

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.



User's Manual

μ PD178003 Subseries

8-bit Single-chip Microcontroller

μ PD178002

μ PD178003

[MEMO]

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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 once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build 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 bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

MS-DOS, Windows, and WindowsNT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

PC/AT and PC DOS are trademarks of IBM Corporation.

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

SPARCstation is a trademark of SPARC International, Inc.

Solaris and SunOS are trademarks of Sun Microsystems, Inc.

Ethernet is a trademark of Xerox Corporation.

NEWS and NEWS-OS are trademarks of Sony Corporation.

OSF/Motif is a trademark of Open Software Foundation, Inc.

TRON is an abbreviation of The Realtime Operating system Nucleus.

ITRON is an abbreviation of Industrial TRON.

The export of this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by the customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

- **The information in this document is subject to change without notice. Before using this document, please confirm that this is the latest version.**
 - **Not all devices/types available in every country. Please check with local NEC 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 Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.
 - NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation 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 the customer's equipment shall be done under the full responsibility of the customer. NEC Corporation assumes no responsibility for any losses incurred by the customer or third parties arising from the use of these circuits, software, and information.
 - While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.
 - NEC devices are classified into the following three quality grades:
"Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device 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 or medical equipment for life support, etc.
- The quality grade of NEC devices is "Standard" unless otherwise specified in NEC's Data Sheets or Data Books. If customers intend to use NEC devices for applications other than those specified for Standard quality grade, they should contact an NEC sales representative in advance.

M7D 98.12

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC 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.

NEC Electronics Inc. (U.S.)

Santa Clara, California
Tel: 408-588-6000
800-366-9782
Fax: 408-588-6130
800-729-9288

NEC Electronics (Germany) GmbH

Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.

Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.l.

Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH

Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.

Velizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.

Spain Office
Madrid, Spain
Tel: 91-504-2787
Fax: 91-504-2860

NEC Electronics (Germany) GmbH

Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

NEC Electronics Hong Kong Ltd.

Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.

Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.

United Square, Singapore 1130
Tel: 65-253-8311
Fax: 65-250-3583

NEC Electronics Taiwan Ltd.

Taipei, Taiwan
Tel: 02-2719-2377
Fax: 02-2719-5951

NEC do Brasil S.A.

Electron Devices Division
Rodovia Presidente Dutra, Km 214
07210-902-Guarulhos-SP Brasil
Tel: 55-11-6465-6810
Fax: 55-11-6465-6829

J99.1

MAJOR REVISIONS IN THIS EDITION

Page	Description
Throughout	Number of A/D converter channels changed from six to three.
p.35 p.36	CHAPTER 2 PIN FUNCTION Table 2-1. I/O Circuit Type of Each Circuit I/O circuit type of P60-P63 changed from 13 to 13-G. Figure 2-1. Pin Input/Output Circuit List (2/2) Type 13 changed to 13-G.
p.116 p.119 p.120	CHAPTER 9 A/D CONVERTER Figure 9-1. A/D Converter Block Diagram ADIS2 and ADM3 changed to 0. Figure 9-2. A/D Converter Mode Register (ADM) Format Bit 3 modified. Figure 9-3. A/D Converter Input Select Register (ADIS) Format Bit 2 modified.
p.217 p.218 p.223 p.225	APPENDIX B DEVELOPMENT TOOLS Supported OS <ul style="list-style-type: none"> WindowsNT Ver.4.0 added Solaris added Version of HP-UX changed Figure B-1. Development Tool Configuration (1/2) (1) When using the in-circuit emulator IE-78K0-NS In-circuit emulator modified B.3.1 Hardware <ul style="list-style-type: none"> The following products added Interface adapter IE-70000-PCI-IF Performance board IE-78K0-NS-PA The following products developed In-circuit emulator IE-78K0-NS Interface adapter IE-70000-98-IF-C, IE-70000-PC-IF-C PC card interface IE-70000-CD-IF-A Emulation board IE-178018-NS-EM1 Emulation probe conversion board IE-78K0-R-EX B.3.2 Software <ul style="list-style-type: none"> The following products developed Integrated debugger ID78K0-NS
p.231	APPENDIX C EMBEDDED SOFTWARE Supported OS <ul style="list-style-type: none"> WindowsNT Ver.4.0 added Solaris added Version of HP-UX changed

The mark ★ shows major revised points.

PREFACE

Readers This manual has been prepared for user engineers who want to understand the functions of the μ PD178003 subseries and design and develop its application systems and programs.

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

Organization The μ PD178003 subseries manual is separated into two parts: this manual and the instruction edition (common to the 78K/0 series).

For the details of the μ PD178P018A, refer to **μ PD178018A User's Manual (U12742E)**.

μ PD178003 subseries User's Manual (This manual)	78K/0 series User's Manual Instruction
<ul style="list-style-type: none">• Pin functions• Internal block functions• Interrupt• Other on-chip peripheral functions	<ul style="list-style-type: none">• CPU functions• Instruction set• Explanation of each instruction

How to Read This Manual Before reading this manual, you should have general knowledge of electric and logic circuits and microcomputers.

- When you want to understand the functions in general:
 - Read this manual in the order of the contents.
- To know the μ PD178003 subseries instruction function in detail:
 - Refer to the **78K/0 series User's Manual -Instructions (U12326E)**
- How to interpret the register format:
 - For the circled bit number, the bit name is defined as a reserved word in DF178018 and RA78K0, and in CC78K0, already defined in the header file named sfrbit.h.
- To know the electrical specifications of the μ PD178003 subseries:
 - Refer to separately available Data Sheet.
 μ PD178002, 178003 Data Sheet (U12628E)
- To know the electrical specifications of the μ PD178P018A:
 - Refer to separately available Data Sheet.
 μ PD178P018A Data Sheet (U12642E)

Conventions	Data representation weight	:	High digits on the left and low digits on the right
	Active low representations	:	$\overline{\text{xxx}}$ (line over the pin and signal names)
	Note	:	Description of Note in the text.
	Caution	:	Information requiring particular attention
	Remark	:	Additional explanatory material
	Numeral representations	:	Binary ... xxxx or xxxxB Decimal ... xxxx Hexadecimal ... xxxxH

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

● **Related documents for μ PD178003 subseries**

Document Name		Document No.	
		Japanese	English
μ PD178002, 178003 Data Sheet		U12628J	U12628E
μ PD178P018A Data Sheet		U12642J	U12642E
μ PD178003 Subseries User's Manual		U13033J	This manual
μ PD178018A Subseries User's Manual		U12742J	U12742E
78K/0 Series User's Manual—Instruction		U12326J	U12326E
78K/0 Series Instruction Table		U10903J	—
78K/0 Series Instruction Set		U10904J	—
78K/0 Series Application Note	Basic II	U10121J	U10121E

● **Development Tool Documents (User's Manuals)**

Document Name		Document Number	
		Japanese	English
RA78K0 Assembler Package	Operation	U11802J	U11802E
	Assembly Language	U11801J	U11801E
	Structured Assembly Language	U11789E	U11789E
RA78K Series Structured Assembler Preprocessor		U12323J	EEU-1402
CC78K0 C Compiler	Operation	U11517J	U11517E
	Language	U11518J	U11518E
PG-1500 PROM Programmer		U11940J	U11940E
PG-1500 Controller PC-9800 Series (MS-DOS™) Base		EEU-704	EEU-1291
PG-1500 Controller IBM PC Series (PC DOS™) Base		EEU-5008	U10540E
★	IE-78K0-NS	U13731J	—
	IE-78001-R-A	Planned	Planned
	IE-78K0-R-EX1	Planned	Planned
★	IE-178018-NS-EM1	U14012J	U14012E
	IE-178018-R-EM	U10668J	U10668E
EP-78230		EEU-985	EEU-1515
	SM78K0 System Simulator Windows™ Base	Reference	U10181J
	SM78K Series System Simulator	External Part User Open Interface Specifications	U10092J
	ID78K/0 Integrated Debugger EWS Base	Reference	U11151J
	ID78K0 Integrated Debugger PC Base	Reference	U11539J
	ID78K0 Integrated Debugger Windows Base	Guide	U11649J
★	ID78K0-NS Integrated Debugger Windows Base	Reference	U12900J

Caution The above documents are subject to change without prior notice. Be sure to use the latest version document when starting design.

● Documents for Embedded Software (User's Manual)

Document Name		Document No.	
		Japanese	English
78K/0 Series Real-Time OS	Fundamentals	U11537J	U11537E
	Installation	U11536J	U11536E
78K/0 Series OS MX78K0	Fundamental	U12257J	U12257E

● Other Documents

	Document Name	Document No.	
		Japanese	English
★	SEMICONDUCTORS SELECTION GUIDE Products & Packages (CD-ROM)	X13769X	
	Semiconductor Device Mounting Technology Manual	C10535J	C10535E
	Quality Grade on NEC Semiconductor Devices	C11531J	C11531E
	Reliability Quality Control on NEC Semiconductor Devices	C10983J	C10983E
	Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892J	C11892E
	Guide to Microcomputer-Related Products by Third Party	U11416J	—

Caution The above documents are subject to change without prior notice. Be sure to use the latest version document when starting design.

