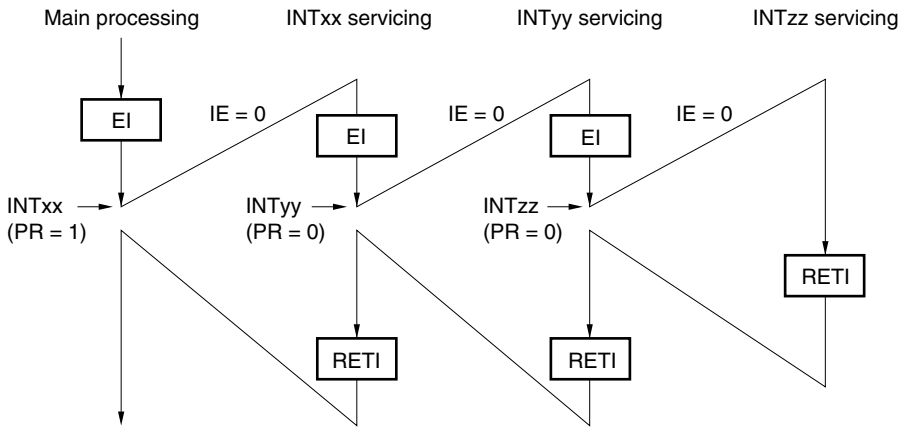
4. ××PR is a flag contained in PR0L, PR0H, PR1L, and PR1H.

××PR = 0: Higher priority level

××PR = 1: Lower priority level

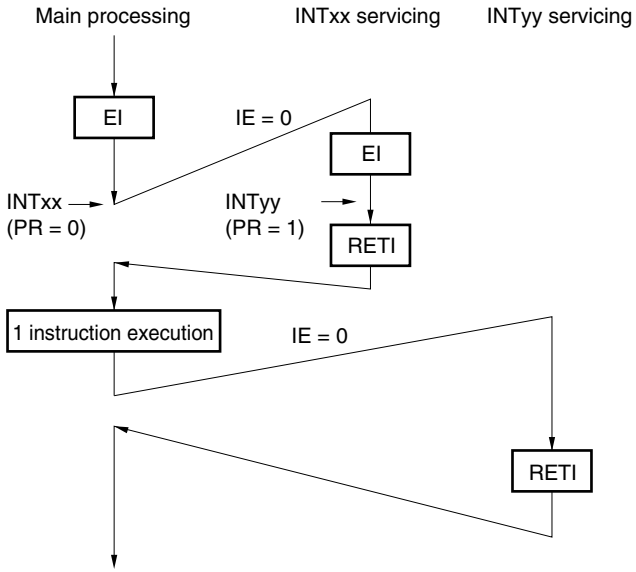
Figure 17-13. Multiple Interrupt Examples (1/2)

Example 1. Multiple interrupts occur twice



During servicing of interrupt INTxx, two interrupt requests, INTyy and INTzz, are acknowledged, and multiple interrupt servicing takes place. Before each interrupt request is acknowledged, the EI instruction must always be issued to enable interrupt request acknowledge.

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

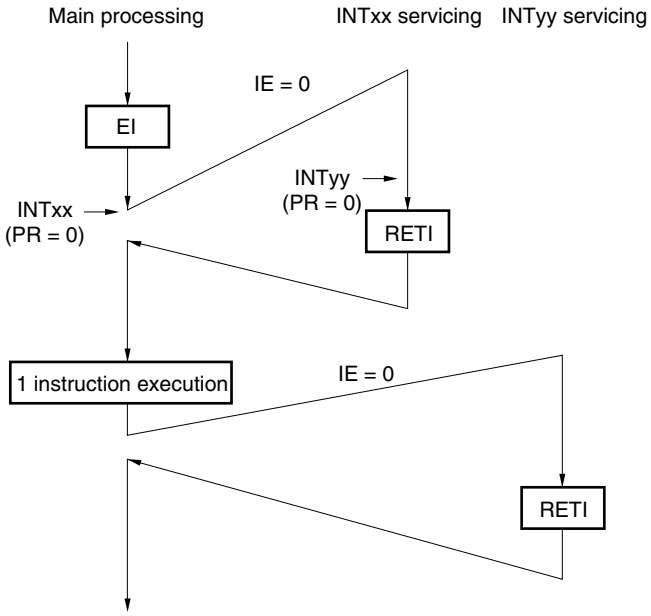


Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 0: Higher priority level
PR = 1: Lower priority level
IE = 0: Interrupt request acknowledge disable

Figure 17-13. Multiple Interrupt Examples (2/2)

Example 3. Multiple interrupt servicing does not occur because interrupt is not enabled



Interrupt is not enabled during servicing of interrupt INTxx (EI instruction is not issued), therefore interrupt request INTyy is not acknowledged and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 0: Higher priority level
IE = 0: Interrupt request acknowledge disabled

17.4.5 Interrupt request hold

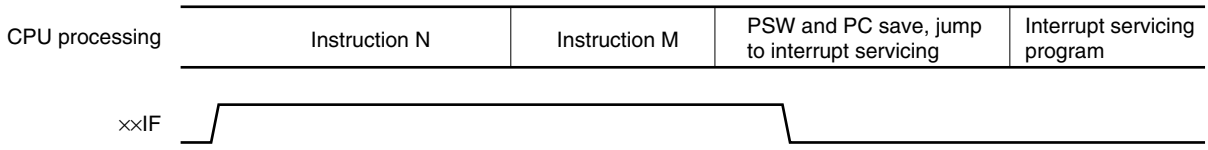
There are instructions where, even if an interrupt request is issued for them while another instruction is executed, request acknowledge is held pending until the end of execution of the next instruction. These instructions (interrupt request hold instructions) are listed below.

- MOV PSW, #byte
- MOV A, PSW
- MOV PSW, A
- MOV1 PSW. bit, CY
- MOV1 CY, PSW. bit
- AND1 CY, PSW. bit
- OR1 CY, PSW. bit
- XOR1 CY, PSW. bit
- SET1 PSW. bit
- CLR1 PSW. bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT PSW. bit, $addr16
- BF PSW. bit, $addr16
- BTCLR PSW. bit, $addr16
- EI
- DI
- Manipulate instructions for the IF0L, IF0H, IF1L, IF1H, MK0L, MK0H, MK1L, MK1H, PR0L, PR0H, PR1L, PR1H, EGP, and EGN registers.

Caution The BRK instruction is not one of the above listed interrupt request hold instructions. However, the software interrupt activated by executing the BRK instruction causes the IE flag to be cleared. Therefore, even if a maskable interrupt requests is generated during execution of the BRK instruction, the interrupt request is not acknowledged. However, a non-maskable interrupt request is acknowledged.

Figure 17-14 shows the timing with which interrupt requests are held pending.

Figure 17-14. Interrupt Request Hold



Remarks 1. Instruction N: Interrupt request hold instruction

2. Instruction M: Instruction other than interrupt request hold instruction

3. The ××PR (priority level) values do not affect the operation of ××IF (instruction request)

CHAPTER 18 STANDBY FUNCTION

18.1 Standby Function and Configuration

18.1.1 Standby function

The standby function is designed to reduce the power consumption of the system. The following two modes are available.

(1) HALT mode

HALT instruction execution sets the HALT mode. The HALT mode is intended to stop the CPU operation clock. The system clock oscillator continues oscillating. In this mode, current consumption is not reduced as much as in the STOP mode. However, the HALT mode is effective for restarting operation immediately upon interrupt request and for carrying out intermittent operations such as watch applications.

(2) STOP mode

STOP instruction execution sets the STOP mode. In the STOP mode, the system clock oscillator stops, stopping the whole system, thereby considerably reducing the CPU power consumption.

Data memory low-voltage hold (down to V_{DD} = 2.0 V) is possible. Thus, the STOP mode is effective for holding data memory contents with ultra-low current consumption.

Because this mode can be cleared upon interrupt request, it enables intermittent operations to be carried out. However, because a wait time is required to secure an oscillation stabilization time after the STOP mode is cleared, select the HALT mode if it is necessary to start processing immediately upon interrupt request.

In either of these two modes, all the contents of registers, flags, and data memory just before the standby mode is set are held. The input/output port output latch and output buffer statuses are also held.

Cautions 1. When operation is transferred to the STOP mode, be sure to stop the peripheral hardware operation before executing the STOP instruction.

2. The following sequence is recommended for power consumption reduction of the A/D converter when the standby function is used: First clear bit 7 (ADCS3) of the A/D converter mode register 3 (ADM3) to 0 to stop the A/D conversion operation, and then execute the HALT or STOP instruction.

18.1.2 Standby function control register

The wait time from when the STOP mode is released by an interrupt request to when the oscillation stables is controlled with the oscillation stabilization time select register (OSTS).

OSTS is set with an 8-bit memory manipulation instruction.

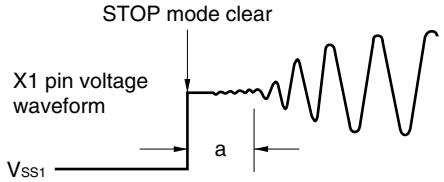
$\overline{\text{RESET}}$ input sets OSTS to 04H. Accordingly, the time required to cancel the STOP mode by means of the $\overline{\text{RESET}}$ input is 2^{17}/fx.

Figure 18-1. Format of Oscillation Stabilization Time Select Register (OSTS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time after STOP mode release
0	0	0	2^{12}/fx (651 μs)
0	0	1	2^{14}/fx (2.60 ms)
0	1	0	2^{15}/fx (5.21 ms)
0	1	1	2^{16}/fx (10.4 ms)
1	0	0	2^{17}/fx (20.8 ms)
Other than above			Setting prohibited

Caution The wait time after the STOP mode is cleared does not include the time (see "a" in the illustration below) from STOP mode clear to clock oscillation start. The time is not included either by $\overline{\text{RESET}}$ input or by interrupt request generation.



Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

18.2 Standby Function Operations

18.2.1 HALT mode

(1) HALT mode setting and operating statuses

The HALT mode is set by executing the HALT instruction.

The operating statuses in the HALT mode are described below.

Table 18-1. HALT Mode Operating Statuses

Item	Operating Status
Clock generator	Can be oscillated. Clock supply to CPU stops.
CPU	Operation stops
Port (Output latch)	Holds the state it was in just before the HALT instruction was executed.
16-bit timer/event counter	Operable
8-bit timer/event counter	
Watch timer	
Watchdog timer	
Clock output/Buzzer output control circuit	
A/D converter	Operation stops (only power-fail operation mode operable)
Serial interface	Operable
External interrupt	
DCAN controller[Note 1]	Operation stops
IEBus controller[Note 2]	Operable

Notes 1. Applies to the μPD780701Y and 78F0701Y only.

2. Applies to the μPD780702Y and 78F0701Y only.

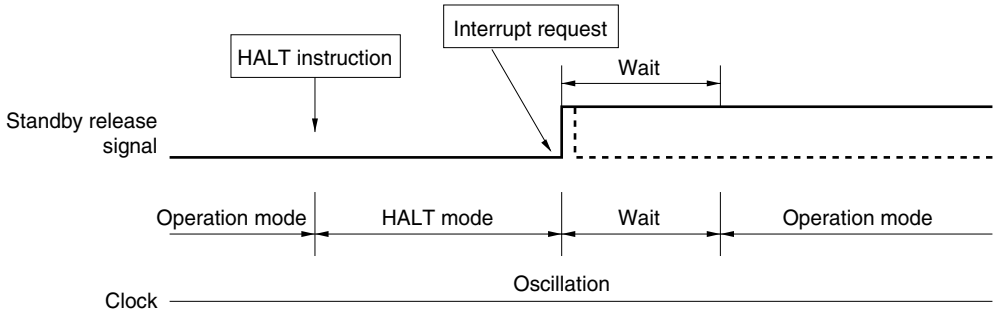
(2) HALT mode release

The HALT mode can be released with the following three types of sources.

(a) Clear upon unmasked interrupt request

When an unmasked interrupt request is generated, the HALT mode is released. If interrupt acknowledge is enabled, vectored interrupt servicing is carried out. If interrupt acknowledge is disabled, the next address instruction is executed.

Figure 18-2. HALT Mode Release by Interrupt Request Generation



Remarks 1. The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

2. Wait times are as follows:

- When vectored interrupt service is carried out: 8 or 9 clocks
- When vectored interrupt service is not carried out: 2 or 3 clocks

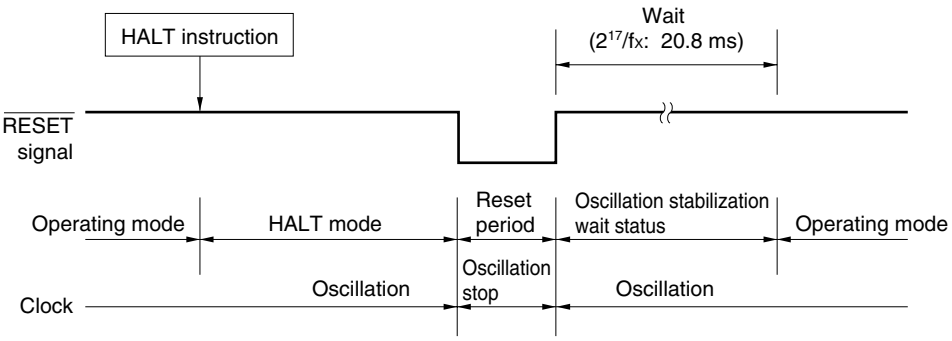
(b) Clear upon non-maskable interrupt request

When a non-maskable interrupt request is generated, the HALT mode is released and vectored interrupt servicing is carried out whether interrupt acknowledge is enabled or disabled.

(c) Release by $\overline{\text{RESET}}$ input

When the $\overline{\text{RESET}}$ signal is input, HALT mode is released, and, as with a normal reset operation, a program is executed after branching to the reset vector address.

Figure 18-3. HALT Mode Release by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

Table 18-2. Operation after HALT Mode Release

Release Source	MK××	PR××	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt servicing execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt servicing execution
	1	×	×	×	HALT mode hold
Non-maskable interrupt request	—	—	×	×	Interrupt servicing execution
$\overline{\text{RESET}}$ input	—	—	×	×	Reset processing

×: don't care

18.2.2 STOP mode

(1) STOP mode setting and operating statuses

The STOP mode is set by executing the STOP instruction.

Cautions 1. When the STOP mode is set, the X2 pin is internally connected to V_{DD1} via a pull-up resistor to minimize the leakage current at the crystal oscillator. Therefore, do not use the STOP mode in a system where an external clock is used for the system clock.

2. Because the interrupt request signal is used to clear the standby mode, if there is an interrupt source with the interrupt request flag set and the interrupt mask flag reset, the standby mode is immediately cleared if set. Thus, the STOP mode is reset to the HALT mode immediately after execution of the STOP instruction. After the wait is set using the oscillation stabilization time select register (OSTS), the operating mode is resumed.

The operating statuses in the STOP mode are described in Table 18-3 below.

Table 18-3. STOP Mode Operating Statuses

Item	Operating Status
Clock generator	Oscillation stops
CPU	Operation stops
Output port (Output latch)	Holds the state it was in just before the STOP instruction was executed.
★ 16-bit timer/event counter	Operation stops
★ 8-bit timer/event counter	
Watch timer	
Watchdog timer	
Clock output/Buzzer output control circuit	
A/D converter	
★ Serial interface	
External interrupt	Operable
DCAN controller[Note 1]	Operation stops
IEBus controller[Note 2]	

Notes 1. Applies to the μPD780701Y and 78F0701Y only.

2. Applies to the μPD780702Y and 78F0701Y only.

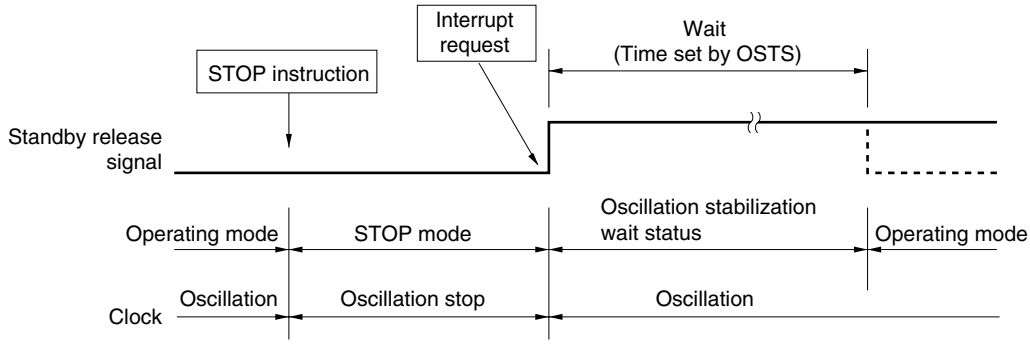
(2) STOP mode release

The STOP mode can be released by the following two types of sources.