CONTENTS

CHAPTER 1 OUTLINE	21
1.1 Features	21
1.2 Applications	22
1.3 Ordering Information	22
1.4 Pin Configuration (Top View)	23
1.5 Development of μPD178003 Subseries and μPD178018A Subseries	25
1.6 Block Diagram	26
1.7 Outline of Function	27
CHAPTER 2 PIN FUNCTION	29
2.1 Pin Function List	29
2.1.1 Port pins	29
2.1.2 Pins other than port pins	30
2.2 Description of Pin Functions	31
2.2.1 P00 to P06 (Port 0)	31
2.2.2 P10 to P15 (Port 1)	31
2.2.3 P20 to P27 (Port 2)	31
2.2.4 P30 to P37 (Port 3)	32
2.2.5 P40 to P47 (Port 4)	32
2.2.6 P50 to P57 (Port 5)	32
2.2.7 P60 to P67 (Port 6)	32
2.2.8 P120 to P125 (Port 12)	32
2.2.9 P132 to P134 (Port 13)	32
2.2.10 EO0, EO1	32
2.2.11 VCOL, VCOH	33
2.2.12 AMIFC	33
2.2.13 FMIFC	33
2.2.14 $\overline{\text{RESET}}$	33
2.2.15 X1 and X2	33
2.2.16 REGOSC	33
2.2.17 REGCPU	33
2.2.18 VDD	33
2.2.19 GND	33
2.2.20 VDDPORT	33
2.2.21 GNDPORT	33
2.2.22 VDDPLL	33
2.2.23 GNDPLL	34
2.2.24 IC	34
2.3 Input/output Circuits and Recommended Connection of Unused Pins	35

CHAPTER 3 CPU ARCHITECTURE	39
3.1 Memory Spaces	39
3.1.1 Internal program memory space	41
3.1.2 Internal data memory space	41
3.1.3 Special Function Register (SFR) area	41
3.1.4 Data memory addressing	42
3.2 Processor Registers	44
3.2.1 Control registers	44
3.2.2 General registers	47
3.2.3 Special Function Register (SFR)	48
3.3 Instruction Address Addressing	51
3.3.1 Relative Addressing	51
3.3.2 Immediate addressing	52
3.3.3 Table indirect addressing	53
3.3.4 Register addressing	54
3.4 Operand Address Addressing	55
3.4.1 Implied addressing	55
3.4.2 Register addressing	56
3.4.3 Direct addressing	57
3.4.4 Short direct addressing	58
3.4.5 Special Function Register (SFR) addressing	60
3.4.6 Register indirect addressing	61
3.4.7 Based addressing	62
3.4.8 Based indexed addressing	63
3.4.9 Stack addressing	63
CHAPTER 4 PORT FUNCTIONS	65
4.1 Port Functions	65
4.2 Port Configuration	67
4.2.1 Port 0	67
4.2.2 Port 1	69
4.2.3 Port 2	70
4.2.4 Port 3	72
4.2.5 Port 4	73
4.2.6 Port 5	74
4.2.7 Port 6	75
4.2.8 Port 12	76
4.2.9 Port 13	77
4.3 Port Function Control Registers	78
4.4 Port Function Operations	82
4.4.1 Writing to input/output port	82
4.4.2 Reading from input/output port	82
4.4.3 Operations on input/output port	82

CHAPTER 5 CLOCK GENERATOR	83
5.1 Clock Generator Functions	83
5.2 Clock Generator Configuration	83
5.3 Clock Generator Control Register	84
5.4 System Clock Oscillator	87
5.4.1 System clock oscillator	87
5.4.2 Scaler	89
5.5 Clock Generator Operations	90
5.6 Changing System Clock and CPU Clock Settings	91
5.6.1 Time required for switchover between system clock and CPU clock	91
5.6.2 System clock and CPU clock switching procedure	92
CHAPTER 6 8-BIT TIMER/EVENT COUNTERS 1 AND 2	93
6.1 8-Bit Timer/Event Counters 1 and 2 Functions	93
6.1.1 8-bit timer/event counter mode	93
6.1.2 16-bit timer/event counter mode	95
6.2 8-Bit Timer/Event Counters 1 and 2 Configurations	96
6.3 8-Bit Timer/Event Counters 1 and 2 Control Registers	97
6.4 8-Bit Timer/Event Counters 1 and 2 Operations	100
6.4.1 8-bit timer/event counter mode	100
6.4.2 16-bit timer/event counter mode	104
6.5 Cautions on 8-Bit Timer/Event Counters	107
CHAPTER 7 BASIC TIMER	109
7.1 Function of Basic Timer	109
7.2 Configuration of Basic Timer	109
7.3 Operation of Basic Timer	109
CHAPTER 8 BUZZER OUTPUT CONTROL CIRCUIT	111
8.1 Buzzer Output Control Circuit Functions	111
8.2 Buzzer Output Control Circuit Configuration	111
8.3 Buzzer Output Function Control Registers	112
CHAPTER 9 A/D CONVERTER	115
9.1 A/D Converter Functions	115
9.2 A/D Converter Configuration	115
9.3 A/D Converter Control Registers	118
9.4 A/D Converter Operations	121
9.4.1 Basic operations of A/D converter	121
9.4.2 Input voltage and conversion results	123
9.4.3 A/D converter operating mode	124
9.5 A/D Converter Cautions	125

CHAPTER 10 SERIAL INTERFACE CHANNEL 1	127
10.1 Serial Interface Channel 1 Functions	127
10.2 Serial Interface Channel 1 Configuration	127
10.3 Serial Interface Channel 1 Control Registers	130
10.4 Serial Interface Channel 1 Operations	132
10.4.1 Operation stop mode	132
10.4.2 3-wire serial I/O mode operation	133
CHAPTER 11 INTERRUPT AND TEST FUNCTIONS	135
11.1 Interrupt Function Types	135
11.2 Interrupt Sources and Configuration	136
11.3 Interrupt Function Control Registers	139
11.4 Interrupt Service Operations	147
11.4.1 Maskable interrupt request acknowledge operation	147
11.4.2 Software interrupt request acknowledge operation	149
11.4.3 Nesting	150
11.4.4 Pending interrupt requests	153
11.5 Test Functions	154
11.5.1 Registers controlling the test function	154
11.5.2 Test input signal acknowledge operation	155
CHAPTER 12 PLL FREQUENCY SYNTHESIZER	157
12.1 Function of PLL Frequency Synthesizer	157
12.2 Configuration of PLL Frequency Synthesizer	158
12.3 Registers Controlling PLL Frequency Synthesizer	160
12.4 Operation of PLL Frequency Synthesizer	164
12.4.1 Operation of each block of PLL frequency synthesizer	164
12.4.2 Operation to set N value of PLL frequency synthesizer	168
12.5 PLL Disable Status	173
12.6 Notes on PLL Frequency Synthesizer	173
CHAPTER 13 FREQUENCY COUNTER	175
13.1 Function of Frequency Counter	175
13.2 Configuration of Frequency Counter	175
13.3 Registers Controlling Frequency Counter	177
13.4 Operation of Frequency Counter	179
13.5 Notes on Frequency Counter	181
CHAPTER 14 STANDBY FUNCTION	183
14.1 Standby Function and Configuration	183
14.1.1 Standby function	183
14.1.2 Standby function control register	184

14.2	Standby Function Operations	185
14.2.1	HALT mode	185
14.2.2	STOP mode	188
CHAPTER 15	RESET FUNCTION	191
15.1	Reset Function	191
15.2	Power Failure Detection Function	198
CHAPTER 16	INSTRUCTION SET	199
16.1	Legends	200
16.1.1	Operand symbols and description	200
16.1.2	Description of “operation” column	201
16.1.3	Description of “flag operation” column	201
16.2	Operation List	202
16.3	Instructions Listed by Addressing Type	210
APPENDIX A	DIFFERENCES BETWEEN μ PD178002, 178003 AND μ PD178P018A	215
APPENDIX B	DEVELOPMENT TOOLS	217
B.1	Language Processing Software	220
B.2	PROM Programming Tools	222
B.2.1	Hardware	222
B.2.2	Software	222
B.3	Debugging Tools	223
B.3.1	Hardware	223
B.3.2	Software	225
B.4	System Upgrade from Former In-circuit Emulator for 78K/0 Series to IE-78001-R-A	227
APPENDIX C	EMBEDDED SOFTWARE	231
APPENDIX D	REVISION HISTORY	233

LIST OF FIGURES (1/3)

Figure No.	Title	Page
2-1	Pin Input/Output Circuit List	36
3-1	Memory Map (μ PD178002)	39
3-2	Memory Map (μ PD178003)	40
3-3	Data Memory Addressing (μ PD178002)	42
3-4	Data Memory Addressing (μ PD178003)	43
3-5	Program Counter Configuration	44
3-6	Program Status Word Configuration	44
3-7	Stack Pointer Configuration	46
3-8	Data to be Saved to Stack Memory	46
3-9	Data to be Restored from Stack Memory	46
3-10	General Register Configuration	47
4-1	Port Types	65
4-2	P00 Block Diagram	68
4-3	P01 to P06 Block Diagram	68
4-4	P10 to P15 Block Diagram	69
4-5	P20, P21, P23 to P27 Block Diagram	70
4-6	P22 Block Diagram	71
4-7	P30 to P37 Block Diagram	72
4-8	P40 to P47 Block Diagram	73
4-9	Falling Edge Detection Circuit Block Diagram	73
4-10	P50 to P57 Block Diagram	74
4-11	P60 to P67 Block Diagram	75
4-12	P120 to P125 Block Diagram	76
4-13	P132 to P134 Block Diagram	77
4-14	Port Mode Register Format	79
4-15	Port Mode Register 4 (MM) Format	80
4-16	Key Return Mode Register (KRM) Format	81
5-1	Clock Generator Block Diagram	83
5-2	Processor Clock Control Register (PCC) Format	84
5-3	Oscillation Mode Selection Register (OSMS) Format	86
5-4	System Clock Waveform due to Writing to OSMS	86
5-5	External Circuit of System Clock Oscillator	87
5-6	Examples of Resonator with Bad Connection	88
5-7	System Clock and CPU Clock Switching	92
6-1	8-Bit Timer/Event Counters 1 and 2 Block Diagram	96
6-2	Timer Clock Select Register 1 (TCL1) Format	98
6-3	8-Bit Timer Mode Control Register (TMC1) Format	99
6-4	Interval Timer Operation Timings	100
6-5	External Event Counter Operation Timings (with Rising Edge Specified)	103

LIST OF FIGURES (2/3)

Figure No.	Title	Page
6-6	Interval Timer Operation Timing	104
6-7	External Event Counter Operation Timings (with Rising Edge Specified)	106
6-8	8-Bit Timer Registers 1 and 2 Start Timing	107
6-9	Event Counter Operation Timing	107
6-10	Timing after Compare Register Change during Timer Count Operation	108
7-1	Basic Timer Block Diagram	109
7-2	Basic Timer Operation Timing	109
7-3	Operating Timing to Poll TMCIF Flag	110
8-1	Buzzer Output Control Circuit Block Diagram	111
8-2	Timer Clock Select Register 2 (TCL2) Format	112
8-3	Port Mode Register 3 (PM3) Format	113
9-1	A/D Converter Block Diagram	116
9-2	A/D Converter Mode Register (ADM) Format	119
9-3	A/D Converter Input Select Register (ADIS) Format	120
9-4	A/D Converter Basic Operation	122
9-5	Relations between Analog Input Voltage and A/D Conversion Result	123
9-6	A/D Conversion	124
9-7	A/D Conversion End Interrupt Request Generation Timing	126
10-1	Serial Interface Channel 1 Block Diagram	128
10-2	Timer Clock Select Register 3 (TCL3) Format	130
10-3	Serial Operating Mode Register 1 (CSIM1) Format	131
10-4	3-Wire Serial I/O Mode Timings	134
11-1	Basic Configuration of Interrupt Function	137
11-2	Interrupt Request Flag Register (IF0L, IF0H) Format	140
11-3	Interrupt Mask Flag Register (MK0L, MK0H) Format	141
11-4	Priority Specification Flag Register (PR0L, PR0H) Format	142
11-5	External Interrupt Mode Register 0 (INTM0) Format	143
11-6	Sampling Clock Select Register (SCS) Format	144
11-7	Noise Eliminator Input/Output Timing (during rising edge detection)	145
11-8	Program Status Word (PSW) Configuration	146
11-9	Interrupt Request Acknowledge Processing Algorithm	148
11-10	Interrupt Request Acknowledge Timing (Minimum Time)	149
11-11	Interrupt Request Acknowledge Timing (Maximum Time)	149
11-12	Nesting Example	151
11-13	Pending Interrupt Request	153
11-14	Basic Configuration of Test Function	154
11-15	Key Return Mode Register (KRM) Format	155