(a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the STOP mode is released. If interrupt acknowledge is enabled after the oscillation stabilization time has elapsed, vectored interrupt servicing is carried out. If interrupt acknowledge is disabled, the next address instruction is executed.

Figure 18-4. STOP Mode Release by Interrupt Request Generation

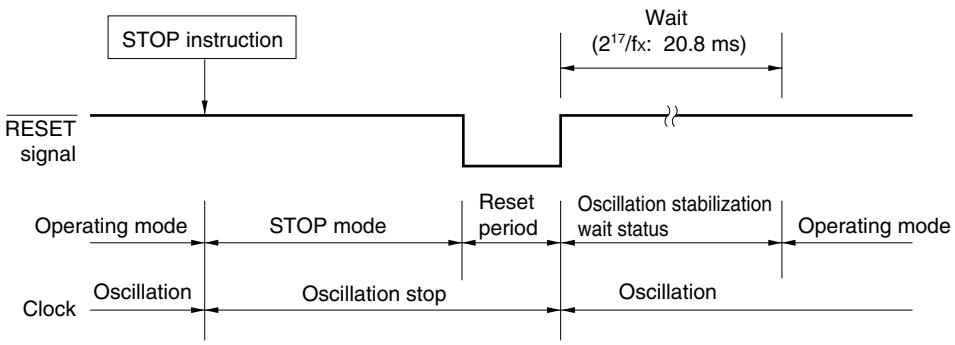


Remark The broken lines indicate the case when the interrupt request that has released the standby status is acknowledged.

(b) Release by $\overline{\text{RESET}}$ input

The STOP mode is released when the $\overline{\text{RESET}}$ signal is input, and after the oscillation stabilization time has elapsed, reset operation is carried out.

Figure 18-5. STOP Mode Release by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 6.29 MHz.

Table 18-4. Operation after STOP Mode Release

Release Source	MK××	PR××	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt servicing execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt servicing execution
	1	×	×	×	STOP mode hold
$\overline{\text{RESET}}$ input	—	—	×	×	Reset processing

×: don't care

CHAPTER 19 RESET FUNCTION

The following two operations are available to generate the reset function.

(1) External reset input via $\overline{\text{RESET}}$ pin
(2) Internal reset by watchdog timer runaway time detection

External reset and internal reset have no functional differences. In both cases, program execution starts at the addresses at 0000H and 0001H by means of $\overline{\text{RESET}}$ input.

When a low level is input to the $\overline{\text{RESET}}$ pin or the watchdog timer overflows, a reset is applied and each hardware is set to the status shown in Table 19-1. Each pin enters high-impedance state during reset input or during oscillation stabilization time just after reset cancellation.

When a high level is input to the $\overline{\text{RESET}}$ pin, the reset is cancelled and program execution starts after the oscillation stabilization time $2^{17}/f_X$ has elapsed. The reset applied by watchdog timer overflow is automatically cancelled after a reset and program execution starts after the oscillation stabilization time $2^{17}/f_X$ has elapsed (see **Figures 19-2** to **19-4**).

Cautions 1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.
2. System clock operation stops during reset input.
3. When the STOP mode is released by reset, the STOP mode contents are held during reset input. However, the port pin becomes high-impedance.

Figure 19-1. Block Diagram of Reset Function

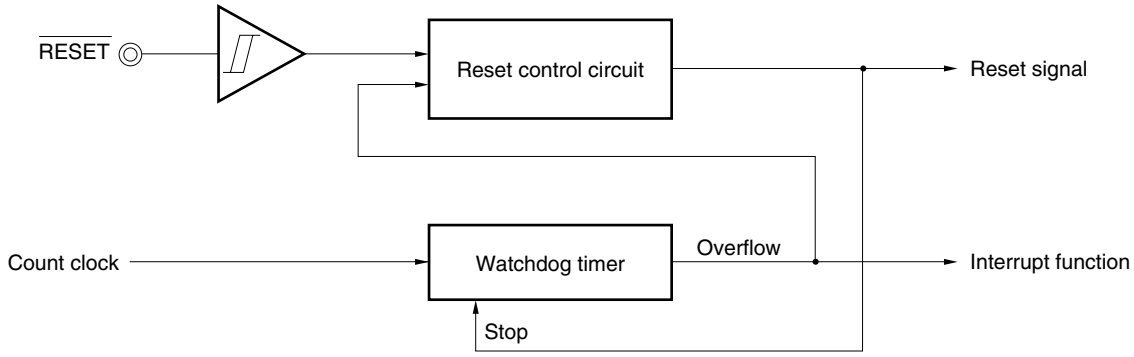


Figure 19-2. Timing of Reset by $\overline{\text{RESET}}$ Input

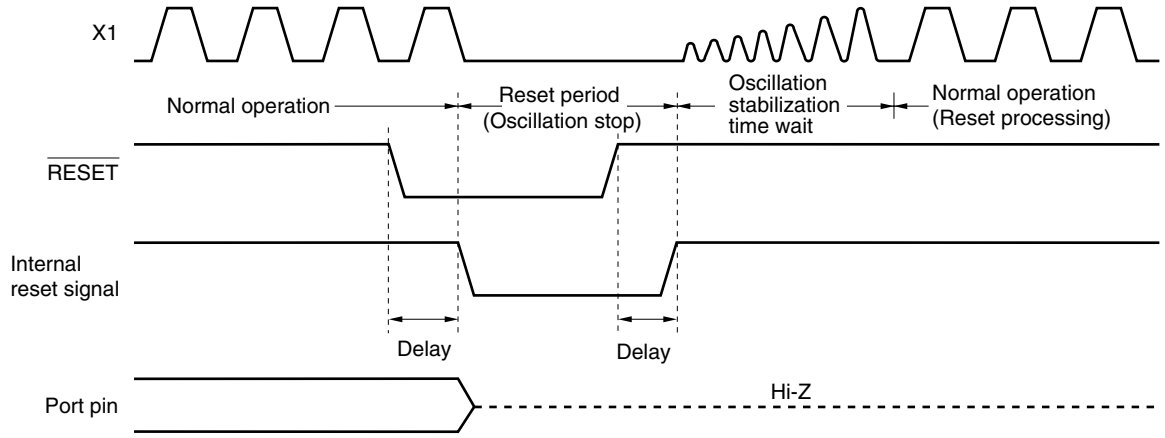


Figure 19-3. Timing of Reset due to Watchdog Timer Overflow

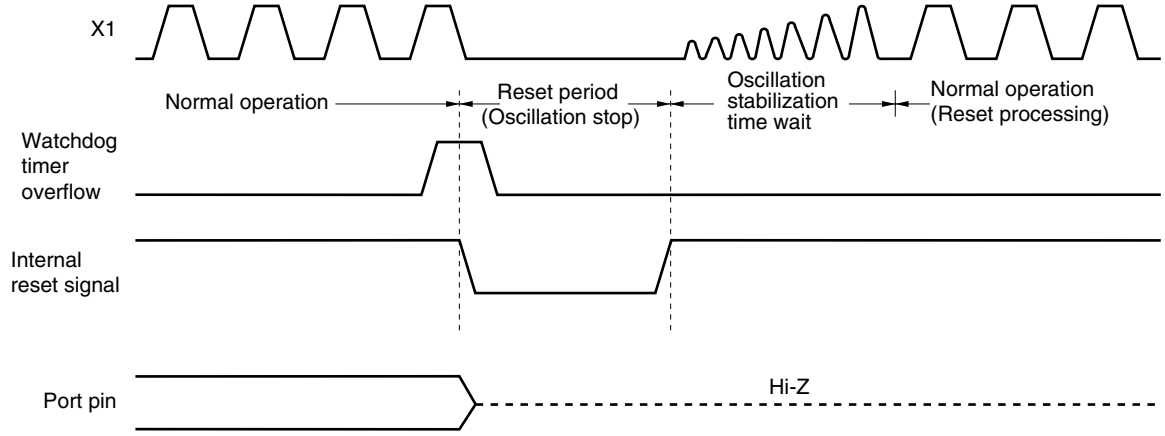


Figure 19-4. Timing of Reset in STOP Mode by $\overline{\text{RESET}}$ Input

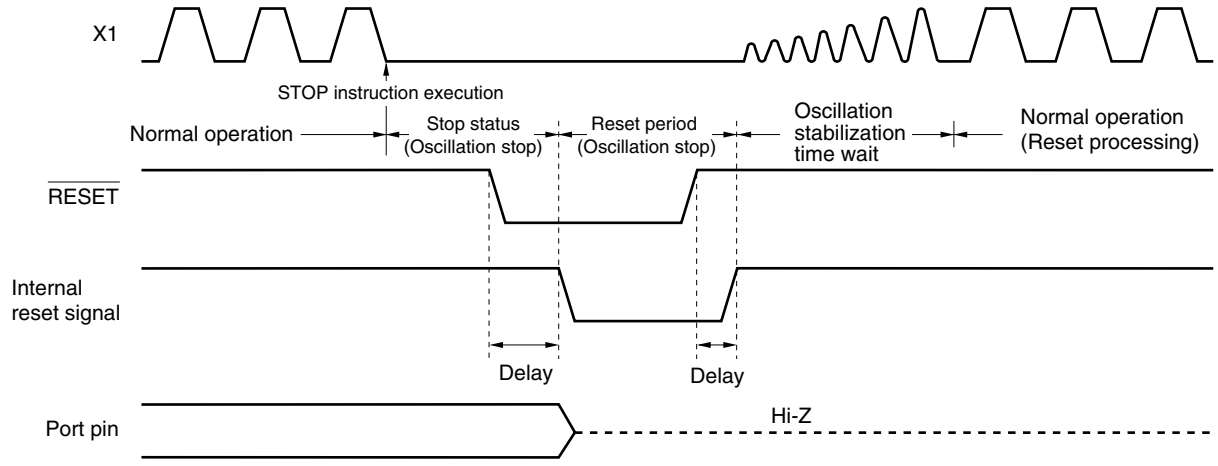


Table 19-1. Hardware Statuses after Reset (1/3)

Hardware		Status After Reset
Program counter (PC)[Note 1]		Contents of reset vector table (0000H, 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined[Note 2]
	General register	Undefined[Note 2]
Port (Output latch)	Ports 0, 2, 3, 7 to 9 (P0, P2, P3, P7 to P9)	00H
	Ports 4 to 6 (P4 to P6)	Undefined
Port mode registers (PM0, PM2 to PM9)		FFH
Pull-up resistor option registers (PU0, PU2 to PU7)		00H
Processor clock control register (PCC)		04H
Memory expansion mode register (MEM)		00H
Internal memory size switching register (IMS)		CFH[Note 3]
Internal expansion RAM size switching register (IXS)		0CH[Note 4]
Flash programming mode control register (FLPMC)		08H[Note 5]
Oscillation stabilization time selection register (OSTS)		04H
16-bit timer/event counter	Timer counters (TM00, TM01)	0000H
★	Capture/compare registers (CR000, CR010, CR001, CR011)	0000H
	Prescaler mode registers (PRM00, PRM01)	00H
	Mode control registers (TMC00, TMC01)	00H
	Capture/compare control registers 0 (CRC00, CRC01)	00H
	Output control registers (TOC00, TOC01)	00H
8-bit timer/event counter	Timer/counters (TM50, TM51, TM52)	00H
	Compare registers (CR50, CR51, CR52)	Undefined
	Clock selection registers (TCL50, TCL51, TCL52)	00H
	Mode control registers (TMC50, TMC51, TMC52)	04H[Note 6]
Watch timer	Mode register (WTNM0)	00H
Watchdog timer	Clock selection register (WDCS)	00H
	Mode register (WDTM)	00H

Notes 1. During reset input or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

2. When a reset is executed in the standby mode, the pre-reset status is held even after reset.

3. Do not set other than the initial value.

4. The initial value is 0CH, however, be sure to use this register after setting it to 08H.

5. Bit 2 values change depending on the V_{PP} level. FLPMC is incorporated only in the μPD78F0701Y.

6. The initial value is 04H, however, it is recognized as 00H during read.

Table 19-1. Hardware Statuses after Reset (2/3)

Hardware		Status After Reset
Clock output/buzzer output	Clock output selection register (CKS)	00H
A/D converter	Mode register 3 (ADM3)	00H
	Conversion result register 3 (ADCR3)	Undefined
	Analog input channel specification register 3 (ADS3)	00H
	Power-fail comparison threshold register 3 (PFT3)	00H
	Power-fail comparison mode register 3 (PFM3)	00H
Serial interface (UART0)	Asynchronous serial interface mode register 0 (ASIM0)	00H
	Asynchronous serial interface status register 0 (ASIS0)	00H
	Baud rate generator control register 0 (BRGC0)	00H
	Transmit shift register 0 (TXS0)	FFH
	Receive buffer register 0 (RXB0)	
Serial interface (SIO30, SIO31)	Operating mode registers (CSIM30, CSIM31)	00H
★	Shift registers (SIO30, SIO31)	00H
Serial interface (IIC0)	Transfer clock selection register (IICCL0)	00H
	Shift register (IIC0)	00H
	Control register (IICC0)	00H
	Status register (IICS0)	00H
	Slave address register 0 (SVA0)	00H
DCAN controller**Note**	CAN control register (CANC)	01H
	Transmit control register (TCR)	00H
	Receive message register (RMES)	00H
	Redefinition control register (REDEF)	00H
	CAN error status register (CANES)	00H
	Transmit error counter (TEC)	00H
	Receive error counter (REC)	00H
	Message count register (MCNT)	C0H
	Bit rate prescaler (BRPRS)	00H
	Synchronization control register 0 (SYNC0)	18H
	Synchronization control register 1 (SYNC1)	0EH
	Mask control register (MASKC)	00H

Note Applies to the μPD780701Y and 78F0701Y only.

Table 19-1. Hardware Statuses after Reset (3/3)

Hardware		Status After Reset
IEBus controller[Note]	Control register (BCR0)	00H
	Control data register (CDR)	01H
	Unit address register (UAR)	0000H
	Slave address register (SAR)	0000H
	Partner address register (PAR)	0000H
	Telegraph length register (DLR)	01H
	Data register (DR)	00H
	Unit status register (USR)	00H
	Interrupt status register (ISR)	00H
	Slave status register (SSR)	41H
	Successful communication counter (SCR)	01H
	Transmission counter (CCR)	20H
Interrupt	Request flag registers (IF0L, IF0H, IF1L, IF1H)	00H
	Mask flag registers (MK0L, MK0H, MK1L, MK1H)	FFH
	Priority specify flag registers (PR0L, PR0H, PR1L, PR1H)	FFH
	External interrupt rising edge enable register (EGP)	00H
	External interrupt falling edge enable register (EGN)	00H

Note Applies to the μPD780702Y and 78F0701Y only.

CHAPTER 20 μPD78F0701Y

The μPD78F0701Y is a flash memory version in the μPD780701Y Subseries.

The μPD78F0701Y incorporates a flash memory, which enables programs to be written, deleted, and rewritten on-board.

Table 20-1 lists differences between the μPD78F0701Y and mask ROM versions (μPD780701Y and 780702Y).

Table 20-1. Differences among μPD78F0701Y and Mask ROM Versions

Item		μPD78F0701Y	Mask ROM Version	
			μPD780701Y	μPD780702Y
Internal ROM configuration		Flash memory	Mask ROM	
IC		None	Available	
V_{PP}		Available	None	
Internal bus controller		DCAN controller/IEBus controller	DCAN controller	IEBus controller
	TX pin	DCAN/IEBus output (software switching)	DCAN output	IEBus output
	RX pin	DCAN/IEBus input (software switching)	DCAN input	IEBus input
Electrical specifications		Refer to relevant product data sheet.		

Caution There are differences in noise immunity and noise radiation between the flash memory and mask ROM versions. When pre-producing an application set with the flash memory version and then mass-producing it with the mask ROM version, be sure to conduct sufficient evaluations on the commercial samples (CS) (not engineering sample, ES) of the mask ROM version.

20.1 Internal Bus Controller (DCAN/IEBus) Switching

The μPD78F0701Y incorporates DCAN and IEBus controllers. Both controllers cannot be used at the same time. The initial status is DCAN controller.

By activating the IEBus unit (setting bit 7 (ENIEBUS) of IEBus control register 0 (BRC0) to 1), controller status is switched to the IEBus controller.

Interrupt request signals and the initial status of pins depend on the internal bus controller to be used.

Table 20-2 shows the interrupt request signals and the initial status of pins.

Table 20-2. Interrupt Request Signal and Initial Status of Pins

Item	When DCAN Controller Is Used	When IEBus Controller Is Used
Initial status of CTXD/ITX0 pin	High level	Low level
Interrupt request signal[Note]	INTCR	INTIE1
	INTCT	INTIE2
	INTCE	None

Note Corresponding flags are also switched. Refer to Table 17-2 for flags corresponding to interrupt request sources.

20.2 Internal Memory Size Switching Register (IMS)

The internal memory size switching register (IMS) is used to set the internal memory capacity.
IMS is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IMS to CFH.