LIST OF FIGURES (3/3)

Figure No.	Title	Page
12-1	PLL Frequency Synthesizer Block Diagram	158
12-2	PLL Mode Select Register (PLLMD) Format	160
12-3	PLL Reference Mode Register (PLLRF) Format	161
12-4	PLL Unlock FF Judge Register (PLLUL) Format	162
12-5	PLL Data Transfer Register (PLLNS) Format	163
12-6	Input Select Block and Programmable Divider Configuration	164
12-7	Reference Frequency Generator Configuration	165
12-8	Phase Comparator, Charge Pump, and Unlock FF Configuration	165
12-9	Relationship between f_r , f_n , \overline{UP} , and \overline{DW}	166
12-10	Error Out Pins Configuration	167
13-1	Frequency Counter Block Diagram	176
13-2	IF Counter Mode Select Register (IFCMD) Format	177
13-3	IF Counter Control Register (IFCR) Format	178
13-4	IF Counter Gate Judge Register (IFCJG) Format	178
13-5	Input Pin and Mode Selection Block Diagram	179
13-6	Gate Timing of Frequency Counter	180
13-7	Frequency Counter Input Pin Circuit	181
13-8	Gate Status When HALT Instruction Is Executed	181
14-1	Oscillation Stabilization Time Select Register (OSTS) Format	184
14-2	HALT Mode Release by Interrupt Generation	186
14-3	HALT Mode Release by \overline{RESET} Input	187
14-4	STOP Mode Release by Interrupt Request Generation	189
14-5	Release by STOP Mode \overline{RESET} Input	190
15-1	Reset Function Block Diagram	192
15-2	Timing of Reset Input by \overline{RESET} Input	193
15-3	Timing of Reset by Power-ON Clear	194
15-4	POC Status Register (POCS) Format	198
B-1	Development Tool Configuration	218
B-2	EV-9200GC-80 Package Drawing (For Reference Only)	228
B-3	Recommended Board Mounting Pattern EV-9200GC-80 (For Reference Only)	229

LIST OF TABLES (1/2)

Table No.	Title	Page
2-1	I/O Circuit Type of Each Circuit	35
3-1	Vector Table	41
3-2	Special Function Register List	49
4-1	Port Functions	66
4-2	Port Configuration	67
4-3	Port Mode Register and Output Latch Settings when Using Alternate Functions	78
5-1	Clock Generator Configuration	83
5-2	Relation between CPU Clock and Minimum Instruction Execution Time	85
5-3	Maximum Time Required for CPU Clock Switchover	91
6-1	8-Bit Timer/Event Counters 1 and 2 Interval Times	94
6-2	Interval Times when 8-Bit Timer/Event Counters 1 and 2 are Used as 16-Bit Timer/Event Counters	95
6-3	8-Bit Timer/Event Counters 1 and 2 Configurations	96
6-4	8-Bit Timer/Event Counter 1 Interval Time	101
6-5	8-Bit Timer/Event Counter 2 Interval Time	102
6-6	Interval Times when 2-Channel 8-Bit Timer/Event Counters (TM1 and TM2) are Used as 16-Bit Timer/Event Counter	105
8-1	Buzzer Output Control Circuit Configuration	111
9-1	A/D Converter Configuration	115
10-1	Serial Interface Channel 1 Configuration	127
11-1	Interrupt Source List	136
11-2	Various Flags Corresponding to Interrupt Request Sources	139
11-3	Times from Maskable Interrupt Request Generation to Interrupt Service	147
11-4	Interrupt Request Enabled for Nesting during Interrupt Service	150
11-5	Test Input Factor	154
11-6	Flag Corresponding to Test Input Signal	154
12-1	Division Mode, Input Pin, and Division Value	157
12-2	PLL Frequency Synthesizer Configuration	158
12-3	Error Out Output Signal	167
12-4	Operation of Each Block and Register Status in PLL Disabled Status	173
13-1	Frequency Counter Configuration	175
14-1	HALT Mode Operating Status	185
14-2	Operation after HALT Mode Release	187

LIST OF TABLES (2/2)

Table No.	Title	Page
14-3	STOP Mode Operating Status	188
14-4	Operation after STOP Mode Release	190
15-1	Hardware Status after Reset.....	196
16-1	Operand Symbols and Descriptions	200
A-1	Differences between μ PD178002, 178003 and μ PD178P018A	215
B-1	System Upgrade Method from Former In-circuit Emulator for 78K/0 Series to the IE-78001-R-A.....	227

CHAPTER 1 OUTLINE

1.1 Features

- On-chip high-capacity ROM and RAM

<div> <div>Type</div> <div>Part Number</div> </div>	Program Memory (ROM)	Data Memory
		High-Speed RAM
μ PD178002	16 Kbytes	512 bytes
μ PD178003	24 Kbytes	

- Instruction set suitable for system control
 - Bit processing across entire address space
 - Multiplication/division instructions
- General-purpose I/O ports: 62 pins
- Hardware for PLL frequency synthesizer
 - Dual modulus prescaler (130 MHz MAX.)
 - Programmable divider
 - Phase comparator
 - Charge pump
- Frequency counter
- ★ 8-bit resolution A/D converter: 3 channels
- Serial interface: clocked 1-channel
 - 3-wire serial I/O mode : 1 channel
- Timer: 3 channels
 - Basic timer (timer carry FF) : 1 channel
 - 8-bit timer/event counter : 2 channels
- Vectored interrupt source : 8 sources
 - Maskable interrupt source : 7 (internal: 5, external: 2)
 - Software interrupt source : 1
- Test input : 1 pin
- Instruction cycle : 0.44 μ s (with 4.5 MHz crystal resonator)
- Supply voltage : $V_{DD} = 4.5$ to 5.5 V (with PLL operating)
 - $V_{DD} = 3.5$ to 5.5 V (with CPU operating, at CPU clock: $f_x/2$ or less)
 - $V_{DD} = 4.5$ to 5.5 V (with CPU operating, at CPU clock: f_x)
- Power-ON clear circuit
- One-time PROM: μ PD178P018A^{Note}

Note For the μ PD178P018A, refer to μ PD178018A Subseries User's Manual.

1.2 Applications

Car stereo, home stereo systems.

1.3 Ordering Information

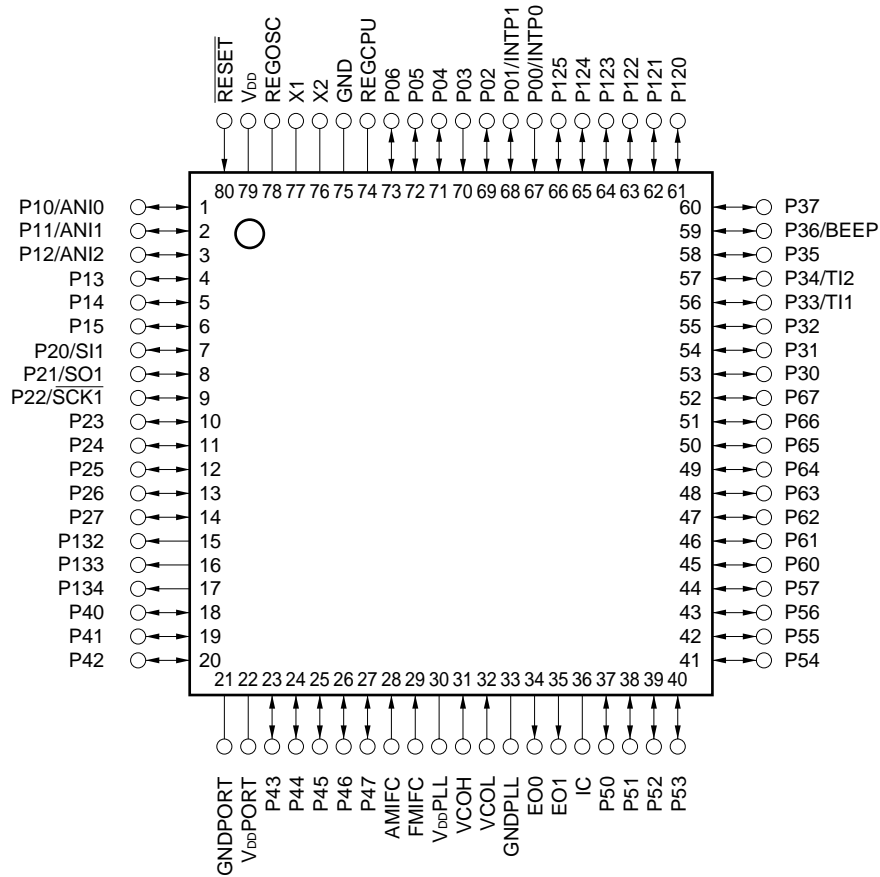
Part Number	Package	Internal ROM
μ PD178002GC-xxx-3B9	80-pin plastic QFP (14 × 14 mm, 0.65 mm pitch)	Mask ROM
μ PD178003GC-xxx-3B9	80-pin plastic QFP (14 × 14 mm, 0.65 mm pitch)	Mask ROM

Remark xxx indicates the ROM code suffix.