Caution IMS is used under the initial value (CFH). Do not set IMS to the values other than CFH.

Figure 20-1. Format of Internal Memory Size Switching Register (IMS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IMS	RAM2	RAM1	RAM0	0	ROM3	ROM2	ROM1	ROM0	FFF0H	CFH	R/W

ROM3	ROM2	ROM1	ROM0	Internal ROM capacity selection
1	1	1	1	60 Kbytes
Other than above				Setting prohibited

RAM2	RAM1	RAM0	Internal high-speed RAM capacity selection
1	1	0	1,024 bytes
Other than above			Setting prohibited

20.3 Internal Expansion RAM Size Switching Register (IXS)

The internal expansion RAM size switching register (IXS) is used to set the internal expansion RAM capacity.
IXS is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IXS to 0CH.

Caution Always set IXS to 08H as the program initial setting. Since a reset set IXS to 0CH, always set IXS to 08H after resetting.

Figure 20-2. Format of Internal Expansion RAM Size Switching Register (IXS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IXS	0	0	0	IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	FFF4H	0CH	R/W

IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	Internal expansion RAM capacity selection
0	1	0	0	0	2,048 bytes
Other than above					Setting prohibited

20.4 Flash Memory Programming

On-board writing of flash memory (with device mounted on target system) is supported.

On-board writing is performed after connecting a dedicated flash programmer (Flashpro II (part number FL-PR2), Flashpro III (part number FL-PR3 and PG-FP3)) to the host machine and target system.

Moreover, writing to flash memory can also be performed using a flash memory writing adapter connected to Flashpro II or Flashpro III.

Remark FL-PR2 and FL-PR3 are products made by Naito Densei Machida Mfg. Co., Ltd.

20.4.1 Selection of communication mode

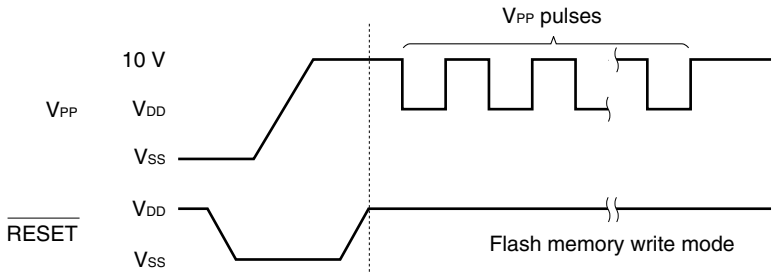
Writing to flash memory is performed using Flashpro II or Flashpro III and serial communication. Select the communication mode for writing from Table 20-3. For the selection of the communication mode, a format like the one shown in Figure 20-3 is used. The communication modes are selected using the V_{PP} pulse numbers shown in Table 20-3.

★ **Table 20-3. Communication Mode**

Communication Mode	Number of Channels	Pin Used	Number of V_{PP} Pulses
3-wire serial I/O	2	SI30/P30 SO30/P31 $\overline{SCK30}$/P32	0
		SI31/P20 SO31/P21 $\overline{SCK31}$/P22	1

Caution Be sure to select the number of V_{PP} pulses shown in Table 20-3 for the communication mode.

Figure 20-3. Communication Mode Selection Format



20.4.2 Flash memory programming function

Flash memory writing is performed through command and data transmit/receive operations using the selected communication mode. The main functions are listed in Table 20-4.

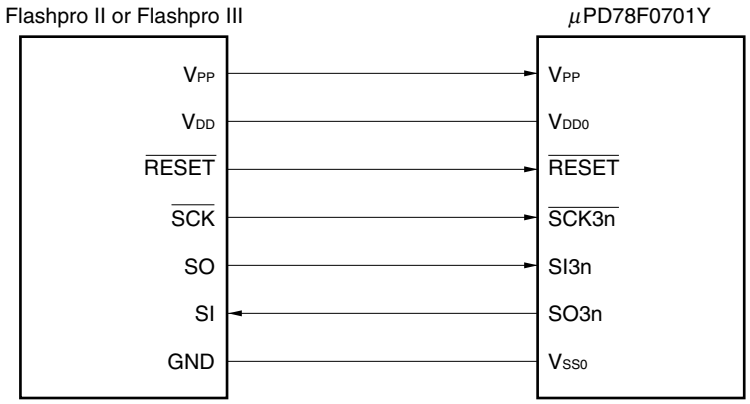
★ **Table 20-4. Main Functions of Flash Memory Programming**

Function	Description
Reset	Used to detect write stop and communication synchronization.
Batch verify	Compares entire memory contents and input data.
Batch contents verify	Compares entire memory contents in the different modes.
Batch delete	Deletes the entire memory contents.
Batch blank check	Checks the deletion status of the entire memory.
High-speed write	Performs writing to flash memory according to write start address and number of write data (bytes).
Continuous write	Performs successive write operations using the data input with high-speed write operation.
Batch prewrite	Writes 00H to entire memory.
Status	Checks the current operation mode and operation end.
Oscillation frequency setting	Inputs the resonator oscillation frequency information.
Delete time setting	Inputs the memory delete time.
Silicon signature read	Outputs the device name, memory capacity, and device block information.

★ 20.4.3 Flashpro II and Flashpro III connections

Connection of the Flashpro II or Flashpro III and the μPD78F0701Y is shown in Figure 20-4.

Figure 20-4. Connection of Flashpro II or Flashpro III Using 3-Wire Serial I/O Mode



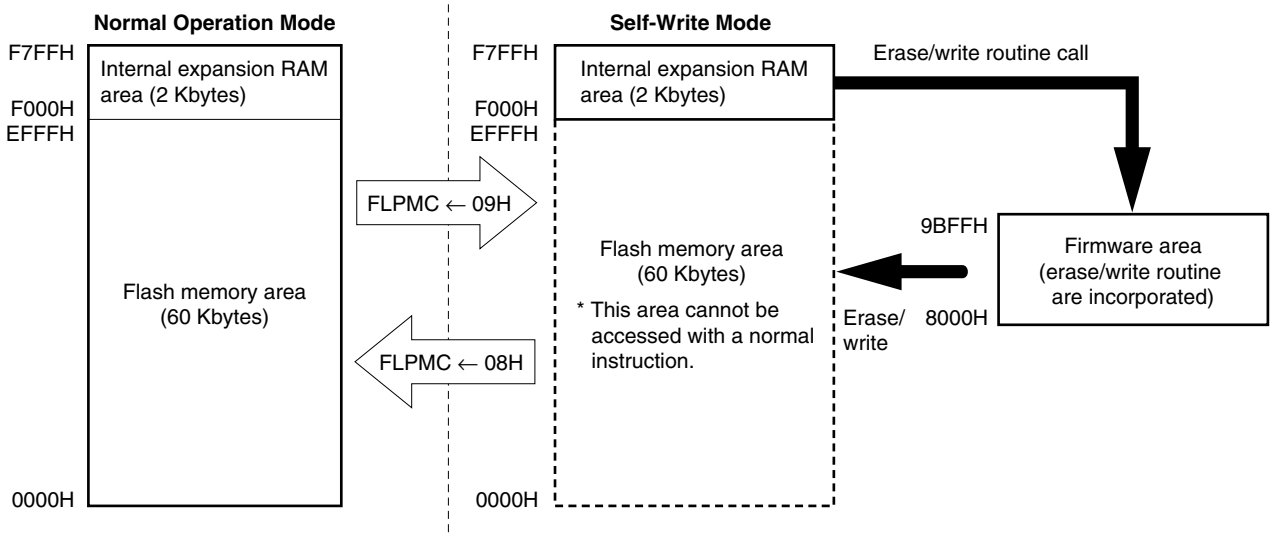
20.5 Flash Memory Programming by Self Write

With the μPD78F0701Y, it is possible to rewrite the flash memory using a program.

20.5.1 Flash memory configuration

The configuration of the flash memory is shown in Figure 20-5.

Figure 20-5. Configuration of Flash Memory



20.5.2 Flash programming mode control register

The flash programming mode control register (FLPMC) is a register for checking the operating mode selection and V_{PP} pin status.

FLPMC is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to 08H.

Figure 20-6. Format of Flash Programming Mode Control Register (FLPMC)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
FLPMC	0	0	0	0	1	VPP	0	FLSPM0	FFCDH	08H Note 1	R/W Note 2

VPP	V_{PP} pin voltage status
0	The voltage required for erase/write is not applied to V_{PP} pin.
1	Voltage greater than that of V_{DD} pin is applied to V_{PP} pin.

FLSPM0	Operating mode selection
0	Normal operating mode
1	Self-write mode

Notes 1. Bit 2 changes depending on the level of V_{PP}.

2. Bit 2 is read only.

Cautions 1. Be sure to set bits 1 and 4 to 7 to 0, and set bit 3 to 1.

2. The VPP bit gives the status of the voltage applied to the V_{PP} pin. If the VPP bit is 0, the voltage required for erase/write is not applied. However, even if VPP bit is 1, it does necessarily mean that the voltage required for erase/write is applied. Configure the hardware so the voltage required for erase/write is applied to the V_{PP} pin.

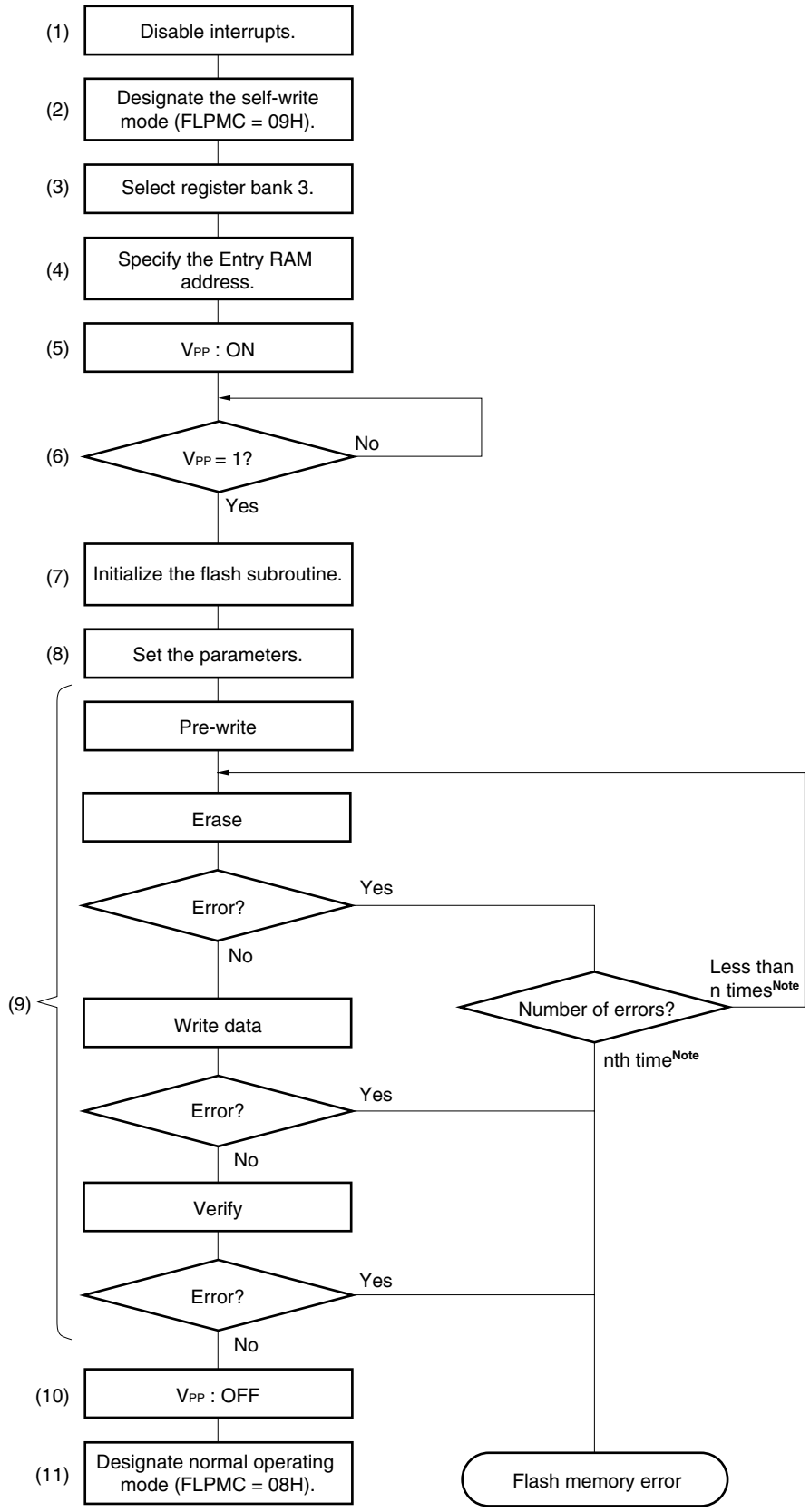
Also, if software will be used in addition to hardware to check that the voltage required for erase/write is applied, provide an external hardware detection circuit and use its output.

20.5.3 Self-write procedure

The procedure for performing self write is shown below (see **Figure 20-7**).

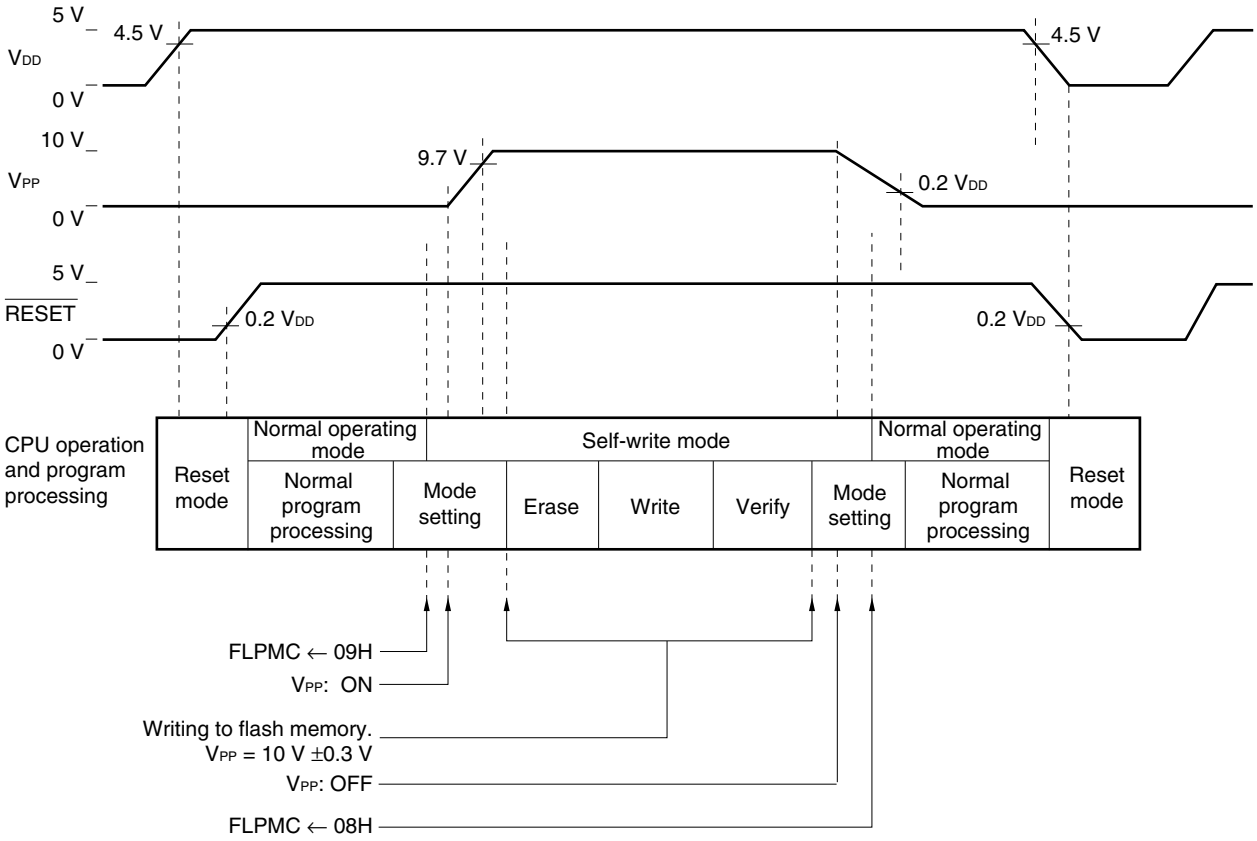
(1) Disable interrupts.
(2) Designate the self-write mode (FLPMC = 09H).
(3) Select register bank 3.
(4) Specify the start address of the entry RAM for the HL register.
(5) V_{PP}: ON (ON signal for power supply IC)
(6) Check the V_{PP} level.
(7) Initialize the flash subroutine.
(8) Set the parameters.
(9) Control the flash memory (erase, write, etc.).
(10) V_{PP}: OFF (OFF signal for power supply IC)
(11) Designate the normal operating mode (FLPMC = 08H).