1.4 Pin Configuration (Top View)

• 80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)

μPD178002GC-xxx-3B9, 178003GC-xxx-3B9

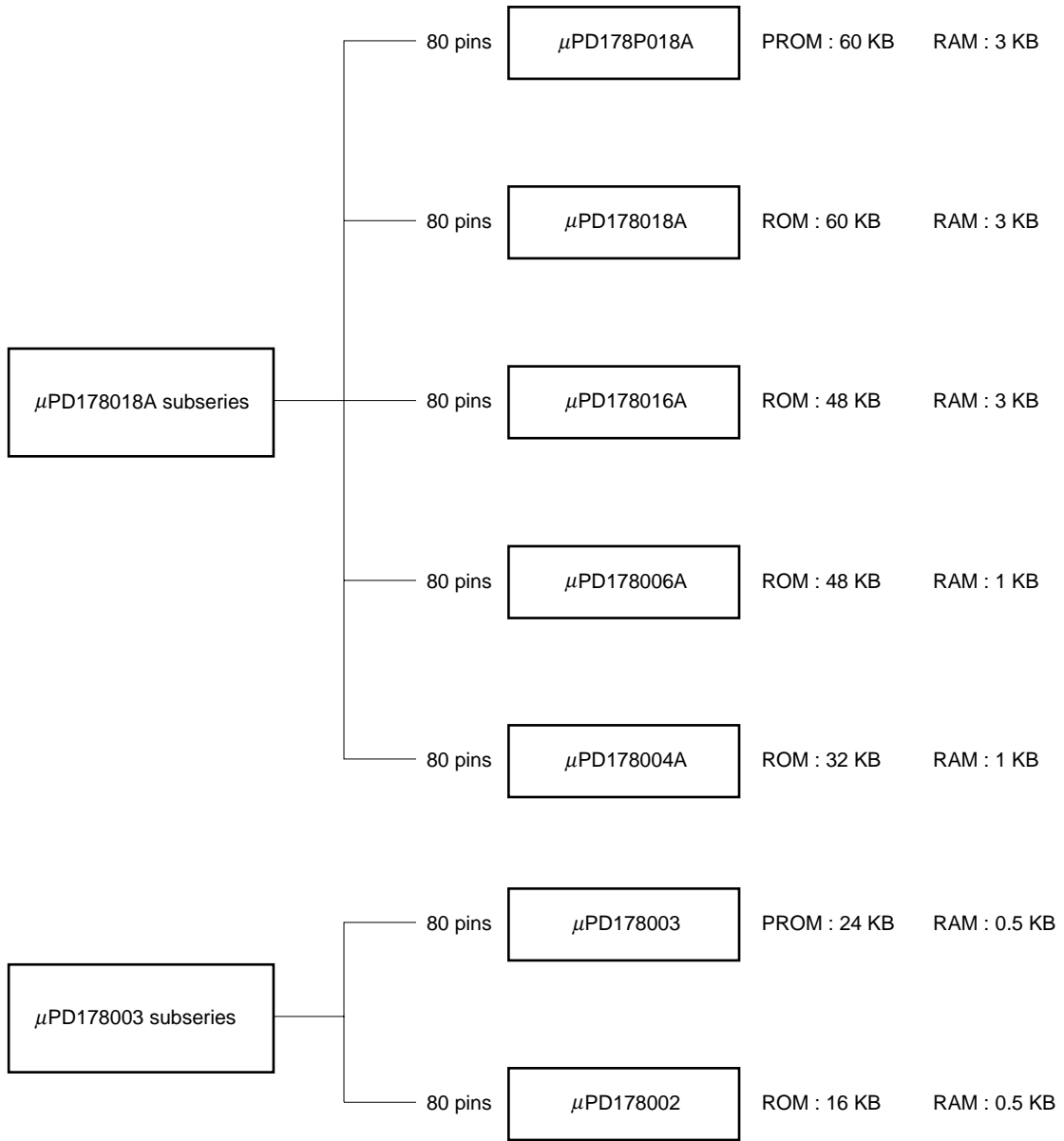


- Cautions**
1. Be sure to connect IC (Internally Connected) pin to GND directly.
 2. Set the potential of VDDPORT and VDDPLL pins to the same as that of VDD.
 3. Set the potential of the GNDPORT and GNDPLL pins to the same as that of GND.
 4. Connect each of the REGOSC and REGCPU pins to GND via a 0.1 μF capacitor.

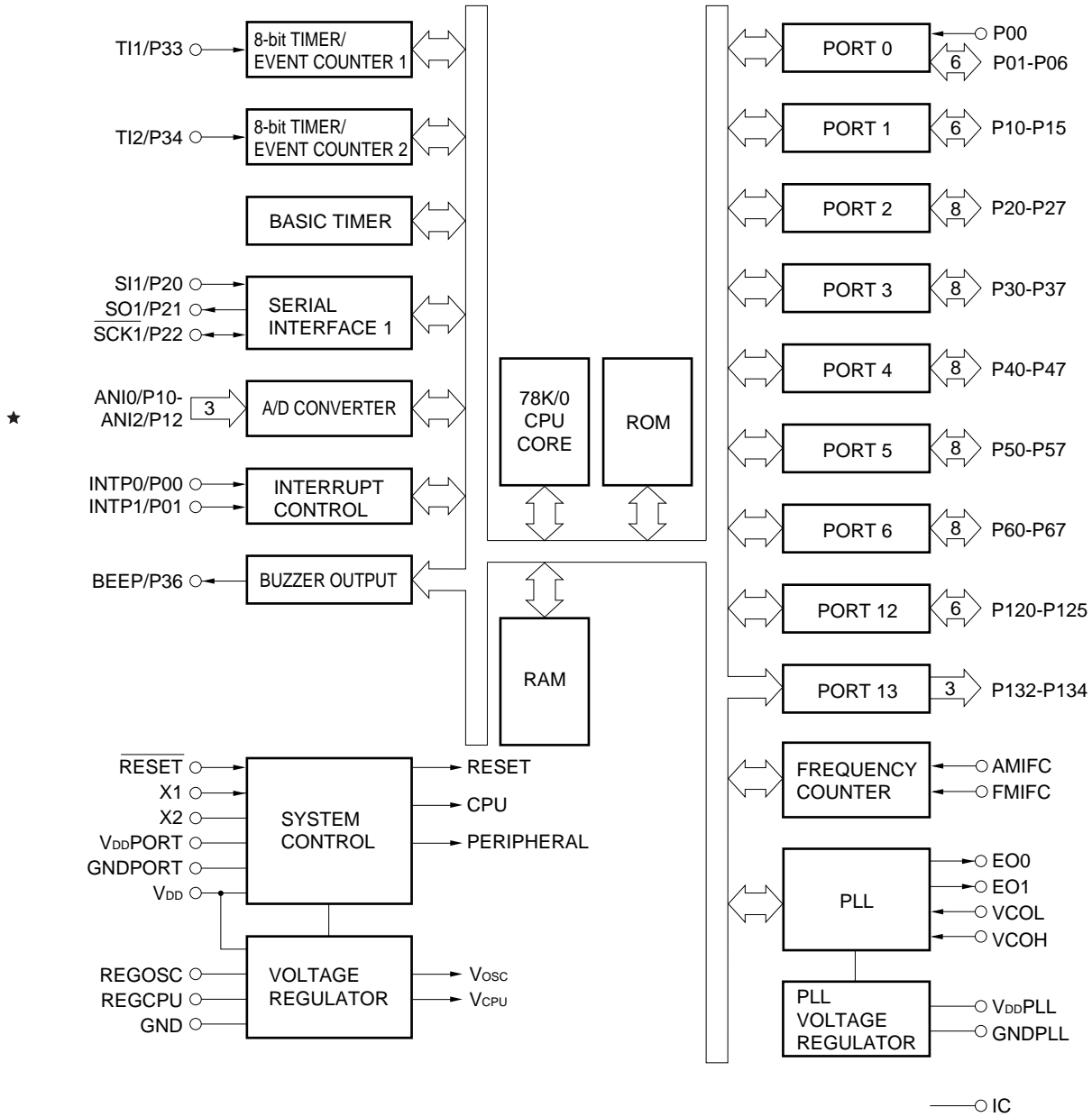
Pin Name

AMIFC	: AM intermediate frequency counter input	P60-P67	: Port 6
★ ANI0-ANI2	: A/D converter input	P120-P125	: Port 12
BEEP	: Buzzer output	P132-P134	: Port 13
EO0, EO1	: Error out output	REGCPU	: CPU power supply regulator
FMIFC	: FM intermediate frequency counter input	REGOSC	: Oscillator regulator
GND	: Ground	RESET	: Reset input
GNDPLL	: PLL ground	SCK1	: Serial clock I/O
GNDPORT	: Port ground	SI1	: Serial data input
IC	: Internally connected	SO1	: Serial data output
INTP0, INTP1	: Interrupt input	TI1, TI2	: Timer clock input
P00-P06	: Port 0	VCOL, VCOH	: Local oscillation input
P10-P15	: Port 1	VDD	: Power supply
P20-P27	: Port 2	VDDPLL	: PLL power supply
P30-P37	: Port 3	VDDPORT	: Port power supply
P40-P47	: Port 4	X1, X2	: Crystal resonator connection
P50-P57	: Port 5		

1.5 Development of μ PD178003 Subseries and μ PD178018A Subseries



1.6 Block Diagram



Remark The internal ROM capacity depends on the product.

1.7 Outline of Function

Part Number		μ PD178002	μ PD178003
Internal memory	ROM (ROM structure)	16K bytes (mask ROM)	24K bytes (mask ROM)
	High-speed RAM	512 bytes	
General-purpose register		8 bits \times 32 registers (8 bits \times 8 registers \times 4 banks)	
Minimum instruction execution time		0.44 μ s/0.88 μ s/1.78 μ s/3.56 μ s/7.11 μ s/14.22 μ s (with 4.5 MHz crystal resonator)	
Instruction set		<ul style="list-style-type: none"> • 16-bit operation • Multiplication/division (8 bits \times 8 bits, 16 bits \div 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc. 	
I/O port		Total : 62 pins CMOS input : 1 pin CMOS I/O : 54 pins N-ch open-drain I/O : 4 pins N-ch open-drain output : 3 pins	
A/D converter		8-bit resolution \times 3 channels	
Serial interface		3-wire serial I/O mode : 1 channel	
Timer		<ul style="list-style-type: none"> • Basic timer (timer carry FF (10 Hz)) : 1 channel • 8-bit timer/event counter : 2 channels 	
Buzzer (BEEP) output		1.5 kHz, 3 kHz, 6 kHz	
Vectored interrupt source	Maskable	Internal: 5, external: 2	
	Software	Internal: 1	
Test input		Internal: 1	
PLL frequency synthesizer	Division mode	Two types <ul style="list-style-type: none"> • Direct division mode (VCOL pin) • Pulse swallow mode (VCOH and VCOL pins) 	
	Reference frequency	7 types selectable by program (1, 3, 5, 9, 10, 25, 50 kHz)	
	Charge pump	Error out output: 2	
	Phase comparator	Unlock detectable by program	
Frequency counter		<ul style="list-style-type: none"> • Frequency measurement <ul style="list-style-type: none"> • AMIFC pin: for 450 kHz count • FMIFC pin: for 450 kHz/10.7 MHz counter 	
Standby function		<ul style="list-style-type: none"> • HALT mode • STOP mode 	
Reset		<ul style="list-style-type: none"> • Reset by $\overline{\text{RESET}}$ pin • Reset by power-ON clear circuit (3-value detection) <ul style="list-style-type: none"> • Detection of less than 4.5 V^{Note} (CPU clock: f_x) • Detection of less than 3.5 V^{Note} (CPU clock: $f_x/2$ or less and on power application) • Detection of less than 2.5 V^{Note} (in STOP mode) 	
Supply voltage		<ul style="list-style-type: none"> • $V_{DD} = 4.5$ to 5.5 V (with PLL operating) • $V_{DD} = 3.5$ to 5.5 V (with CPU operating, CPU clock: $f_x/2$ or less) • $V_{DD} = 4.5$ to 5.5 V (with CPU operating, CPU clock: f_x) 	
Package		80-pin plastic QFP (14 \times 14 mm, 0.65 mm pitch)	
One-time PROM		μ PD178P018A	

Note For these values, refer to **CHAPTER 15 RESET FUNCTION**.

[MEMO]

CHAPTER 2 PIN FUNCTION

2.1 Pin Function List

2.1.1 Port pins

Pin Name	I/O	Function		After Reset	Alternative Function
P00	Input	Port 0.	Input only	Input	INTP0
P01	I/O	7-bit input/output port.	Input/output mode can be specified bit-wise.	Input	INTP1
P02-P06					—
P10-P12	I/O	Port 1. 6-bit input/output port. Input/output mode can be specified bit-wise.		Input	ANI0-ANI2
P13-P15					—
P20	I/O	Port 2. 8-bit input/output port. Input/output mode can be specified bit-wise.		Input	SI1
P21					SO1
P22					SCK1
P23-P27					—
P30-P32	I/O	Port 3. 8-bit input/output port. Input/output mode can be specified bit-wise.		Input	—
P33					TI1
P34					TI2
P35					—
P36					BEEP
P37					—
P40-P47	I/O	Port 4. 8-bit input/output port. Input/output mode can be specified in 8-bit units. Test input flag (KRIF) is set to 1 by falling edge detection.		Input	—
P50-P57	I/O	Port 5. 8-bit input/output port. Input/output mode can be specified bit-wise.		Input	—
P60-P63	I/O	Port 6. 8-bit input/output port.	N-ch open-drain input/output port. LEDs can be driven directly.	Input	—
P64-P67		Input/output mode can be specified bit-wise.			
P120-P125	I/O	Port 12. 6-bit input/output port. Input/output mode can be specified bit-wise.		Input	—
P132-P134	Output	Port 13. 3-bit output port. N-ch open-drain output port.		—	—

2.1.2 Pins other than port pins

Pin Name	I/O	Function	After Reset	Alternative Function
INTP0, INTP1	Input	External maskable interrupt request inputs for which the effective edges (rising edge, falling edge, both rising and falling edges) can be input.	Input	P00, P01
SI1	Input	Serial interface serial data input	Input	P20
SO1	Output	Serial interface serial data output	Input	P21
SCK1	I/O	Serial interface serial clock input/output	Input	P22
TI1	Input	External count clock input to 8-bit timer (TM1)	Input	P33
TI2		External count clock input to 8-bit timer (TM2)		P34
BEEP	Output	Buzzer output	Input	P36
★ ANI0-ANI2	Input	A/D converter analog input	Input	P10-P12
EO0, EO1	Output	Error out output from charge pump of the PLL frequency synthesizer	—	—
VCOL	Input	Inputs PLL local band frequency (In HF, MF mode)	—	—
VCOH	Input	Inputs PLL local band frequency (In VHF mode)	—	—
AMIFC	Input	Inputs AM intermediate frequency counter	—	—
FMIFC	Input	Inputs FM or AM intermediate frequency counter	—	—
RESET	Input	System reset input	—	—
X1	Input	System clock oscillation resonator connection	—	—
X2	—		—	—
REGOSC	—	Oscillation regulator. Connect to GND via a 0.1 μ F capacitor.	—	—
REGCPU	—	CPU power supply regulator. Connect to GND via a 0.1 μ F capacitor.	—	—
V _{DD}	—	Positive power supply	—	—
GND	—	Ground	—	—
V _{DD} PORT	—	Positive power supply for port block	—	—
GNDPORT	—	Ground for port block	—	—
V _{DD} PLL ^{Note}	—	Positive power supply for PLL	—	—
GNDPLL ^{Note}	—	Ground for PLL	—	—
IC	—	Internally connected. Connect to GND or GNDPORT directly.	—	—

Note Connect 1000 pF capacitor between V_{DD}PLL pin and GNDPLL pin.

2.2 Description of Pin Functions

2.2.1 P00 to P06 (Port 0)

These are 7-bit input/output ports. Besides serving as input/output ports, they function as an external interrupt request input. The following operating modes can be specified bit-wise.

(1) Port mode

P00 functions as input-only port and P01 to P06 function as input/output ports.

P01 to P06 can be specified for input or output ports bit-wise with a port mode register 0 (PM0).

(2) Control mode

In this mode, these ports function as an external interrupt request input pin (INTP0, INTP1).

INTP0 and INTP1 are external interrupt request input pins which can specify valid edges (rising edge, falling edge, and both rising and falling edges).

2.2.2 P10 to P15 (Port 1)

These are 6-bit input/output ports. Besides serving as input/output ports, they function as an A/D converter analog input.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 6-bit input/output ports.

They can be specified bit-wise as input or output ports with a port mode register 1 (PM1).

(2) Control mode

These ports function as A/D converter analog input pins (ANI0-ANI2).

★

2.2.3 P20 to P27 (Port 2)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as data input/output to/from the serial interface and clock input/output functions.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 8-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 2 (PM2).

(2) Control mode

These ports function as serial interface data input/output and clock input/output functions.

(a) SI1, SO1

Serial interface serial data input/output pins

(b) $\overline{\text{SCK1}}$

Serial interface serial clock input/output pins

Caution When P20 to P22 are used as a serial interface, the I/O and output latches must be set according to the function the user requires. For the setting, refer to Figure 10-3 Serial Operating Mode Register 1 (CSIM1) Format.

2.2.4 P30 to P37 (Port 3)

These are 8-bit input/output ports. Beside serving as input/output ports, they function as timer input and buzzer output.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 8-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 3 (PM3).

(2) Control mode

These ports function as timer input and buzzer output (BEEP).

(a) TI1 and TI2

Pin for external clock input to the 8-bit timer/event counter.

(b) BEEP

Buzzer (BEEP) output pin.

2.2.5 P40 to P47 (Port 4)

These are 8-bit input/output ports.

The test input flag (KRIF) can be set to 1 by detecting a falling edge.

They can be specified in 8-bit units for input or output ports by using the port mode register 4 (MM).

2.2.6 P50 to P57 (Port 5)

These are 8-bit input/output ports.

Port 5 can drive LEDs directly.

They can be specified bit-wise as input/output ports with port mode register 5 (PM5).

2.2.7 P60 to P67 (Port 6)

These are 8-bit input/output ports. P60 to P63 can drive LEDs directly.

They can be specified bit-wise as input or output ports with port mode register 6 (PM6).

P60 to P63 are N-ch open drain outputs.

2.2.8 P120 to P125 (Port 12)

These are 6-bit input/output ports.

They can be specified bit-wise as input or output ports with port mode register 12 (PM12).

2.2.9 P132 to P134 (Port 13)

These are 3-bit output ports. They are used for N-ch open-drain output.

2.2.10 EO0, EO1

These are the output pins of the charge pump of the PLL frequency synthesizer.

They output the result of phase comparison between the frequency divided by the programmable divider of the local oscillation input (VCOL and VCOH pins) and the reference frequency.

2.2.11 VCOL, VCOH

These pins input the local oscillation frequency (VCO) of the PLL.

Because signals are input to these pins via an AC amplifier, cut the DC component of the input signals by using a capacitor.

- VCOL
 - HF, MF input
 - Becomes active when the HF or AM mode is selected by program; otherwise, goes into a high-impedance state.
- VCOH
 - VHF input
 - Becomes active when the FM mode is selected by program; otherwise, goes into a high-impedance state.

2.2.12 AMIFC

This is the input pin of the AM intermediate frequency counter.

2.2.13 FMIFC

This is the input pin of the FM or AM intermediate frequency counter.

2.2.14 RESET

This is a low-level active system reset input pin.

2.2.15 X1 and X2

Crystal resonator connect pins for system clock oscillation.

2.2.16 REGOSC

Oscillator regulator pin. Connect to GND via a 0.1 μ F capacitor.

2.2.17 REGCPU

CPU power supply regulator pin. Connect to GND via a 0.1 μ F capacitor.

2.2.18 V_{DD}

Positive power supply pin.

2.2.19 GND

Ground potential pin

2.2.20 V_{DD}PORT

Positive power supply pin for port. Set the same potential as that of V_{DD} pin.

2.2.21 GNDPORT

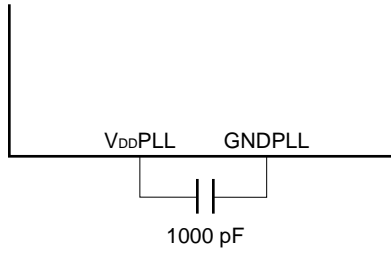
Ground potential pin for port. Set the same potential as that of GND pin.

2.2.22 V_{DD}PLL

Positive power supply pin for PLL. Connect 1000 pF capacitor between V_{DD}PLL pin and GNDPLL pin.

2.2.23 GNDPLL

Ground potential pin for PLL. Connect 1000 pF capacitor between GNDPLL pin and V_{DD}PLL pin.

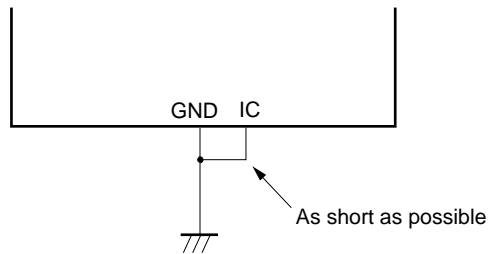


2.2.24 IC

The IC (Internally Connected) pin is provided to set the test mode to check the μ PD178003 subseries at delivery. Connect it directly to the GND pin with the shortest possible wire in the normal operating mode.

When a potential difference is produced between the IC pin and GND 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 run normally.