Figure 20-7. Self-Programming Flowchart



Note Differs depending on the user program.

Figure 20-8. Self-Write Timing



20.5.4 CPU resources

The CPU resources used during self write are as follows:

- Register bank: BANK3 (8 bytes)
  B register: Status flag
  C register: Function number
  HL register: Entry RAM area starting address

- Stack area: Maximum 16 bytes

- Write data storage area: 1 to 256 bytes

- Entry RAM area: 32 bytes
  RAM area used by the self-write subroutines.
  Can be specified by the user using the HL register.

- Status flag

7	6	5	4	3	2	1	0
Parameter setting error	—	—	Verify error	Write error	—	Blank check error	—

20.5.5 Entry RAM area

A description of the entry RAM area is shown in Table 20-5.

Table 20-5. Entry RAM Area

Offset Value	Description
+0	Reserved area (1 byte)
+1	Reserved area (1 byte)
+2	Flash memory start address (2 bytes)
+4	Flash memory end address (2 bytes)
+6	No. of bytes written to flash memory (1 byte)
+7	Write time data (1 byte)
+8	Erase time data (3 bytes)
+11	Reserved area (3 bytes)
+14	Write data storage buffer for address (2 bytes)
+16	Total block number (1 byte)
+17	Total area number (1 byte)
+18 :	Reserved area (14 bytes)

Example When the value of the HL register of register bank 3 is 0FD00H
0FD00H: Status
0FD02H: Flash memory start address
0FD06H: Number of bytes written to flash memory
:

The following is a detailed explanation of the entry RAM area.

(a) Flash memory start address

This is the flash memory address value used by the _FlashByteWrite subroutine.

(b) Flash memory end address

This is the flash memory address value used by the _FlashGetInfo subroutine.

(c) Number of bytes written in flash memory

Area number and number of bytes written in the flash memory.

(d) Write time data

Set the following values depending on the operating frequency.

f_X (MHz)	Setting Value
1.00 to 1.28	20H
1.29 to 2.56	40H
2.57 to 5.12	60H
5.13 to 8.38	80H

(e) Erase time data

Setting value = Erase time (s) × Operating frequency/2^9 + 1

(Erase time range: 0.5 to 20 seconds)

Example Erase time: 2 seconds, Operating frequency: 6.29 MHz

Setting value = 2 × 6,291,456/512 + 1

= 24,577 (decimal)

= 6001H (hexadecimal)

(f) Write data storage buffer top address

This area contains the top address of the write data storage buffer area. The RAM data (write data), specified by the address data in this area, is written to the flash memory (_FlashByteWrite subroutine). The data in this area is specified as the top address and it is possible to specify up to a maximum of 256 bytes of write data.

(g) Total block number

This is the total flash memory block number used by the _FlashGetInfo subroutine.

(h) Total area number

This is the total flash memory area used by the _FlashGetInfo subroutine.

20.5.6 Self-write subroutines

The self-write subroutines and their functions are shown in Table 20-6 below.

Table 20-6. List of Self-Write Subroutines

Function Number		Subroutine Name	Function
Decimal	Hexadecimal		
0	00H	_FlashEnv	Initializes the flash subroutine.
1	01H	_FlashSetEnv	Sets the parameters.
2	02H	_FlashGetInfo	Reads flash memory data
16	10H	_FlashAreaBlankCheck	Performs a blank check of a specified area.
32	20H	_FlashAreaPreWrite	Performs prewrite for a specified area.
48	30H	_FlashAreaErase	Erases a specified area.
80	50H	_FlashByteWrite	Writes continuously in byte units.
96	60H	_FlashAreaIVerify	Performs internal verification of a specified area.

(1) _FlashEnv subroutine

[Function]

Initializes the flash subroutine.

[Argument]

Entry RAM address ..... 2 bytes (HL register)

[Return value]

None

[Register/memory status after called]

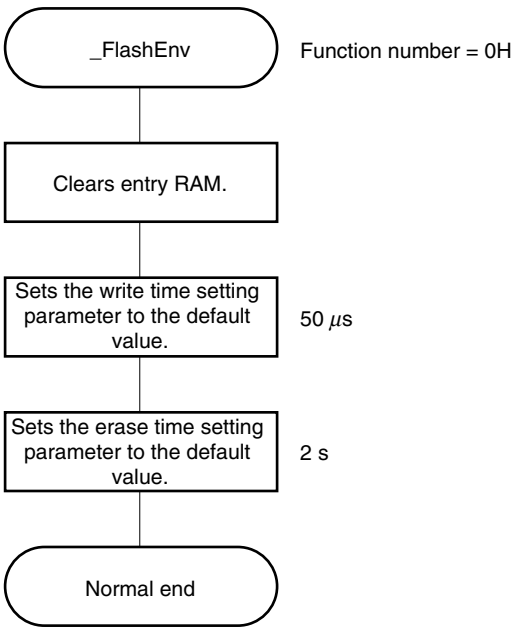
Entry RAM address

[Call example]

When the entry RAM address = 0FC30H

```
                DI
                SET1    FLSPM0
LOOP:           BF      VPP, $LOOP
                SEL     RB3
                MOVW    HL, #0FC30H         ; Entry RAM address
; * * * * * * * * * * Initialization * * * * * * * * * *
                MOV     C, #0H              ; FlashEnv (function number setting)
                CALL    !8100H
                        .
                        .
                        .
```

[Flowchart]



(2) _FlashSetEnv subroutine

[Function]

Sets the parameters.

[Argument]

Write time data: 2 bytes (offset value: +7)

Erase time data: 3 bytes (offset value: +8 to 10)

[Return value]

Status (B register or offset value: +0)

00H: Normal end

80H: Parameter setting error

[Register/memory status after called]

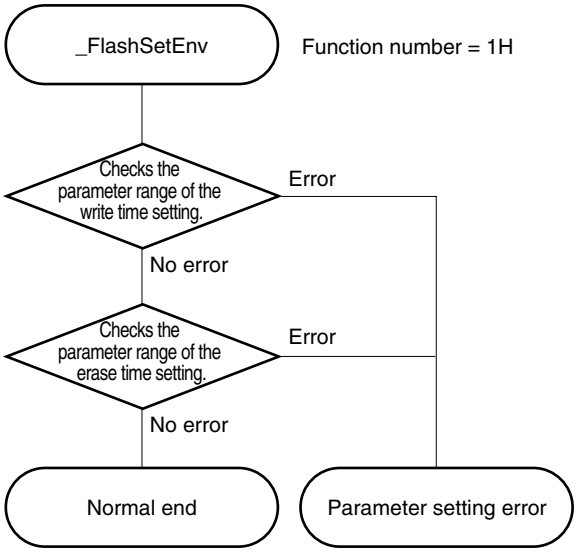
Entry RAM address, write time data, erase time data

[Call example]

When the entry RAM address = 0FC30H

```
        MOV     A, #20H         ; Write time data
        MOV     !0FC37H, A
        MOV     A, #30H         ; Erase time data
        MOV     !0FC38H, A      ; 8 s : 13130H
        MOV     A, #31H
        MOV     !0FC39H, A
        MOV     A, #01H
        MOV     !0FC3AH, A
    ;
        MOV     C, #1H          ;  FlashSetEnv (function number setting)
        CALL    !8100H
                .
                .
                .
```

[Flowchart]



(3) _FlashGetInfo subroutine

[Function]

Reads flash memory data

[Argument]

Option number (= 0, 1, 2, 3): 1 byte (offset value: +6)

0: Flash firmware version
1: Area number and block number
2: Area 0 end address
3: Area 1 end address

[Return Value]

Status (B register or offset value: +6)

00H: Normal end
80H: Optional number specification error

One of the information below can be specified in optional number.

- Flash firmware version (A register or offset value: +6)
- Area number and block number (AX register or offset value: +16 to 17)
- Specification area end address (AX register or offset value: +4 to 5)

[Register/memory status after called]

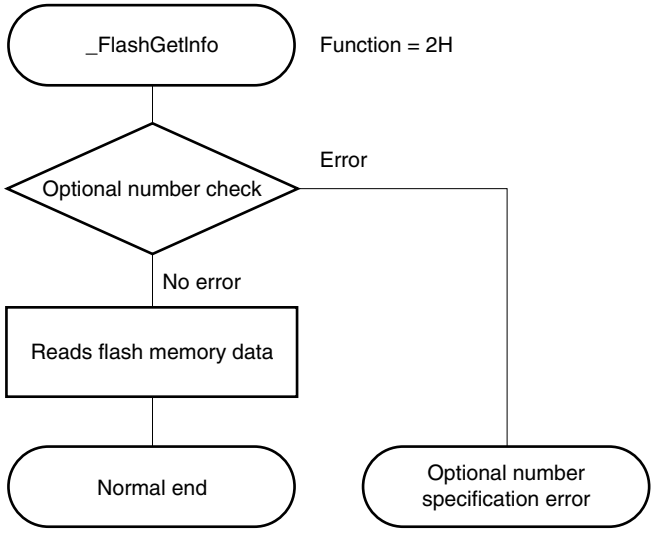
Entry RAM address optional number (Flash firmware version)

[Call example]

When the entry RAM address = 0FC30H

```
MOV     A, #02H         ; Area 1 end address information
MOV     !0FC36H, A
MOV     C, #02H         ; FlashGetInfo (Function number setting)
CALL    !8100H
        .
        .
        .
```

[Flowchart]



(4) _FlashAreaBlankCheck subroutine

[Function]

Performs a blank check of a specified area.

[Argument]

Area number (= 0, 1): 1 byte (offset value: +6)

0: Blank check of area 0000H to 7FFFH

1: Blank check of area 8000H to EFFFH

[Return value]

Status (B register or offset value: +0)

00H: Normal end

02H: Blank check error

80H: Area number specification error

[Register/memory status after called]

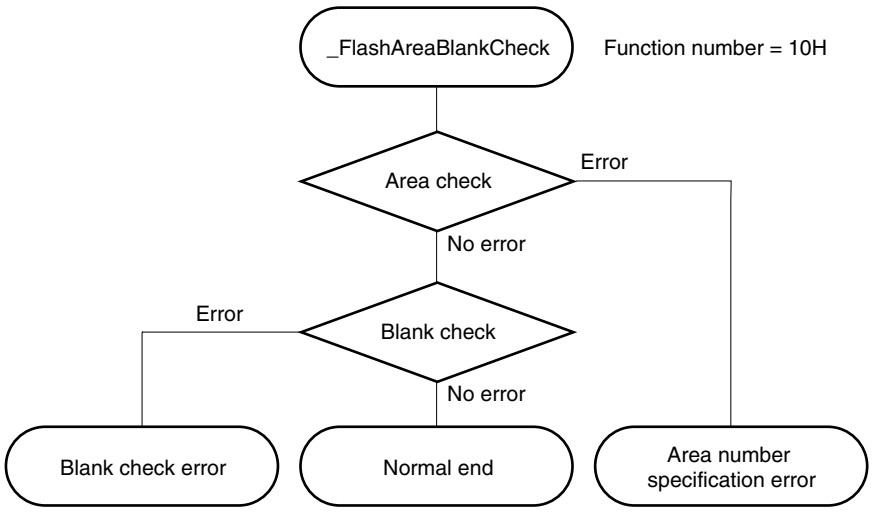
Entry RAM address, area number

[Call example]

When the entry RAM address = 0FC30H

```
MOV     A, #01H         ; Specifies area 1
MOV     !0FC36H, A
MOV     C, #10H         ; FlashAreaBlankCheck (function number setting)
CALL    !8100H
        .
        .
        .
```

[Flowchart]



(5) _FlashAreaPreWrite subroutine

[Function]

Performs prewrite for a specified area (writes 00H to a specified area).

[Argument]

Area number (= 0, 1): 1 byte (offset value: +6)

0: Prewrites area 0000H to 7FFFH.

1: Prewrites area 8000H to EFFFH.

[Return value]

Status (B register or offset value: +0)

00H: Normal end

08H: Write error

80H: Area number specification error

[Register/memory status after called]

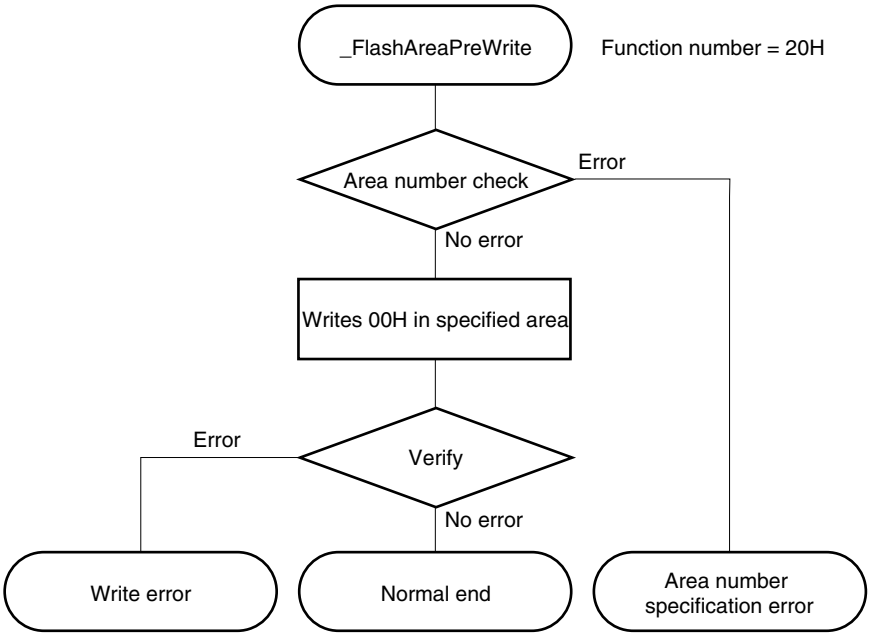
Entry RAM address, area number

[Call example]

When the entry RAM address = 0FC30H

```
        MOV     A, #1H          ; Specifies 8000H to EFFFH.
        MOV     !0FC36H, A
    ;
        MOV     C, #20H         ; FlashAreaPreWrite (function number setting)
        CALL    !8100H
                .
                .
                .
```

[Flowchart]



(6) _FlashAreaErase subroutine

[Function]

Erases a specified area.

[Argument]

Area number (= 0, 1): 1 byte (offset value: +6)

0: Erases area 0000H to 7FFFH.

1: Erases area 8000H to EFFFH.

[Return value]

Status (B register or offset value: +0)

00H: Normal end

02H: Blank check error

80H: Area number specification error

[Register/memory status after called]

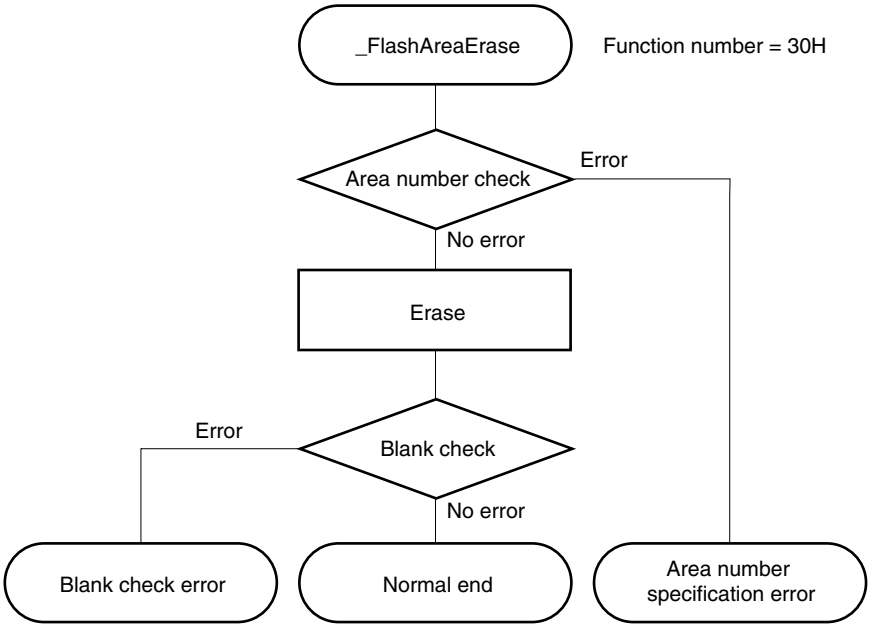
Entry RAM address, area number

[Call example]

When the entry RAM address = 0FC30H

```
    MOV     A, #1H          ; Specifies 8000H to EFFFH.
    MOV     !0FC36H, A
;
    MOV     C, #30H         ; FlashAreaErase (function number setting)
    CALL    !8100H
            .
            .
            .
```

[Flowchart]



(7) _FlashByteWrite subroutine

[Function]

Writes continuously in byte units.

[Argument]

Flash memory write start address: 2 bytes (offset value: +2)

Number of bytes[Note] written to flash memory: 1 byte (offset value: +6)

Write data storage buffer top address: 2 bytes (offset value: +14)

Note If 0 is set, it is possible to set a maximum of 256 bytes.