- **Connect IC pin to GND pin directly.**



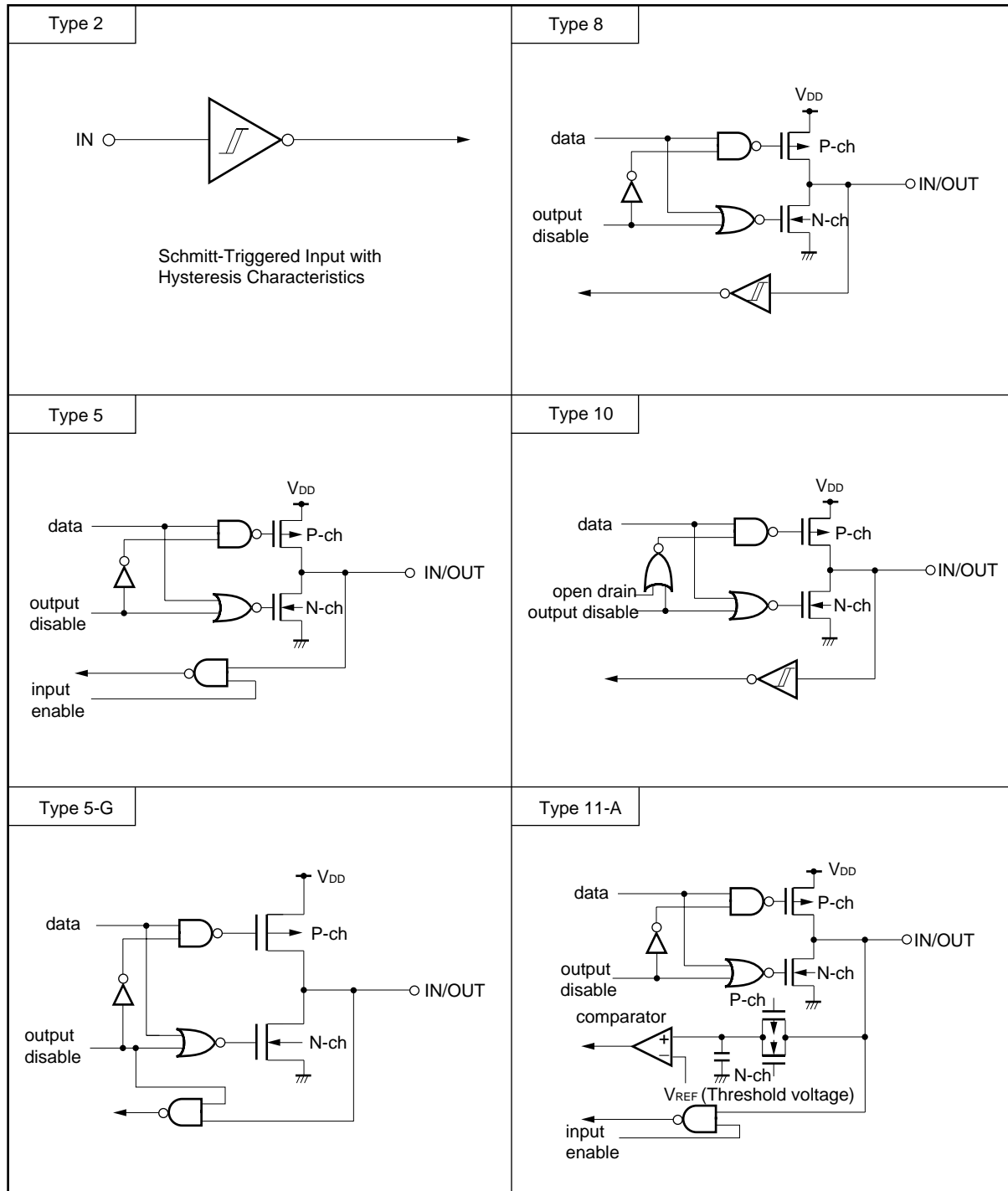
2.3 Input/output Circuits and Recommended Connection of Unused Pins

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

Table 2-1. I/O Circuit Type of Each Circuit

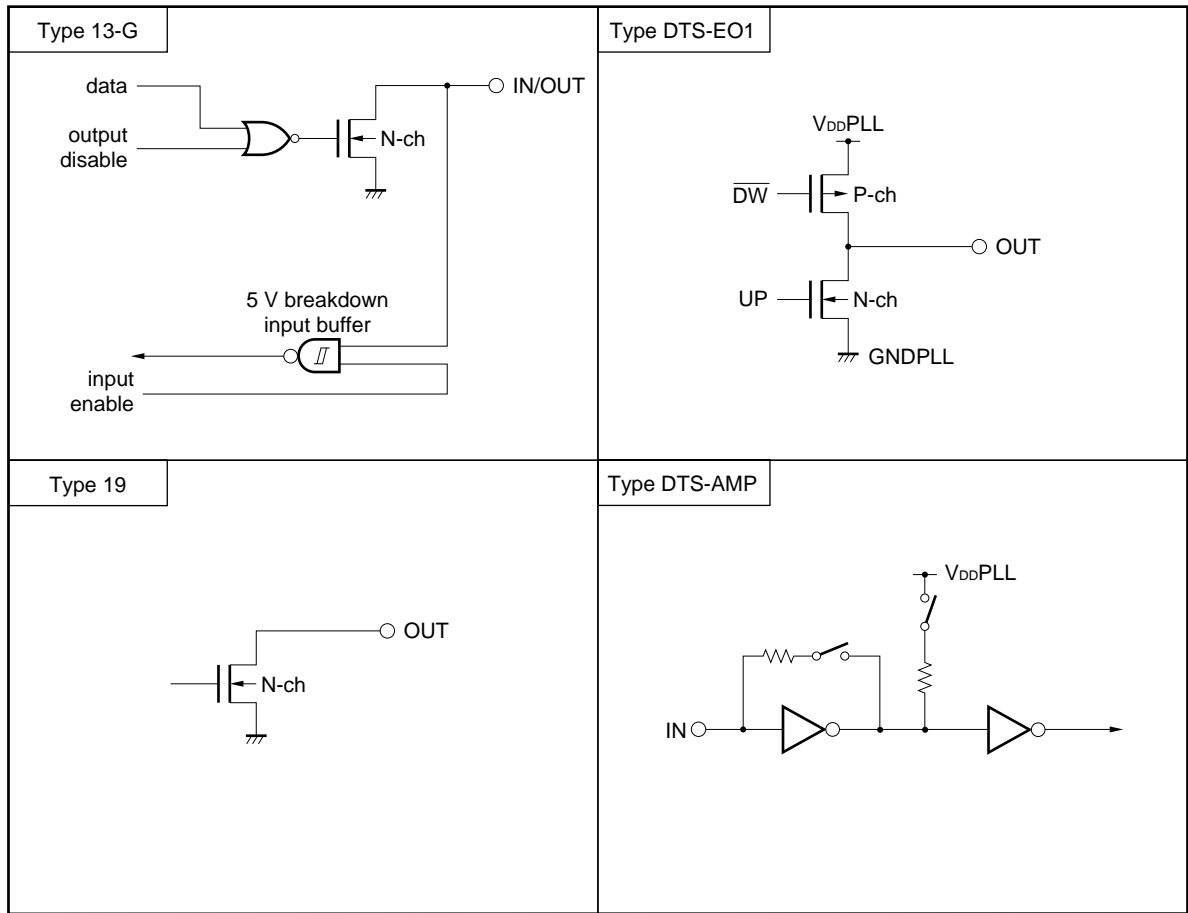
Pin Name	I/O Circuit Type	I/O	Recommended Connections of Unused Pins
P00/INTP0	2	Input	Connect to GND or GNDPORT
P01/INTP1, P02-P06	8	I/O	Set in general-purpose input port mode by software and individually connect to V _{DD} , V _{DD} PORT, GND, or GNDPORT via resistor.
P10/ANI0-P12/ANI2	11-A		
P13-P15	5		
P20/SI1	8		
P21/SO1	5		
P22/ $\overline{\text{SCK1}}$	8		
P23	5		
P24	8		
P25-P27	10		
P30-P32	5		
P33/TI1, P34/TI2	8		
P35	5		
P36/BEEP			
P37			
P40-P47	5-G		
P50-P57	5		
P60-P63	13-G		
P64-P67	5		
P120-P125			
P132-P134	19	Output	Set to low-level output by software and open
EO0, EO1	DTS-EO1		Open
VCOL, VCOH	DTS-AMP	Input	Set to disabled status by software and open
AMIFC, FMIFC			
IC	—	—	Connect to GND or GNDPORT directly

Figure 2-1. Pin Input/Output Circuit List (1/2)



Remark All V_{DD} and GND in the above figures are the positive power supply and ground potential of the ports, and should be taken as V_{DDPORT} and $GNDPORT$, respectively.

Figure 2-1. Pin Input/Output Circuit List (2/2)



Remark All V_{DD} and GND in the above figures are the positive power supply and ground potential of the ports, and should be taken as V_{DDPORT} and $GNDPORT$, respectively.

[MEMO]

CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Spaces

Figures 3-1 and 3-2 show memory maps.

Figure 3-1. Memory Map (μ PD178002)