[Return value]

Status (B register or offset value: +0)

- 00H: Normal end
- 08H: Write error
- 80H: Write address error

[Register/memory status after called]

Entry RAM address, number of bytes written in flash memory

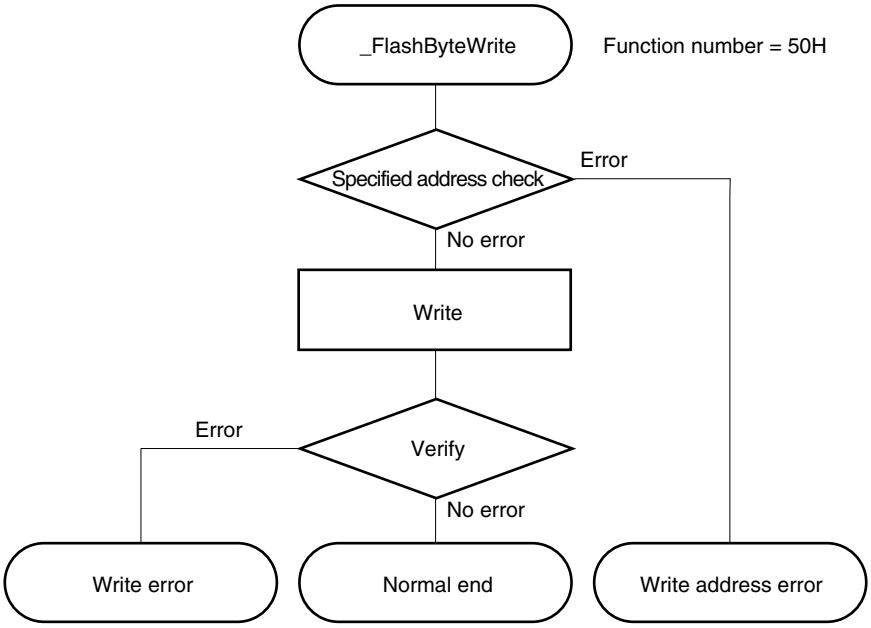
The flash memory write start address is updated to the address at the end of writing.

[Call example]

When the entry RAM address = 0FC30H

```
        MOVW    AX, #0FD00H   ; Write data storage buffer top address
        MOVW    !0FC3EH, AX
        MOVW    AX, #2000H    ; Flash memory write start address
        MOVW    !0FC32H, AX
        MOV     A, #0H        ; Number of bytes written to flash memory (256 bytes)
        MOV     !0FC36H, A
;
        MOV     C, #50H       ; FlashByteWrite (function number setting)
        CALL    !8100H
                .
                .
                .
```

[Flowchart]


_FlashByteWrite
Function number = 50H
Specified address check
Error
No error
Write
Verify
Error
No error
Write error
Normal end
Write address error

(8) _FlashArealVerify subroutine

[Function]

Performs internal verification of a specified area (reads the flash memory of a specified area in a different mode, and makes a comparison).

[Argument]

Area number (= 0, 1): 1 byte (offset value: +6)

0: Performs internal verification of area 0000H to 7FFFH.

1: Performs internal verification of area 8000H to EFFFH.

[Return value]

Status (B register or offset value: +0)

00H: Normal end

10H: Verify error

80H: Area number specification error

[Register/memory status after called]

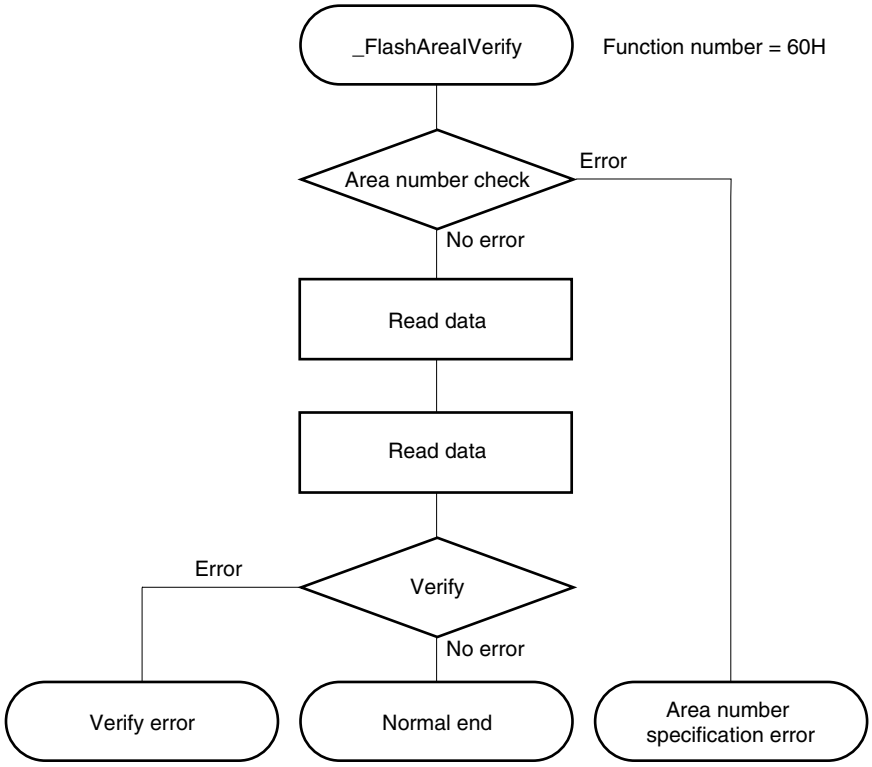
Entry RAM address, area number

[Call example]

When the entry RAM address = 0FC30H

```
        MOV     A, #01H         ; Area 1 specification
        MOV     !0FC36H, A
    ;
        MOV     C, #60H         ; FlashAreaIVerify (function number setting)
        CALL    !8100H
                .
                .
                .
```

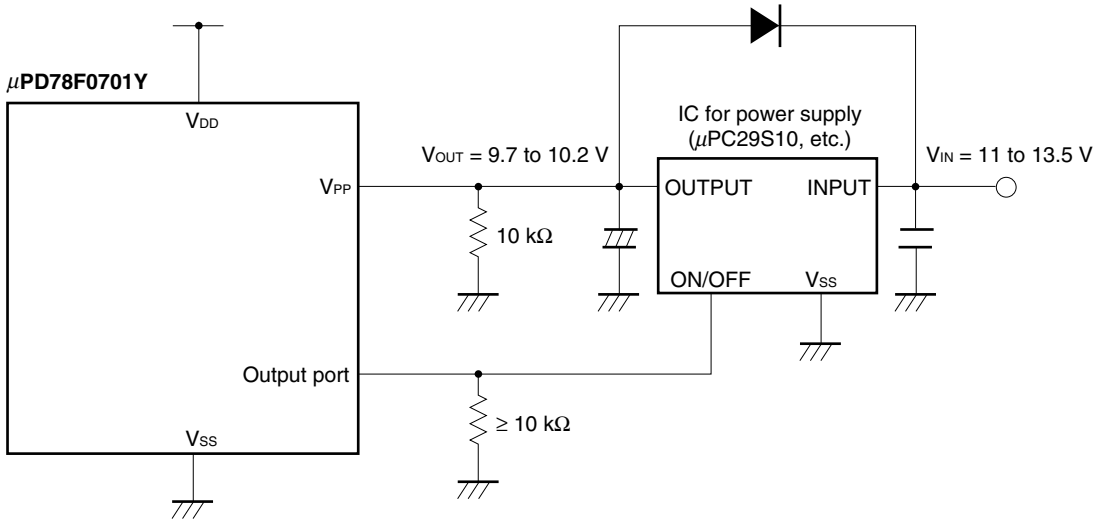
[Flowchart]



20.5.7 Self-write circuit configuration

The configuration of the self-write circuit is shown in Figure 20-9.

Figure 20-9. Configuration of Self-Write Circuit



CHAPTER 21 INSTRUCTION SET

This chapter lists each instruction set of the μPD780701Y Subseries. For details of their operation and operation codes, refer to the separate document **78K/0 Series User's Manual Instruction (U12326E)**.

21.1 Conventions

21.1.1 Operand identifiers and description methods

Operands are described in the “Operand” column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for detail). When there are two or more description methods, select one of them. Alphabetic letters in capitals and symbols, #, !, $ and [ ] are key words and must be described as they are. Each symbol has the following meaning.

- #: Immediate data specification
- !: Absolute address specification
- $: Relative address specification
- [ ]: Indirect address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, $, and [ ] symbols.

For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parentheses in the table below, R0, R1, R2, etc.) can be used for description.

Table 21-1. Operand Identifiers and Description Methods

Identifier	Description Method
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special-function register symbol**Note**
sfrp	Special-function register symbol (16-bit manipulatable register even addresses only)**Note**
saddr	FE20H to FF1FH Immediate data or labels
saddrp	FE20H to FF1FH Immediate data or labels (even address only)
addr16	0000H to FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions)
addr11	0800H to 0FFFH Immediate data or labels
addr5	0040H to 007FH Immediate data or labels (even address only)
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
RBn	RB0 to RB3

Note Addresses from FFD0H to FFDFH cannot be accessed with these operands.

Remark For special-function register symbols, refer to **Table 3-4 Special-Function Register List**.

21.1.2 Description of “operation” column

A : A register; 8-bit accumulator
X : X register
B : B register
C : C register
D : D register
E : E register
H : H register
L : L register
AX : AX register pair; 16-bit accumulator
BC : BC register pair
DE : DE register pair
HL : HL register pair
PC : Program counter
SP : Stack pointer
PSW : Program status word
CY : Carry flag
AC : Auxiliary carry flag
Z : Zero flag
RBS : Register bank select flag
IE : Interrupt request enable flag
NMIS : Non-maskable interrupt servicing flag
( ) : Memory contents indicated by address or register contents in parentheses
X_H, X_L : Higher 8 bits and lower 8 bits of 16-bit register
$\wedge$: Logical product (AND)
$\vee$: Logical sum (OR)
$\veebar$: Exclusive logical sum (exclusive OR)
$\overline{\quad}$: Inverted data
addr16 : 16-bit immediate data or label
jdisp8 : Signed 8-bit data (displacement value)

21.1.3 Description of “flag operation” column

(Blank) : Not affected
0 : Cleared to 0
1 : Set to 1
× : Set/cleared according to the result
R : Previously saved value is restored

21.2 Operation List

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit data transfer	**MOV**	r, #byte	2	4	–	r ← byte	
		saddr, #byte	3	6	7	(saddr) ← byte	
		sfr, #byte	3	–	7	sfr ← byte	
		A, r Note 3	1	2	–	A ← r	
		r, A Note 3	1	2	–	r ← A	
		A, saddr	2	4	5	A ← (saddr)	
		saddr, A	2	4	5	(saddr) ← A	
		A, sfr	2	–	5	A ← sfr	
		sfr, A	2	–	5	sfr ← A	
		A, !addr16	3	8	9	A ← (addr16)	
		!addr16, A	3	8	9	(addr16) ← A	
		PSW, #byte	3	–	7	PSW ← byte	× × ×
		A, PSW	2	–	5	A ← PSW	
		PSW, A	2	–	5	PSW ← A	× × ×
		A, [DE]	1	4	5	A ← (DE)	
		[DE], A	1	4	5	(DE) ← A	
		A, [HL]	1	4	5	A ← (HL)	
		[HL], A	1	4	5	(HL) ← A	
		A, [HL + byte]	2	8	9	A ← (HL + byte)	
		[HL + byte], A	2	8	9	(HL + byte) ← A	
		A, [HL + B]	1	6	7	A ← (HL + B)	
		[HL + B], A	1	6	7	(HL + B) ← A	
		A, [HL + C]	1	6	7	A ← (HL + C)	
		[HL + C], A	1	6	7	(HL + C) ← A	
	XCH	A, r Note 3	1	2	–	A ↔ r	
		A, saddr	2	4	6	A ↔ (saddr)	
		A, sfr	2	–	6	A ↔ (sfr)	
		A, !addr16	3	8	10	A ↔ (addr16)	
		A, [DE]	1	4	6	A ↔ (DE)	
		A, [HL]	1	4	6	A ↔ (HL)	
		A, [HL + byte]	2	8	10	A ↔ (HL + byte)	
		A, [HL + B]	2	8	10	A ↔ (HL + B)	
		A, [HL + C]	2	8	10	A ↔ (HL + C)	

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit data transfer	**MOVW**	rp, #word	3	6	–	rp ← word			
		saddrp, #word	4	8	10	(saddrp) ← word			
		sfrp, #word	4	–	10	sfrp ← word			
		AX, saddrp	2	6	8	AX ← (saddrp)			
		saddrp, AX	2	6	8	(saddrp) ← AX			
		AX, sfrp	2	–	8	AX ← sfrp			
		sfrp, AX	2	–	8	sfrp ← AX			
		AX, rp Note 3	1	4	–	AX ← rp			
		rp, AX Note 3	1	4	–	rp ← AX			
		AX, !addr16	3	10	12	AX ← (addr16)			
		!addr16, AX	3	10	12	(addr16) ← AX			
	XCHW	AX, rp Note 3	1	4	–	AX ↔ rp			
8-bit operation	**ADD**	A, #byte	2	4	–	A, CY ← A + byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte	×	×	×
		A, r Note 4	2	4	–	A, CY ← A + r	×	×	×
		r, A	2	4	–	r, CY ← r + A	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr)	×	×	×
		A, !addr16	3	8	9	A, CY ← A + (addr16)	×	×	×
		A, [HL]	1	4	5	A, CY ← A + (HL)	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A + (HL + byte)	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A + (HL + B)	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A + (HL + C)	×	×	×
	ADDC	A, #byte	2	4	–	A, CY ← A + byte + CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte + CY	×	×	×
		A, r Note 4	2	4	–	A, CY ← A + r + CY	×	×	×
		r, A	2	4	–	r, CY ← r + A + CY	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr) + CY	×	×	×
		A, !addr16	3	8	9	A, CY ← A + (addr16) + CY	×	×	×
		A, [HL]	1	4	5	A, CY ← A + (HL) + CY	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A + (HL + byte) + CY	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A + (HL + B) + CY	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A + (HL + C) + CY	×	×	×

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Only when rp = BC, DE or HL

4. Except “r = A”

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	**SUB**	A, #byte	2	4	–	A, CY ← A – byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte	×	×	×
		A, r Note 3	2	4	–	A, CY ← A – r	×	×	×
		r, A	2	4	–	r, CY ← r – A	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr)	×	×	×
		A, !addr16	3	8	9	A, CY ← A – (addr16)	×	×	×
		A, [HL]	1	4	5	A, CY ← A – (HL)	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A – (HL + byte)	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A – (HL + B)	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A – (HL + C)	×	×	×
	SUBC	A, #byte	2	4	–	A, CY ← A – byte – CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte – CY	×	×	×
		A, r Note 3	2	4	–	A, CY ← A – r – CY	×	×	×
		r, A	2	4	–	r, CY ← r – A – CY	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr) – CY	×	×	×
		A, !addr16	3	8	9	A, CY ← A – (addr16) – CY	×	×	×
		A, [HL]	1	4	5	A, CY ← A – (HL) – CY	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A – (HL + byte) – CY	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A – (HL + B) – CY	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A – (HL + C) – CY	×	×	×
	AND	A, #byte	2	4	–	A ← A ∧ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∧ byte	×		
		A, r Note 3	2	4	–	A ← A ∧ r	×		
		r, A	2	4	–	r ← r ∧ A	×		
		A, saddr	2	4	5	A ← A ∧ (saddr)	×		
		A, !addr16	3	8	9	A ← A ∧ (addr16)	×		
		A, [HL]	1	4	5	A ← A ∧ (HL)	×		
		A, [HL + byte]	2	8	9	A ← A ∧ (HL + byte)	×		
		A, [HL + B]	2	8	9	A ← A ∧ (HL + B)	×		
		A, [HL + C]	2	8	9	A ← A ∧ (HL + C)	×		

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except “r = A”