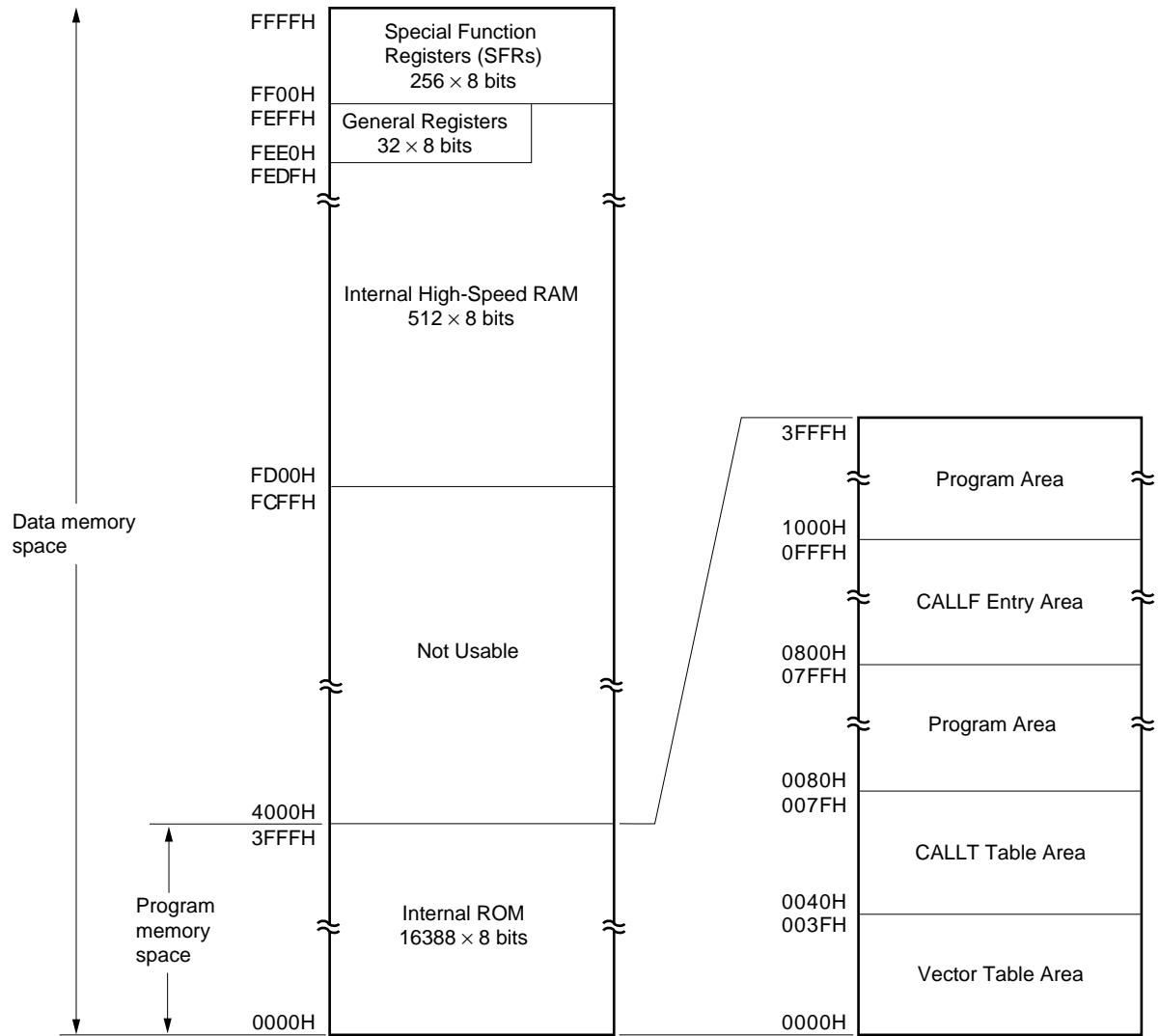
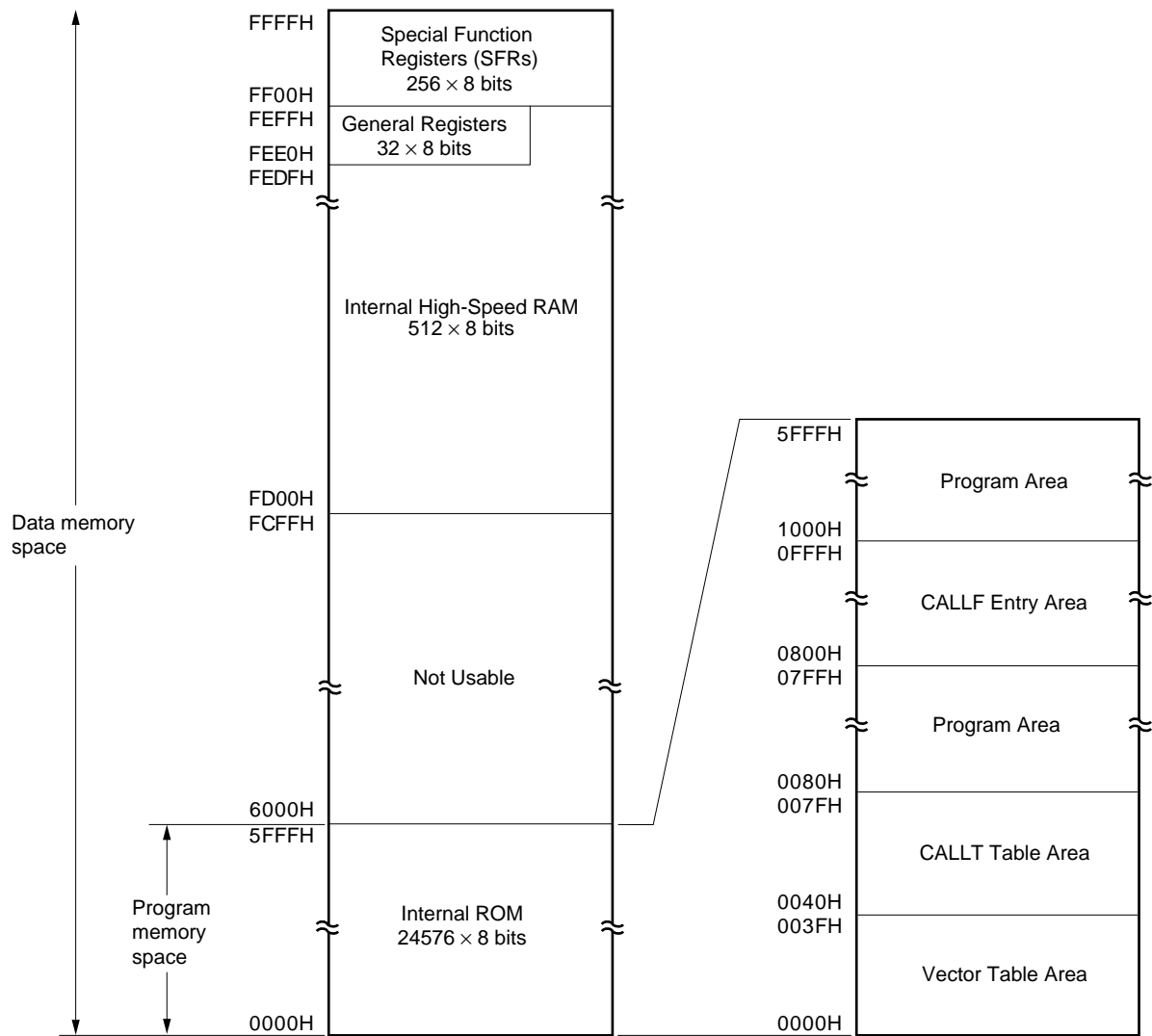


Figure 3-2. Memory Map (μ PD178003)

3.1.1 Internal program memory space

The μ PD178002 has a mask ROM of 16388×8 bits, and the μ PD178003 has a mask ROM of 24576×8 bits. It stores programs and table data. Normally, they are addressed with a program counter (PC).

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

(1) Vector table area

The 64-byte area 0000H to 003FH is reserved as a vector table area. The 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 address, low-order 8 bits are stored at even addresses and high-order 8 bits are stored at odd addresses.

Table 3-1. Vector Table

Vector Table Address	Interrupt Source
0000H	Reset input
0006H	INTP0
0008H	INTP1
0016H	INTCSI1
0018H	INTTMC
001CH	INTTM1
001EH	INTTM2
0020H	INTAD
003EH	BRK

(2) CALLT instruction table area

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

(3) CALLF instruction entry area

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

3.1.2 Internal data memory space

The μ PD178003 subseries units incorporate an internal high-speed RAM of 512×8 bits.

In this area, four banks of general registers, each bank consisting of eight 8-bit registers, are allocated in the 32-byte area FEE0H to FFFFH.

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

3.1.3 Special Function Register (SFR) area

An on-chip peripheral hardware special-function register (SFR) is allocated in the area FF00H to FFFFH. (Refer to **3.2.3 Special Function Register (SFR) Table 3-2 Special Function Register List**).

Caution Do not access addresses where the SFR is not assigned.

3.1.4 Data memory addressing

The method of specifying the address of the instruction to be executed next or the address of the register or memory location to be manipulated when an instruction is executed is called addressing.

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

To address the memory location to be manipulated when an instruction is executed, the μ PD178003 subseries offers variety of addressing modes to provide good operability. In particular, at addresses corresponding to data memory area (FD00H to FFFFH), particular addressing modes are possible to meet the functions of the special function registers (SFRs) and general registers. Figures 3-3 and 3-4 show the data memory addressing modes. For details of the addressing modes, refer to **3.4 Operand Address Addressing**.

Figure 3-3. Data Memory Addressing (μ PD178002)