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	**OR**	A, #byte	2	4	–	A ← A∨byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr)∨byte	×		
		A, r Note 3	2	4	–	A ← A∨r	×		
		r, A	2	4	–	r ← r∨A	×		
		A, saddr	2	4	5	A ← A∨(saddr)	×		
		A, !addr16	3	8	9	A ← A∨(addr16)	×		
		A, [HL]	1	4	5	A ← A∨(HL)	×		
		A, [HL + byte]	2	8	9	A ← A∨(HL + byte)	×		
		A, [HL + B]	2	8	9	A ← A∨(HL + B)	×		
		A, [HL + C]	2	8	9	A ← A∨(HL + C)	×		
	XOR	A, #byte	2	4	–	A ← A⩛byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr)⩛byte	×		
		A, r Note 3	2	4	–	A ← A⩛r	×		
		r, A	2	4	–	r ← r⩛A	×		
		A, saddr	2	4	5	A ← A⩛(saddr)	×		
		A, !addr16	3	8	9	A ← A⩛(addr16)	×		
		A, [HL]	1	4	5	A ← A⩛(HL)	×		
		A, [HL + byte]	2	8	9	A ← A⩛(HL + byte)	×		
		A, [HL + B]	2	8	9	A ← A⩛(HL + B)	×		
		A, [HL + C]	2	8	9	A ← A⩛(HL + C)	×		
	CMP	A, #byte	2	4	–	A – byte	×	×	×
		saddr, #byte	3	6	8	(saddr) – byte	×	×	×
		A, r Note 3	2	4	–	A – r	×	×	×
		r, A	2	4	–	r – A	×	×	×
		A, saddr	2	4	5	A – (saddr)	×	×	×
		A, !addr16	3	8	9	A – (addr16)	×	×	×
		A, [HL]	1	4	5	A – (HL)	×	×	×
		A, [HL + byte]	2	8	9	A – (HL + byte)	×	×	×
		A, [HL + B]	2	8	9	A – (HL + B)	×	×	×
		A, [HL + C]	2	8	9	A – (HL + C)	×	×	×

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except “r = A”

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit operation	**ADDW**	AX, #word	3	6	–	AX, CY ← AX + word	×	×	×
	SUBW	AX, #word	3	6	–	AX, CY ← AX – word	×	×	×
	CMPW	AX, #word	3	6	–	AX – word	×	×	×
Multiply/ divide	**MULU**	X	2	16	–	AX ← A × X			
	DIVUW	C	2	25	–	AX (Quotient), C (Remainder) ← AX ÷ C			
Increment/ decrement	**INC**	r	1	2	–	r ← r + 1	×	×	
		saddr	2	4	6	(saddr) ← (saddr) + 1	×	×	
	DEC	r	1	2	–	r ← r – 1	×	×	
		saddr	2	4	6	(saddr) ← (saddr) – 1	×	×	
	INCW	rp	1	4	–	rp ← rp + 1			
	DECW	rp	1	4	–	rp ← rp – 1			
Rotate	**ROR**	A, 1	1	2	–	(CY, $A_7 \leftarrow A_0$, $A_{m-1} \leftarrow A_m$) × 1 time			×
	ROL	A, 1	1	2	–	(CY, $A_0 \leftarrow A_7$, $A_{m+1} \leftarrow A_m$) × 1 time			×
	RORC	A, 1	1	2	–	(CY ← A_0, $A_7 \leftarrow$ CY, $A_{m-1} \leftarrow A_m$) × 1 time			×
	ROLC	A, 1	1	2	–	(CY ← A_7, $A_0 \leftarrow$ CY, $A_{m+1} \leftarrow A_m$) × 1 time			×
	ROR4	[HL]	2	10	12	$A_{3-0} \leftarrow (HL)_{3-0}$, $(HL)_{7-4} \leftarrow A_{3-0}$, $(HL)_{3-0} \leftarrow (HL)_{7-4}$			
	ROL4	[HL]	2	10	12	$A_{3-0} \leftarrow (HL)_{7-4}$, $(HL)_{3-0} \leftarrow A_{3-0}$, $(HL)_{7-4} \leftarrow (HL)_{3-0}$			
BCD adjust	**ADJBA**		2	4	–	Decimal Adjust Accumulator after Addition	×	×	×
	ADJBS		2	4	–	Decimal Adjust Accumulator after Subtract	×	×	×
Bit manipu- late	**MOV1**	CY, saddr.bit	3	6	7	CY ← (saddr.bit)			×
		CY, sfr.bit	3	–	7	CY ← sfr.bit			×
		CY, A.bit	2	4	–	CY ← A.bit			×
		CY, PSW.bit	3	–	7	CY ← PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← (HL).bit			×
		saddr.bit, CY	3	6	8	(saddr.bit) ← CY			
		sfr.bit, CY	3	–	8	sfr.bit ← CY			
		A.bit, CY	2	4	–	A.bit ← CY			
		PSW.bit, CY	3	–	8	PSW.bit ← CY	×	×	
		[HL].bit, CY	2	6	8	(HL).bit ← CY			

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Bit manipulate	**AND1**	CY, saddr.bit	3	6	7	CY ← CY ∧ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ∧ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ∧ A.bit	×
		CY, PSW.bit	3	–	7	CY ← CY ∧ PSW.bit	×
		CY, [HL].bit	2	6	7	CY ← CY ∧ (HL).bit	×
	OR1	CY, saddr.bit	3	6	7	CY ← CY ∨ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ∨ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ∨ A.bit	×
		CY, PSW.bit	3	–	7	CY ← CY ∨ PSW.bit	×
		CY, [HL].bit	2	6	7	CY ← CY ∨ (HL).bit	×
	XOR1	CY, saddr.bit	3	6	7	CY ← CY ⊻ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ⊻ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ⊻ A.bit	×
		CY, PSW. bit	3	–	7	CY ← CY ⊻ PSW.bit	×
		CY, [HL].bit	2	6	7	CY ← CY ⊻ (HL).bit	×
	SET1	saddr.bit	2	4	6	(saddr.bit) ← 1	
		sfr.bit	3	–	8	sfr.bit ← 1	
		A.bit	2	4	–	A.bit ← 1	
		PSW.bit	2	–	6	PSW.bit ← 1	× × ×
		[HL].bit	2	6	8	(HL).bit ← 1	
	CLR1	saddr.bit	2	4	6	(saddr.bit) ← 0	
		sfr.bit	3	–	8	sfr.bit ← 0	
		A.bit	2	4	–	A.bit ← 0	
		PSW.bit	2	–	6	PSW.bit ← 0	× × ×
		[HL].bit	2	6	8	(HL).bit ← 0	
	SET1	CY	1	2	–	CY ← 1	1
	CLR1	CY	1	2	–	CY ← 0	0
	NOT1	CY	1	2	–	CY ← $\overline{CY}$	×

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Call/return	**CALL**	!addr16	3	7	–	(SP – 1) ← (PC + 3)H, (SP – 2) ← (PC + 3)L, PC ← addr16, SP ← SP – 2	
	CALLF	!addr11	2	5	–	(SP – 1) ← (PC + 2)H, (SP – 2) ← (PC + 2)L, PC15 – 11 ← 00001, PC10 – 0 ← addr11, SP ← SP – 2	
	CALLT	[addr5]	1	6	–	(SP – 1) ← (PC + 1)H, (SP – 2) ← (PC + 1)L, PCH ← (00000000, addr5 + 1), PCL ← (00000000, addr5), SP ← SP – 2	
	BRK		1	6	–	(SP – 1) ← PSW, (SP – 2) ← (PC + 1)H, (SP – 3) ← (PC + 1)L, PCH ← (003FH), PCL ← (003EH), SP ← SP – 3, IE ← 0	
	RET		1	6	–	PCH ← (SP + 1), PCL ← (SP), SP ← SP + 2	
	RETI		1	6	–	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R R R
	RETB		1	6	–	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3	R R R
Stack manipulate	**PUSH**	PSW	1	2	–	(SP – 1) ← PSW, SP ← SP – 1	
		rp	1	4	–	(SP – 1) ← rpH, (SP – 2) ← rpL, SP ← SP – 2	
	POP	PSW	1	2	–	PSW ← (SP), SP ← SP + 1	R R R
		rp	1	4	–	rpH ← (SP + 1), rpL ← (SP), SP ← SP + 2	
	MOVW	SP, #word	4	–	10	SP ← word	
		SP, AX	2	–	8	SP ← AX	
		AX, SP	2	–	8	AX ← SP	
Unconditional branch	**BR**	!addr16	3	6	–	PC ← addr16	
		$addr16	2	6	–	PC ← PC + 2 + jdisp8	
		AX	2	8	–	PCH ← A, PCL ← X	
Conditional branch	**BC**	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if CY = 1	
	BNC	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if CY = 0	
	BZ	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if Z = 1	
	BNZ	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if Z = 0	

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Condi-tional branch	**BT**	saddr.bit, $addr16	3	8	9	PC ← PC + 3 + jdisp8 if(saddr.bit) = 1	
		sfr.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if sfr.bit = 1	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 1	
		PSW.bit, $addr16	3	–	9	PC ← PC + 3 + jdisp8 if PSW.bit = 1	
		[HL].bit, $addr16	3	10	11	PC ← PC + 3 + jdisp8 if (HL).bit = 1	
	BF	saddr.bit, $addr16	4	10	11	PC ← PC + 4 + jdisp8 if(saddr.bit) = 0	
		sfr.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if sfr.bit = 0	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 0	
		PSW.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if PSW. bit = 0	
		[HL].bit, $addr16	3	10	11	PC ← PC + 3 + jdisp8 if (HL).bit = 0	
	BTCLR	saddr.bit, $addr16	4	10	12	PC ← PC + 4 + jdisp8 if(saddr.bit) = 1 then reset(saddr.bit)	
		sfr.bit, $addr16	4	–	12	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit	
		PSW.bit, $addr16	4	–	12	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	× × ×
		[HL].bit, $addr16	3	10	12	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit	
	DBNZ	B, $addr16	2	6	–	B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0	
		C, $addr16	2	6	–	C ← C –1, then PC ← PC + 2 + jdisp8 if C ≠ 0	
		saddr. $addr16	3	8	10	(saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if(saddr) ≠ 0	
CPU control	**SEL**	RBn	2	4	–	RBS1, 0 ← n	
	NOP		1	2	–	No Operation	
	EI		2	–	6	IE ← 1(Enable Interrupt)	
	DI		2	–	6	IE ← 0(Disable Interrupt)	
	HALT		2	6	–	Set HALT Mode	
	STOP		2	6	–	Set STOP Mode	

Notes 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remark One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).

21.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

First Operand \ Second Operand	#byte	A	r[Note]	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL + byte] [HL + B] [HL + C]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV ADD ADDC SUB SUBC AND OR XOR CMP											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV									DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											ROR4 ROL4
[HL + byte] [HL + B] [HL + C]		MOV											
X													MULU
C													DIVUW

Note Except r = A

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

Second Operand / First Operand	#word	AX	rp[Note]	sfrp	saddrp	!addr16	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW[Note]						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
SP	MOVW	MOVW						

Note Only when rp = BC, DE, HL

(3) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

Second Operand / First Operand	A.bit	sfr.bit	saddr.bit	PSW.bit	[HL].bit	CY	$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call instructions/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

First Operand \ Second Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruction					BT BF BTCLR DBNZ

(5) Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for the development of systems that employ the μPD780701Y Subseries.

Figure A-1 shows the development tool configuration.

Support of PC98-NX Series

Unless otherwise specified, products that operate in IBM PC/AT™ and compatibles can operate in the PC98-NX Series. When using the PC98-NX Series, refer to the descriptions for IBM PC/AT™ and compatibles.

Windows

Unless otherwise specified, "Windows" refers the following OSs.

- Windows 3.1
- Windows 95
- Windows NT™ Ver.4.0

Figure A-1. Development Tool Configuration (1/2)

(1) When using the in-circuit emulator IE-78K0-NS

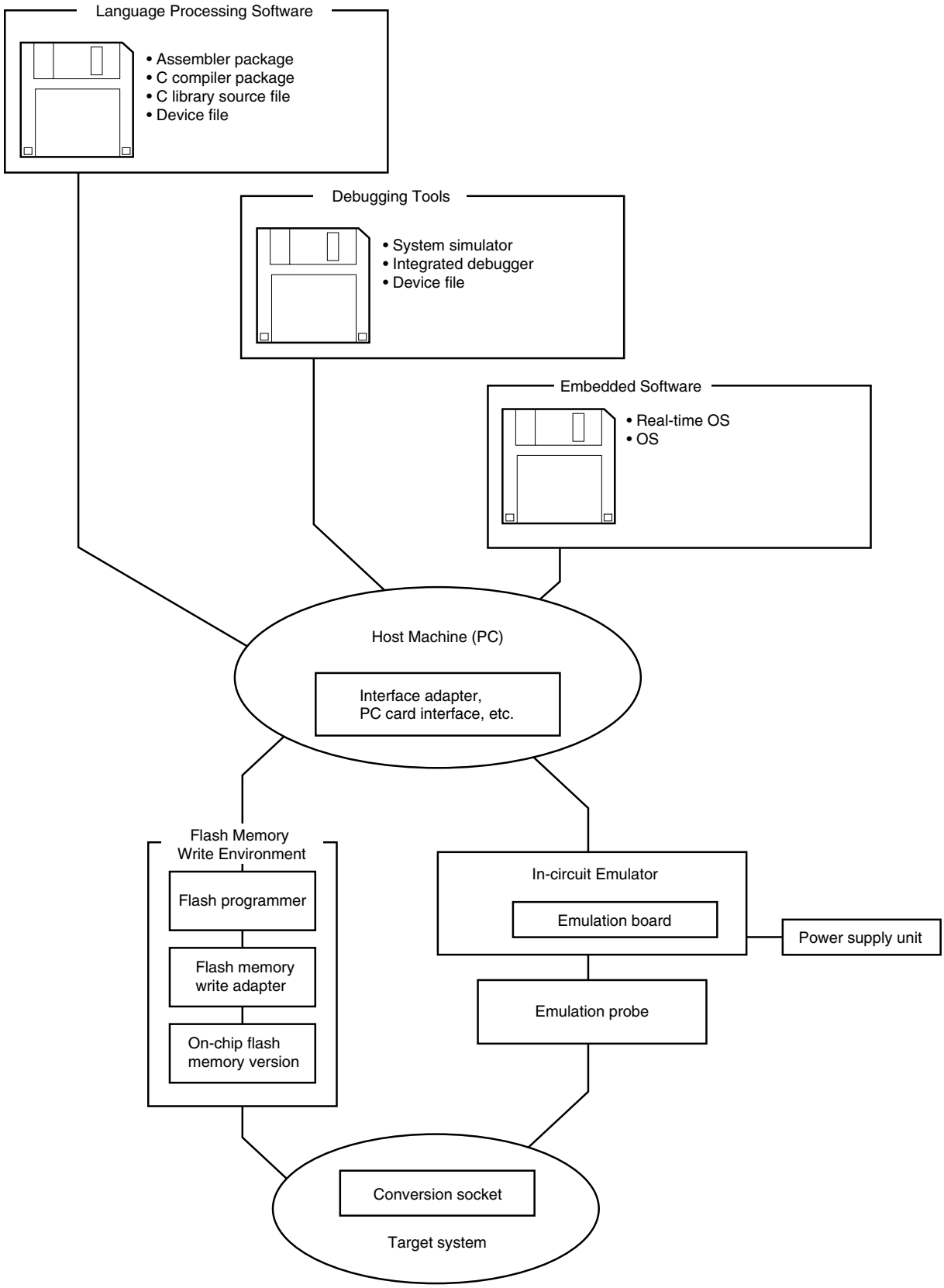
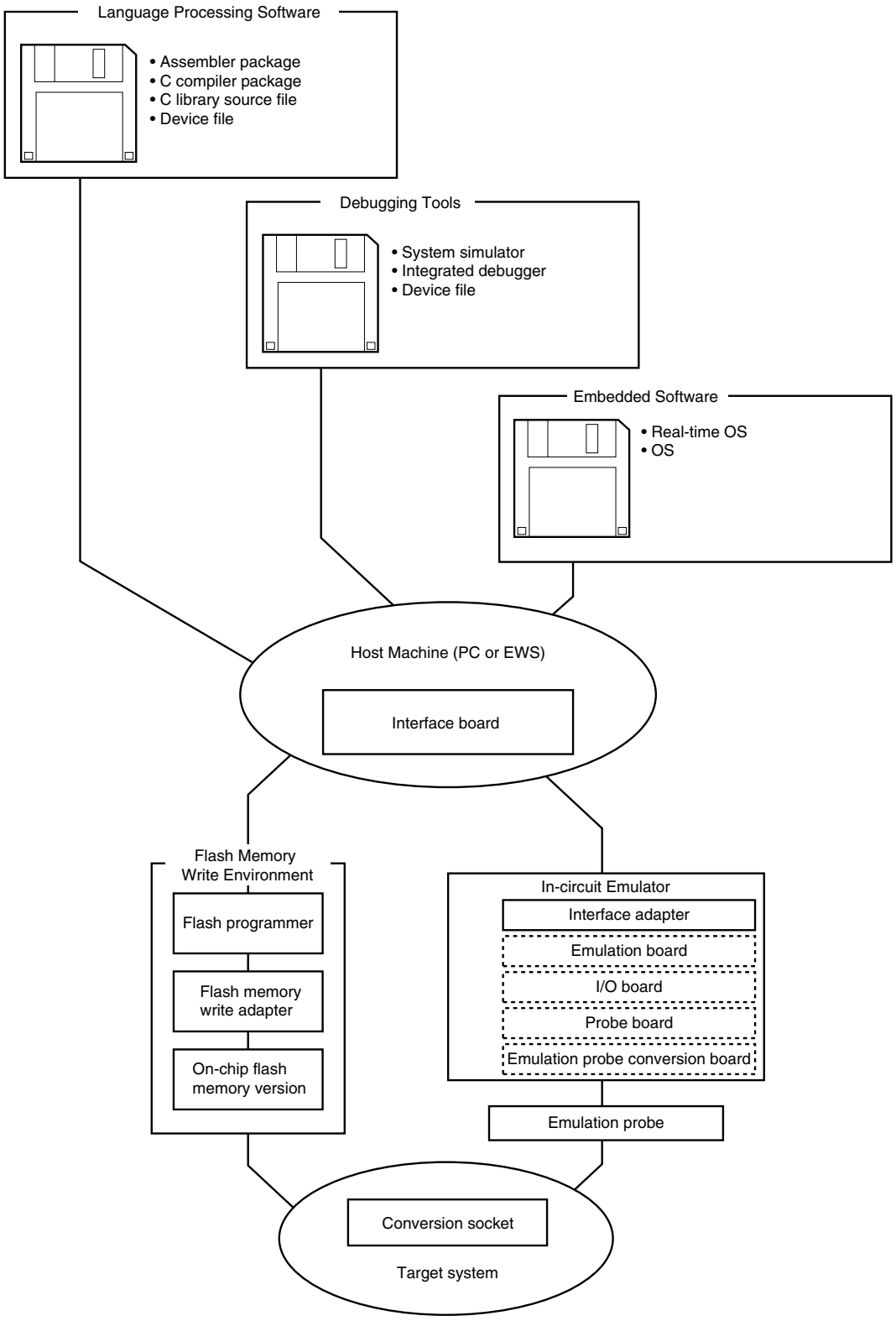


Figure A-1. Development Tool Configuration (2/2)

(2) When using the in-circuit emulator IE-78001-R-A



Remark Items in broken-line boxes differ according to the development environment. Refer to **A.3.1. Hardware**.

A.1 Language Processing Software

RA78K/0 Assembler Package	This assembler converts programs written in mnemonics into an object codes executable with a microcontroller. Further, this assembler is provided with functions capable of automatically creating symbol tables and branch instruction optimization. This assembler should be used in combination with an optical device file (DF780701). **<Caution when using RA78K/0 in PC environment>** This assembler package is a DOS-based application. It can also be used in Windows, however, by using the Project Manager (included in assembler package) on Windows.
	Part Number: μS××××RA78K0
CC78K/0 C Compiler Package	This compiler converts programs written in C language into object codes executable with a microcontroller. This compiler should be used in combination with an optical assembler package and device file. **<Caution when using CC78K/0 in PC environment>** This C compiler package is a DOS-based application. It can also be used in Windows, however, by using the Project Manager (included in assembler package) on Windows.
	Part Number: μS××××CC78K0
DF780701[Note] Device File	This file contains information peculiar to the device. This device file should be used in combination with an optical tool (RA78K/0, CC78K/0, SM78K0, ID78K0-NS, and ID78K0). Corresponding OS and host machine differ depending on the tool to be used with.
	Part Number: μS××××DF780701
CC78K/0-L C Library Source File	This is a source file of functions configuring the object library included in the C compiler package (CC78K/0). This file is required to match the object library included in C compiler package to the customer's specifications.
	Part Number: μS××××CC78K0-L

Note The DF780701 can be used in common with the RA78K/0, CC78K/0, SM78K0, ID78K0-NS, and ID78K0. DF780701 is under development.

Remark ××××in the part number differs depending on the host machine and OS used.

μS××××RA78K0
μS××××CC78K0
μS××××DF780701
μS××××CC78K0-L

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)[Note]	3.5-inch 2HD FD
AB13	IBM PC/AT and compatibles	Windows (Japanese version)[Note]	3.5-inch 2HC FD
BB13		Windows (English version)[Note]	
3P16	HP9000 series 700™	HP-UX™ (Rel. 10.10)	DAT (DDS)
3K13	SPARCstation™	SunOS™ (Rel. 4.1.4), Solaris™ (Rel. 2.5.1)	3.5-inch 2HC FD
3K15			1/4-inch CGMT
3R13	NEWS™ (RISC)	NEWS-OS™ (Rel. 6.1)	3.5-inch 2HC FD

Note Can be operated in the DOS environment.

A.2 Flash Memory Writing Tools

Flashpro II (part number FL-PR2) Flashpro III (part number FL-PR3, PG-FP3) Flash Programmer	Flash programmer dedicated to microcontrollers with on-chip flash memory.
FA-80GC Flash Memory Writing Adapter	Flash memory writing adapter used connected to the Flashpro II or Flashpro III (80-pin plastic QFP (GC-8BT type)).
Flashpro II controller, Flashpro III controller	These are programs controlled by a personal computer. They are included in the Flashpro II or Flashpro III.

Remark FL-PR2, FL-PR3, and FA-80GC are products made by Naito Densei Machida Mfg. Co., Ltd.
For further information, contact to Naito Densei Machida Mfg. Co., Ltd. (TEL +81-44-822-3813)

A.3 Debugging Tools

A.3.1 Hardware (1/2)

(1) When using the in-circuit emulator IE-78K0-NS

IE-78K0-NS In-circuit Emulator		The in-circuit emulator serves to debug hardware and software when developing application systems using a 78K/0 Series product. It corresponds to integrated debugger (ID78K0-NS). This emulator should be used in combination with power supply unit, emulation probe, and interface adapter which is required to connect this emulator to the host machine.
IE-70000-MC-PS-B Power Supply Unit		This adapter is used for supplying power from a receptacle of 100-V to 240-V AC.
IE-70000-98-IF-C Interface Adapter		This adapter is required when using the PC-9800 Series computer (except notebook type) as the IE-78K0-NS host machine (C-bus support).
IE-70000-CD-IF-A PC Card Interface		This is PC card and interface cable required when using a notebook-type computer as the IE-78K0-NS host machine (PCMCIA-socket supported).
IE-70000-PC-IF-C Interface Adapter		This adapter is required when using the IBM PC/AT or compatible computers as the IE-78K0-NS host machine (ISA-bus supported).
IE-70000-PCI-IF Interface Adapter		Adapter required when connecting a personal computer that includes a PCI bus as the host machine of the IE-78K0-NS.
IE-780701-NS-EM1[Note] Probe 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 and I/O board.
NP-80GC Emulation Probe		This probe is used to connect the in-circuit emulator to the target system and is designed for 80-pin plastic QFP (GC-8BT type).
	EV-9200GC-80 Conversion Socket (Refer to **Figures A-2 and A-3**)	This conversion socket connects the NP-80GC to the target system board designed to mount an 80-pin plastic QFP (GC-8BT type).

Note Under development

Remarks 1. NP-80GC is a product made by Naito Densei Machida Mfg. Co., Ltd.
For further information, contact to Naito Densei Machida Mfg. Co., Ltd.
(TEL +81-44-822-3813)

2. EV-9200GC-80 is sold in sets of five.

A.3.1 Hardware (2/2)

(2) When using the in-circuit emulator IE-78001-R-A

IE-78001-R-A In-circuit Emulator		The in-circuit emulator serves to debug hardware and software when developing application systems using a 78K/0 Series product. It corresponds to integrated debugger (ID78K0). This emulator should be used in combination with emulation probe and interface adapter, which is required to connect this emulator to the host machine.
IE-70000-98-IF-C Interface Adapter		This adapter is required when using the PC-9800 Series (except notebook type) as the IE-78001-R-A host machine.
IE-70000-PC-IF-C Interface Adapter		This adapter is required when using IBM PC/AT or compatibles as the IE-78001-R-A host machine.
IE-70000-PCI-IF Interface Adapter		Adapter required when connecting a personal computer that includes PCI bus as the host machine of the IE-78001-R-A.
IE-78000-R-SV3 Interface Adapter		This is adapter and cable required when using an EWS as the IE-78001-R-A host machine, and is used connected to the board in the IE-78001-R-A. 10Base-5 is supported for Ethernet™. For other methods, a commercially available conversion adapter is required.
IE-780701-NS-EM1**Note** Probe 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 and emulation probe conversion board.
	IE-78K0-R-EX1**Note** Emulation Probe Conversion Board	This board is required when using the IE-780701-NS-EM1 on the IE-78001-R-A.
EP-78230GC-R Emulation Probe		This probe is used to connect the in-circuit emulator to the target system and is designed for 80-pin plastic QFP (GC-8BT type).
	EV-9200GC-80 Conversion Socket (Refer to **Figures A-2 and A-3**)	This conversion socket connects the EP-78230GC-R to the target system board designed to mount an 80-pin plastic QFP (GC-8BT type).

Note Under development

Remark EV-9200GC-80 is sold in sets of five.

A.3.2 Software (1/2)

SM78K0 System Simulator	This system simulator is used to perform debugging at C source level or assembler level while simulating the operation of the target system on a host machine. This simulator runs on Windows. Use of the SM78K0 allows the execution of application logical testing and performance testing on an independent basis from hardware development without having to use an in-circuit emulator, thereby providing higher development efficiency and software quality. The SM78K0 should be used in combination with the optical device file (DF780701).
	Part Number: μS××××SM78K0

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××SM78K0

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT and compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB13		Windows (English version)	

A.3.2 Software (2/2)

ID78K0-NS Integrated Debugger (supporting in-circuit emulator IE-78K0-NS)	This debugger is a control program to debug 78K/0 Series microcontrollers. It adopts a graphical user interface, which is equivalent visually and operationally to Windows or OSF/Motif™. It also has an enhanced debugging function for C language programs, and thus trace results can be displayed on screen in C-language level by using the windows integration function which links a trace result with its source program, disassembled display, and memory display. In addition, by incorporating function modules such as task debugger and system performance analyzer, the efficiency of debugging programs, which run on real-time OSs can be improved. It should be used in combination with the optional device file.
ID78K0 Integrated Debugger (supporting in-circuit emulator IE-78001-R-A)	
	Part Number: μS××××ID78K0-NS, μS××××ID78K0

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××ID78K0-NS

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT and compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB13		Windows (English version)	

μS××××ID78K0

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT and compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB13		Windows (English version)	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT (DDS)
3K13	SPARCstation	SunOS (Rel. 4.1.4),	3.5-inch 2HC FD
3K15		Solaris (Rel. 2.5.1)	1/4-inch CGMT
3R13	NEWS (RISC)	NEWS-OS (Rel. 6.1)	3.5-inch 2HC FD

A.4 Upgrading from Former In-circuit Emulator for 78K/0 Series to IE-78001-R-A

If you already have a former in-circuit emulator for 78K/0 Series microcontrollers (IE-78000-R or IE-78000-R-A), that in-circuit emulator can operate as an equivalent to the IE-78001-R-A by replacing its internal break board with the IE-78001-R-BK.

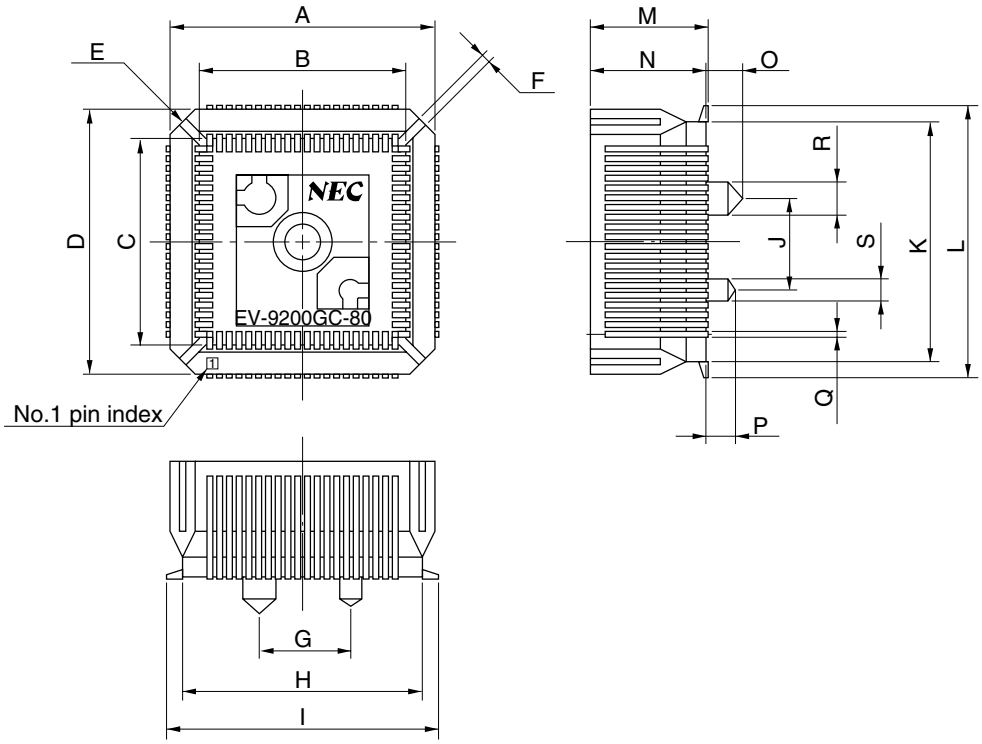
Table A-1. Upgrading Method from Former In-circuit Emulator for 78K/0 Series to IE-78001-R-A

In-circuit Emulator Owned	In-circuit Emulator Casing Upgrading[Note]	Board to be Purchased
IE-78000-R	Required	IE-78001-R-BK
IE-78000-R-A	Not required	

Note For upgrading a casing, send your in-circuit emulator to NEC Electronics.

Conversion Socket (EV-9200GC-80) Package Drawing and Footprints

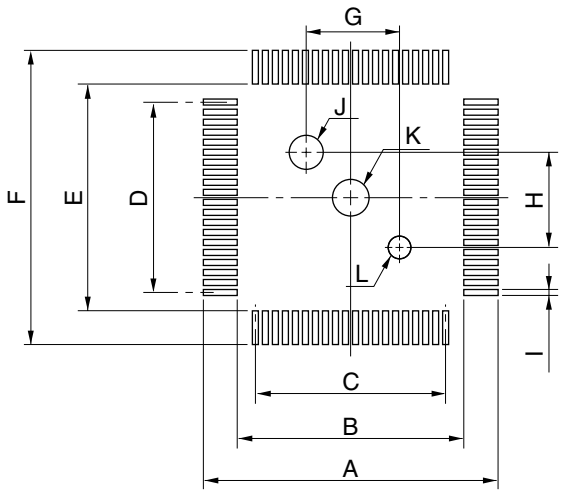
Figure A-2. EV-9200GC-80 Package Drawing (for Reference Only) (Unit: mm)



EV-9200GC-80-G0E

ITEM	MILLIMETERS	INCHES
A	18.0	0.709
B	14.4	0.567
C	14.4	0.567
D	18.0	0.709
E	4-C 2.0	4-C 0.079
F	0.8	0.031
G	6.0	0.236
H	16.0	0.63
I	18.7	0.736
J	6.0	0.236
K	16.0	0.63
L	18.7	0.736
M	8.2	0.323
O	8.0	0.315
N	2.5	0.098
P	2.0	0.079
Q	0.35	0.014
R	ϕ2.3	ϕ0.091
S	1.5	0.059

Figure A-3. EV-9200GC-80 Footprints (for Reference Only)



EV-9200GC-80-P1E

ITEM	MILLIMETERS	INCHES
A	19.7	0.776
B	15.0	0.591
C	$0.65\pm0.02\times19=12.35\pm0.05$	$0.026^{+0.001}_{-0.002}\times0.748=0.486^{+0.003}_{-0.002}$
D	$0.65\pm0.02\times19=12.35\pm0.05$	$0.026^{+0.001}_{-0.002}\times0.748=0.486^{+0.003}_{-0.002}$
E	15.0	0.591
F	19.7	0.776
G	6.0 ± 0.05	$0.236^{+0.003}_{-0.002}$
H	6.0 ± 0.05	$0.236^{+0.003}_{-0.002}$
I	0.35 ± 0.02	$0.014^{+0.001}_{-0.001}$
J	$\phi2.36\pm0.03$	$\phi0.093^{+0.001}_{-0.002}$
K	$\phi2.3$	$\phi0.091$
L	$\phi1.57\pm0.03$	$\phi0.062^{+0.001}_{-0.002}$

Caution Dimensions of mount pad for EV-9200 and that for target device (QFP) may be different in some parts. For the recommended mount pad dimensions for QFP, refer to "Semiconductor Device Mounting Technology Manual" (C10535E).

APPENDIX B EMBEDDED SOFTWARE

For efficient development and maintenance of the μPD780701Y Subseries, the following embedded products are available.

Real-Time OS (1/2)

RX78K/0 Real-time OS	RX78K/0 is a real-time OS conforming to the μITRON specifications. A tool (configurator) for generating the nucleus of RX78K/0 and plural information tables is supplied. Used in combination with an optional assembler package (RA78K/0) and device file (DF780701). **<Caution when using RX78K/0 in PC environment>** The real-time OS is a DOS-based application. It should be used in the DOS Prompt when using in Windows.
	Part number: μS××××RX78013-ΔΔΔΔ

Caution When purchasing the RX78K/0, fill in the purchase application form in advance and sign the user agreement.

Remark ×××× and ΔΔΔΔ in the part number differ depending on the host machine and OS used.

μS××××RX78013-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	Source program for mass-produced object

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)[Note]	3.5-inch 2HD FD
AB13	IBM PC/AT and compatibles	Windows (Japanese version)[Note]	3.5-inch 2HC FD
BB13		Windows (English version)[Note]	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT (DDS)
3K13	SPARCstation	SunOS (Rel. 4.1.4),	3.5-inch 2HC FD
3K15		Solaris (Rel. 2.5.1)	1/4-inch CGMT
3R13	NEWS (RISC)	NEWS-OS (Rel. 6.1)	3.5-inch 2HC FD

Note Can also be operated in the DOS environment.

Real-Time OS (2/2)

MX78K0 OS	MX78K0 is an OS for μITRON specification subsets. A nucleus for the MX78K0 is also included as a companion product. This manages tasks, events, and time. In the task management, determining the task execution order and switching from task to the next task are performed. **<Caution when using MX78K0 in PC environment>** The MX78K0 is a DOS-based application. It should be used in the DOS Prompt when using in Windows.
	Part number: μS××××MX78K0-ΔΔΔ

Remark ×××× and ΔΔΔ in the part number differ depending on the host machine and OS used.

μS××××MX78K0-ΔΔΔ

ΔΔΔ	Product Outline	Maximum Number for Use in Mass Production
001	Evaluation object	Use in preproduction stages.
××	Mass-production object	Use in mass production stages.
S01	Source program	This program is for the use of mass-production object purchasers.

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)[Note]	3.5-inch 2HD FD
AB13	IBM PC/AT and compatibles	Windows (Japanese version)[Note]	3.5-inch 2HC FD
BB13		Windows (English version)[Note]	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT (DDS)
3K13	SPARCstation	SunOS (Rel. 4.1.4),	3.5-inch 2HC FD
3K15		Solaris (Rel. 2.5.1)	1/4-inch CGMT
3R13	NEWS (RISC)	NEWS-OS (Rel. 6.1)	3.5-inch 2HC FD

Note Can also be operated in the DOS environment.

APPENDIX C REGISTER INDEX

C.1 Register Index (In Alphabetical Order with Respect to Register Names)

[A]

A/D conversion result register 3 (ADCR3) … 175
A/D converter mode register 3 (ADM3) … 177
Analog input channel specification register 3 (ADS3) … 178
Asynchronous serial interface mode register 0 (ASIM0) … 190
Asynchronous serial interface status register 0 (ASIS0) … 192

[B]

Baud rate generator control register 0 (BRGC0) … 192
Bit rate prescaler (BRPRS) ... 313

[C]

CAN control register (CANC) ... 305
CAN error status register (CANES) ... 309
Capture/compare control register 00 (CRC00) … 105
Capture/compare control register 01 (CRC01) … 105
Clock output selection register (CKS) … 169

[E]

8-bit compare register 50 (CR50) ... 138
8-bit compare register 51 (CR51) ... 138
8-bit compare register 52 (CR52) ... 138
8-bit timer mode control register 50 (TMC50) ... 139
8-bit timer mode control register 51 (TMC51) ... 139
8-bit timer mode control register 52 (TMC52) ... 139
8-bit timer/counter 50 (TM50) ... 138
8-bit timer/counter 51 (TM51) ... 138
8-bit timer/counter 52 (TM52) ... 138
External interrupt falling edge enable register (EGN) … 399
External interrupt rising edge enable register (EGP) … 399

[F]

Flash programming mode control register (FLPMC) … 430

[I]

IEBus communication success counter (SCR) ... 371
IEBus control data register (CDR) ... 357
IEBus control register 0 (BCR0) ... 353
IEBus data register (DR) ... 361
IEBus interrupt status register (ISR) ... 365
IEBus partner address register (PAR) ... 356
IEBus slave address register (SAR) ... 356
IEBus slave status register (SSR) ... 370

IEBus telegraph length register (DLR) ... 360
IEBus transmit counter (CCR) ... 371
IEBus unit address register (UAR) ... 356
IEBus unit status register (USR) ... 362
IIC0 control register (IICC0) … 216
IIC0 shift register (IIC0) … 224
IIC0 status register (IICS0) … 220
IIC0 transfer clock select register (IICCL0) … 223
Internal expansion RAM size switching register (IXS) … 426
Internal memory size switching register (IMS) … 426
Interrupt mask flag register 0H (MK0H) … 397
Interrupt mask flag register 0L (MK0L) … 397
Interrupt mask flag register 1H (MK1H) … 397
Interrupt mask flag register 1L (MK1L) … 397
Interrupt request flag register 0H (IF0H) … 396
Interrupt request flag register 0L (IF0L) … 396
Interrupt request flag register 1H (IF1H) … 396
Interrupt request flag register 1L (IF1L) … 396

[M]

Mask control register (MASKC) ... 321
Memory expansion mode register (MEM) … 88
Message count register (MCNT) ... 312

[O]

Oscillation stabilization time select register (OSTS) … 165, 412

[P]

Port 0 (P0) … 74
Port 2 (P2) … 76
Port 3 (P3) … 77
Port 4 (P4) … 79
Port 5 (P5) … 80
Port 6 (P6) … 81
Port 7 (P7) … 82
Port 8 (P8) … 84
Port 9 (P9) … 85
Port mode register 0 (PM0) … 86
Port mode register 2 (PM2) … 86, 171
Port mode register 3 (PM3) … 86, 111
Port mode register 4 (PM4) … 86
Port mode register 5 (PM5) … 86
Port mode register 6 (PM6) … 86
Port mode register 7 (PM7) … 86, 111, 145
Port mode register 8 (PM8) … 86
Port mode register 9 (PM9) … 86
Power-fail comparison mode register 3 (PFM3) ... 179
Power-fail comparison threshold register 3 (PFT3) ... 179

Prescaler mode register 00 (PRM00) ... 109
Prescaler mode register 01 (PRM01) ... 109
Priority specify flag register 0H (PR0H) ... 398
Priority specify flag register 0L (PR0L) ... 398
Priority specify flag register 1H (PR1H) ... 398
Priority specify flag register 1L (PR1L) ... 398
Processor clock control register (PCC) ... 91
Program status word (PSW) ... 49, 400
Pull-up resistor option register 0 (PU0) ... 87
Pull-up resistor option register 2 (PU2) ... 87
Pull-up resistor option register 3 (PU3) ... 87
Pull-up resistor option register 4 (PU4) ... 87
Pull-up resistor option register 5 (PU5) ... 87
Pull-up resistor option register 6 (PU6) ... 87
Pull-up resistor option register 7 (PU7) ... 87

[R]

Receive buffer register 0 (RXB0) ... 189
Receive error counter (REC) ... 311
Receive message register (RMES) ... 320
Redefinition control register (REDEF) ... 323

[S]

Serial I/O shift register 30 (SIO30) ... 207
Serial I/O shift register 31 (SIO31) ... 207
Serial operation mode register 30 (CSIM30) ... 208
Serial operation mode register 31 (CSIM31) ... 208
16-bit timer capture/compare register 000 (CR000) ... 101
16-bit timer capture/compare register 001 (CR001) ... 101
16-bit timer capture/compare register 010 (CR010) ... 102
16-bit timer capture/compare register 011 (CR011) ... 102
16-bit timer mode control register 00 (TMC00) ... 102
16-bit timer mode control register 01 (TMC01) ... 102
16-bit timer output control register 00 (TOC00) ... 107
16-bit timer output control register 01 (TOC01) ... 107
16-bit timer/counter 00 (TM00) ... 100
16-bit timer/counter 01 (TM01) ... 100
Slave address register 0 (SVA0) ... 224
Synchronization control register 0 (SYNC0) ... 314
Synchronization control register 1 (SYNC1) ... 314

[T]

Timer clock select register 50 (TCL50) ... 143
Timer clock select register 51 (TCL51) ... 143
Timer clock select register 52 (TCL52) ... 143
Transmit control register (TCR) ... 318
Transmit error counter (TEC) ... 311
Transmit shift register 0 (TXS0) ... 189

[W]

Watch timer operation mode register 0 (WTNM0) … 158
Watchdog timer clock select register (WDCS) … 163
Watchdog timer mode register (WDTM) … 164

C.2 Register Index (In Alphabetical Order with Respect to Register Symbol)

[A]

ADCR3: A/D conversion result register 3 … 175
ADM3: A/D converter mode register 3 … 177
ADS3: Analog input channel specification register 3 … 178
ASIM0: Asynchronous serial interface mode register 0 … 190
ASIS0: Asynchronous serial interface status register 0 … 192

[B]

BCR0: IEBus control register 0 ... 353
BRGC0: Baud rate generator control register 0 … 192
BRPRS: Bit rate prescaler ... 313

[C]

CANC: CAN control register ... 305
CANES: CAN error status register ... 309
CCR: IEBus transmit counter ... 371
CDR: IEBus control data register ... 357
CKS: Clock output selection register … 169
CR000: 16-bit timer capture/compare register 000 … 101
CR010: 16-bit timer capture/compare register 010 … 102
CR001: 16-bit timer capture/compare register 001 … 101
CR011: 16-bit timer capture/compare register 011 … 102
CR50: 8-bit compare register 50 ... 138
CR51: 8-bit compare register 51 ... 138
CR52: 8-bit compare register 52 ... 138
CRC00: Capture/compare control register 00 … 105
CRC01: Capture/compare control register 01 … 105
CSIM30: Serial operation mode register 30 … 208
CSIM31: Serial operation mode register 31 … 208

[D]

DLR: IEBus telegraph length register ... 360
DR: IEBus data register ... 361

[E]

EGN: External interrupt falling edge enable register … 399
EGP: External interrupt rising edge enable register … 399

[F]

FLPMC: Flash programming mode control register … 430

[I]

IF0H: Interrupt request flag register 0H … 396
IF0L: Interrupt request flag register 0L … 396
IF1H: Interrupt request flag register 1H … 396
IF1L: Interrupt request flag register 1L … 396

IIC0: IIC0 shift register … 224
IICC0: IIC0 control register … 216
IICCL0: IIC0 transfer clock select register … 223
IICS0: IIC0 status register … 220
IMS: Internal memory size switching register … 426
ISR: IEBus interrupt status register ... 365
IXS: Internal expansion RAM size switching register … 426

[M]

MASKC: Mask control register ... 321
MCNT: Message count register ... 312
MEM: Memory expansion mode register … 88
MK0H: Interrupt mask flag register 0H … 397
MK0L: Interrupt mask flag register 0L … 397
MK1H: Interrupt mask flag register 1H … 397
MK1L: Interrupt mask flag register 1L … 397

[O]

OSTS: Oscillation stabilization time select register … 165, 412

[P]

P0: Port 0 … 74
P2: Port 2 … 76
P3: Port 3 … 77
P4: Port 4 … 79
P5: Port 5 … 80
P6: Port 6 … 81
P7: Port 7 … 82
P8: Port 8 … 84
P9: Port 9 … 85
PAR: IEBus partner address register ... 356
PCC: Processor clock control register … 91
PFM3: Power-fail comparison mode register 3 ... 179
PFT3: Power-fail comparison threshold register 3 ... 179
PM0: Port mode register 0 … 86
PM2: Port mode register 2 … 86, 171
PM3: Port mode register 3 … 86, 111
PM4: Port mode register 4 … 86
PM5: Port mode register 5 … 86
PM6: Port mode register 6 … 86
PM7: Port mode register 7 … 86, 111, 145
PM8: Port mode register 8 … 86
PM9: Port mode register 9 … 86
PR0H: Priority specify flag register 0H … 398
PR0L: Priority specify flag register 0L … 398
PR1H: Priority specify flag register 1H … 398
PR1L: Priority specify flag register 1L … 398
PRM00: Prescaler mode register 00 … 109

PRM01: Prescaler mode register 01 … 109
PSW: Program status word … 49, 400
PU0: Pull-up resistor option register 0 … 87
PU2: Pull-up resistor option register 2 … 87
PU3: Pull-up resistor option register 3 … 87
PU4: Pull-up resistor option register 4 … 87
PU5: Pull-up resistor option register 5 … 87
PU6: Pull-up resistor option register 6 … 87
PU7: Pull-up resistor option register 7 … 87

[R]

REC: Receive error counter ... 311
REDEF: Redefinition control register ... 323
RMES: Receive message register ... 320
RXB0: Receive buffer register 0 ... 189

[S]

SAR: IEBus slave address register ... 356
SCR: IEBus communication success counter ... 371
SIO30: Serial I/O shift register 30 … 207
SIO31: Serial I/O shift register 31 … 207
SSR: IEBus slave status register ... 370
SVA0: Slave address register 0 … 224
SYNC0: Synchronization control register 0 ... 314
SYNC1: Synchronization control register 1 ... 314

[T]

TCL50: Timer clock select register 50 ... 143
TCL51: Timer clock select register 51 ... 143
TCL52: Timer clock select register 52 ... 143
TCR: Transmit control register ... 318
TEC: Transmit error counter ... 311
TM00: 16-bit timer/counter 00 … 100
TM01: 16-bit timer/counter 01 ... 100
TM50: 8-bit timer/counter 50 ... 138
TM51: 8-bit timer/counter 51 ... 138
TM52: 8-bit timer/counter 52 ... 138
TMC00: 16-bit timer mode control register 00 … 102
TMC01: 16-bit timer mode control register 01 … 102
TMC50: 8-bit timer mode control register 50 ... 139
TMC51: 8-bit timer mode control register 51 ... 139
TMC52: 8-bit timer mode control register 52 ... 139
TOC00: 16-bit timer output control register 00 … 107
TOC01: 16-bit timer output control register 01 … 107
TXS0: Transmit shift register 0 … 189

[U]

UAR: IEBus unit address register ... 356
USR: IEBus unit status register ... 362

[W]

WDCS: Watchdog timer clock select register … 163
WDTM: Watchdog timer mode register … 164
WTNM0: Watch timer operation mode register 0 … 158

★

APPENDIX D REVISION HISTORY

D.1 Major Revisions in This Edition

Page	Description
p. 55	**CHAPTER 3 CPU ARCHITECTURE** • **Table 3-4 Special Function Register List** Modification of values after reset of 16-bit timer capture/compare register 000 (CR000), 16-bit timer capture/compare register 010 (CR010), 16-bit timer capture/compare register 001 (CR001), and 16-bit timer capture/compare register 011 (CR011) from undefined to 0000H Modification of values after reset of serial I/O shift register 30 (SIO30) and serial I/O shift register 31 (SIO31) from undefined to 00H
p. 75	**CHAPTER 4 PORT FUNCTIONS** • Modification of **Figure 4-2 Block Diagram of P00 to P07**
p. 97 p. 101 p. 102	**CHAPTER 6 16-BIT TIMER/EVENT COUNTER** • Modification of description in **6.1 (3) Watch timer (WTN0)** • Modification of values after reset in **6.3 (2) 16-bit timer capture/compare register 000, 001 (CR000, CR001)** from undefined to 0000H • Modification of values after reset in **6.3 (3) 16-bit timer capture/compare register 010, 011 (CR010, CR011)** from undefined to 0000H
p. 326	**CHAPTER 15 DCAN CONTROLLER (μPD780701Y, 78F0701Y ONLY)** • Addition of **Caution** in **15.15.3 DCAN sleep mode**
pp. 333 to 388	**CHAPTER 16 IEBus CONTROLLER (μPD780702Y, 78F0701Y ONLY)** • General modification of chapter
p. 416	**CHAPTER 18 STANDBY FUNCTION** • **Table 18-3 STOP Mode Operating Statuses** Modification of operating statuses of 16-bit timer/event counter, 8-bit timer/event counter, and serial interface
pp. 421, 422	**CHAPTER 19 RESET FUNCTION** • **Table 19-1 Hardware Statuses after Reset** Modification of statuses after reset of 16-bit timer capture/compare register 000 (CR000), 16-bit timer capture/compare register 010 (CR010), 16-bit timer capture/compare register 001 (CR001), and 16-bit timer capture/compare register 011 (CR011) from undefined to 0000H Modification of statuses after reset of serial I/O shift register 30 (SIO30) and serial I/O shift register 31 (SIO31) from undefined to 00H
p. 427 p. 428 p. 428	**CHAPTER 20 μPD78F0701Y** • **Table 20-3 Communication Mode** Deletion of I^2C bus and UART from communication modes • **Table 20-4 Main Functions of Flash Memory Programming** Deletion of baud rate setting and I^2C mode setting from functions • **20.4.3 Flashpro II and Flashpro III connections** Deletion of connection diagrams in I^2C and UART modes
p. 483	Addition of **APPENDIX D REVISION HISTORY**