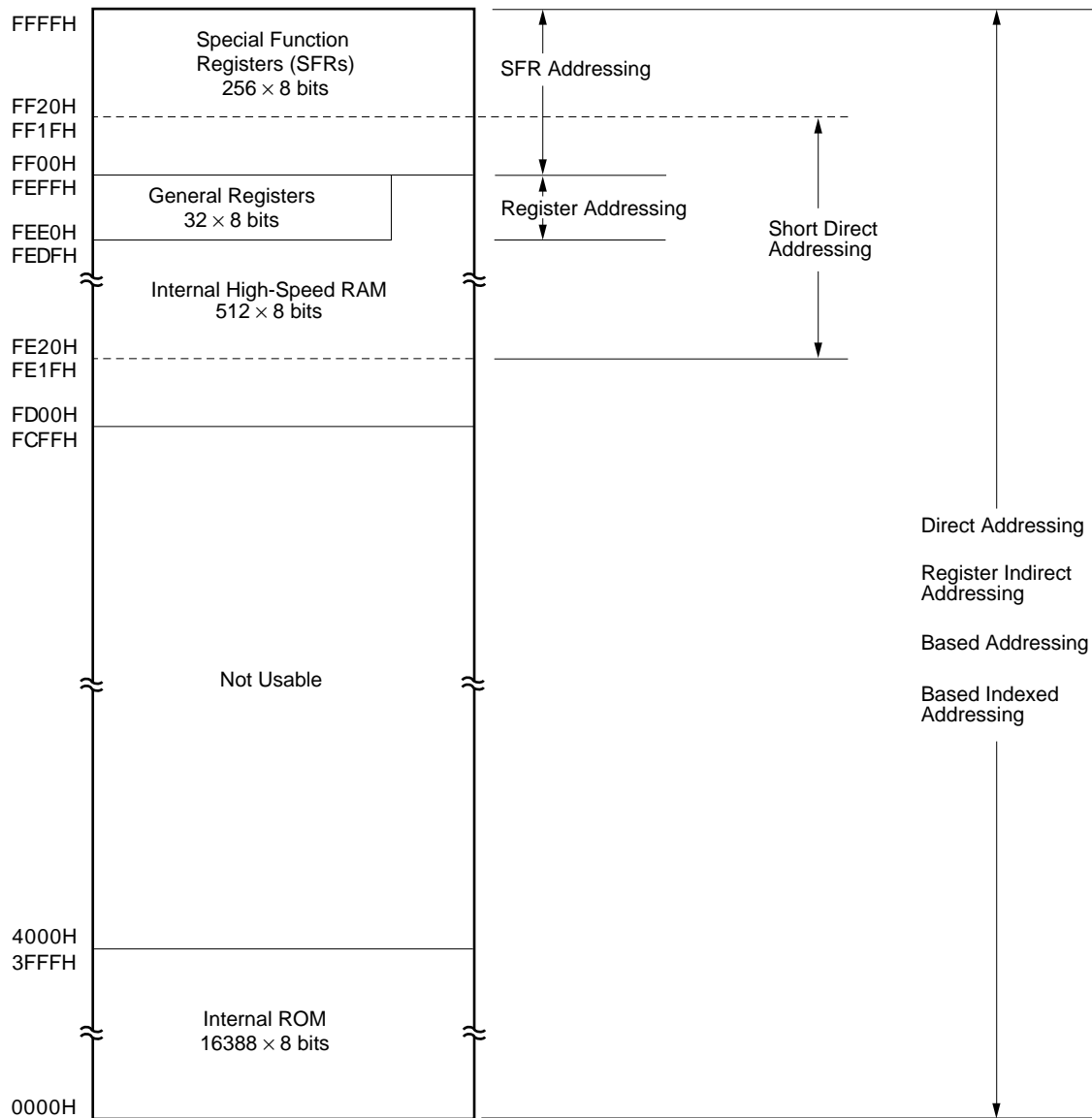
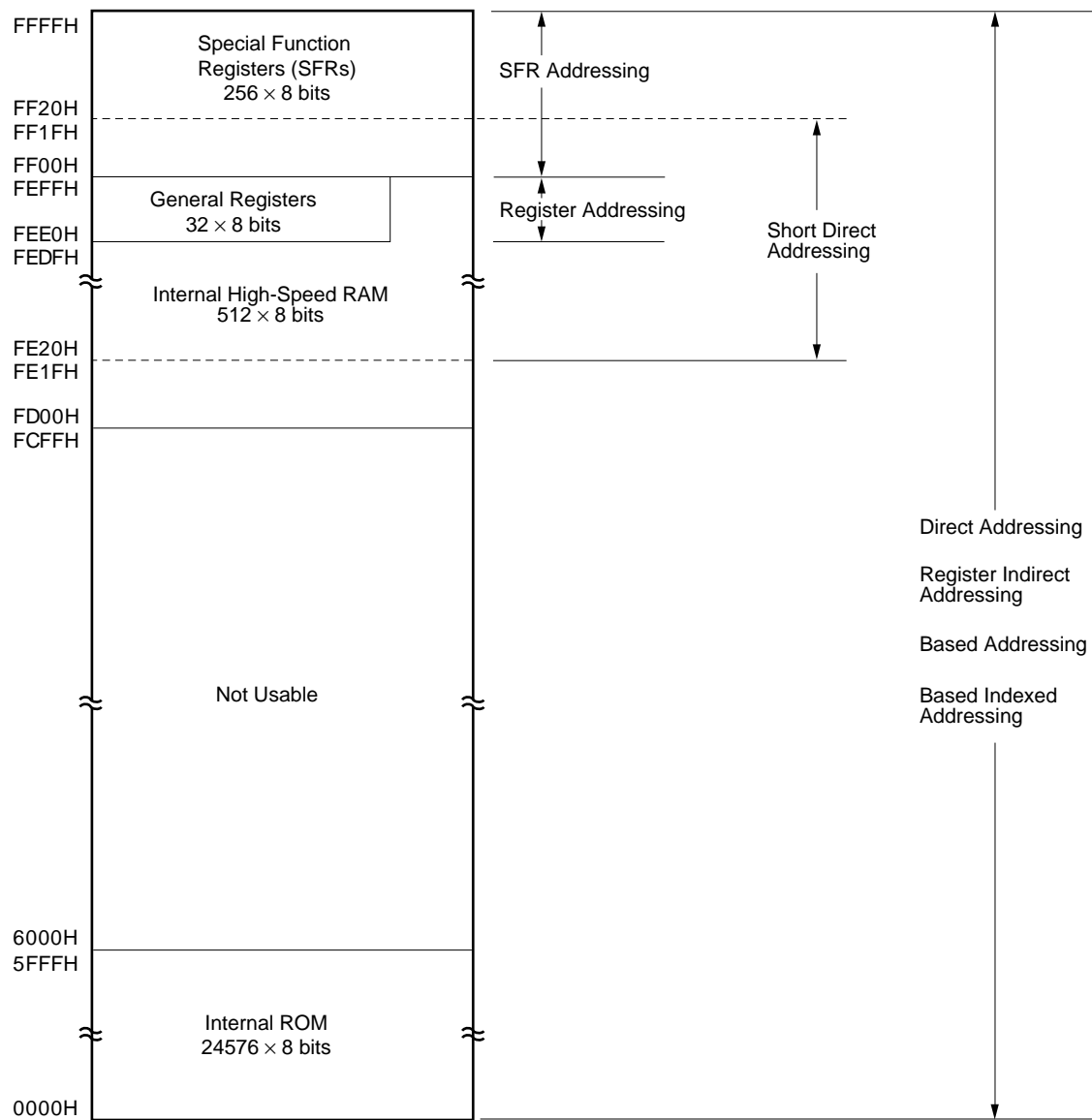


Figure 3-4. Data Memory Addressing (μ PD178003)



3.2 Processor Registers

The μ PD178003 subseries units 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).

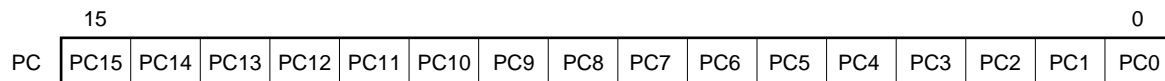
(1) Program counter (PC)

The program counter is a 16-bit register which 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.

Reset input sets the reset vector table values at addresses 0000H and 0001H to the program counter.

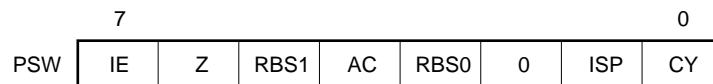
Figure 3-5. Program Counter Configuration



(2) Program status word (PSW)

The program status word is an 8-bit register consisting of various flags to be set/reset by instruction execution. Program status word contents are automatically stacked upon interrupt request generation or PUSH PSW instruction execution and are automatically restored upon execution of the RETB, RETI and POP PSW instructions. Reset input sets the PSW to 02H.

Figure 3-6. Program Status Word Configuration



(a) Interrupt enable flag (IE)

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

When IE = 0, all the interrupts are disabled (DI) except the non-maskable interrupt.

When IE = 1, the interrupts are enabled (EI). At this time, the acknowledging of interrupts is controlled by the in-service priority flag (ISP), the interrupt mask flag corresponding to each interrupt, and the interrupt priority specification flag.

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

(b) Zero flag (Z)

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

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

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

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

(d) Auxiliary carry flag (AC)

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

(e) In-service priority flag (ISP)

This flag manages the priority of acknowledgeable maskable vectored interrupts. When ISP = 0, acknowledging the vectored interrupt requests to which a low priority is assigned by the priority specification flag registers (PR0L, PR0H) (refer to **11.3 (3) Priority specification flag registers (PR0L, PR0H)**) is disabled. Whether an interrupt request is actually accepted depends on the status of the interrupt enable flag (IE).

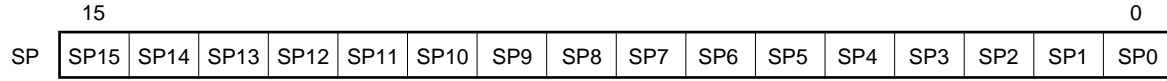
(f) Carry flag (CY)

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 to hold the start address of the memory stack area. Only the internal high-speed RAM area (FD00H-FEFFFH) can be set as the stack area.

Figure 3-7. Stack Pointer Configuration



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

Each stack operation saves/restores data as shown in Figures 3-8 and 3-9.

Caution Since reset input makes SP contents undefined, be sure to initialize the SP before instruction execution.

Figure 3-8. Data to be Saved to Stack Memory

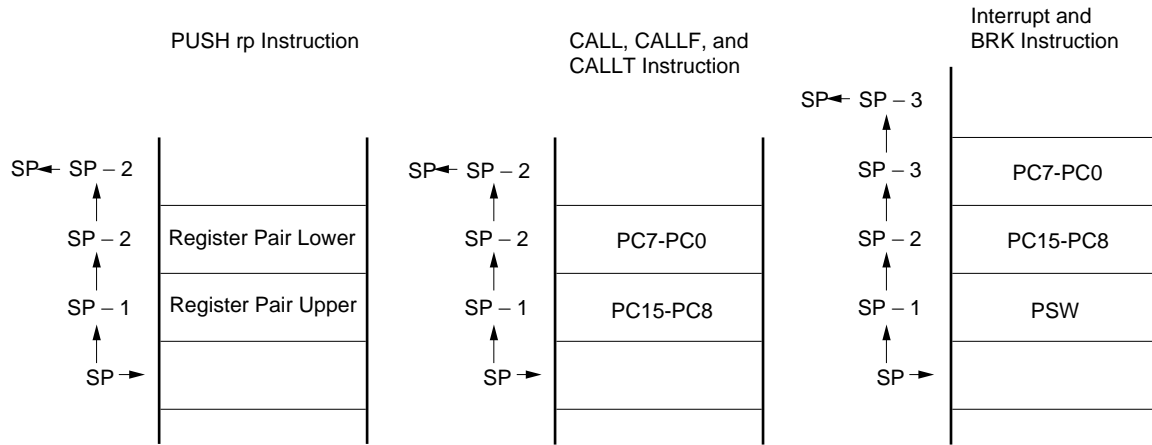
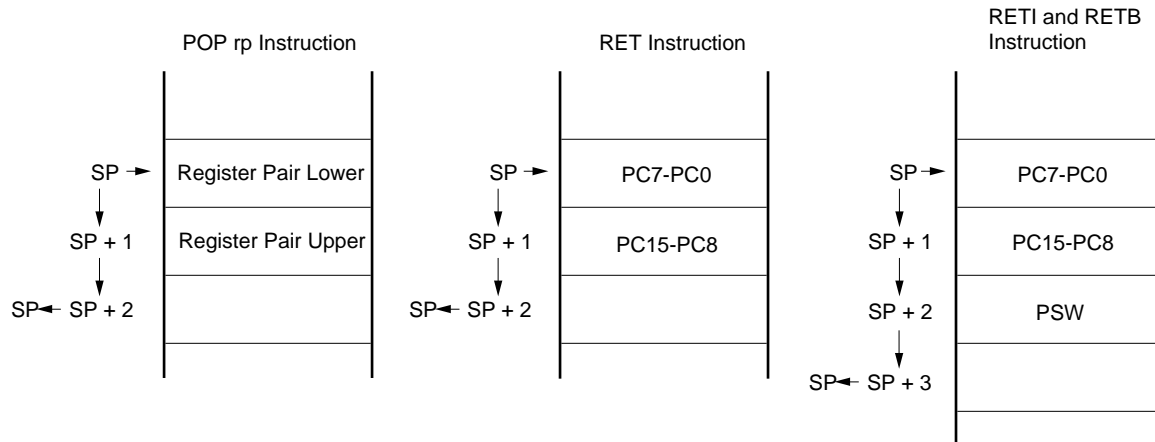


Figure 3-9. Data to be Restored from Stack Memory



3.2.2 General registers

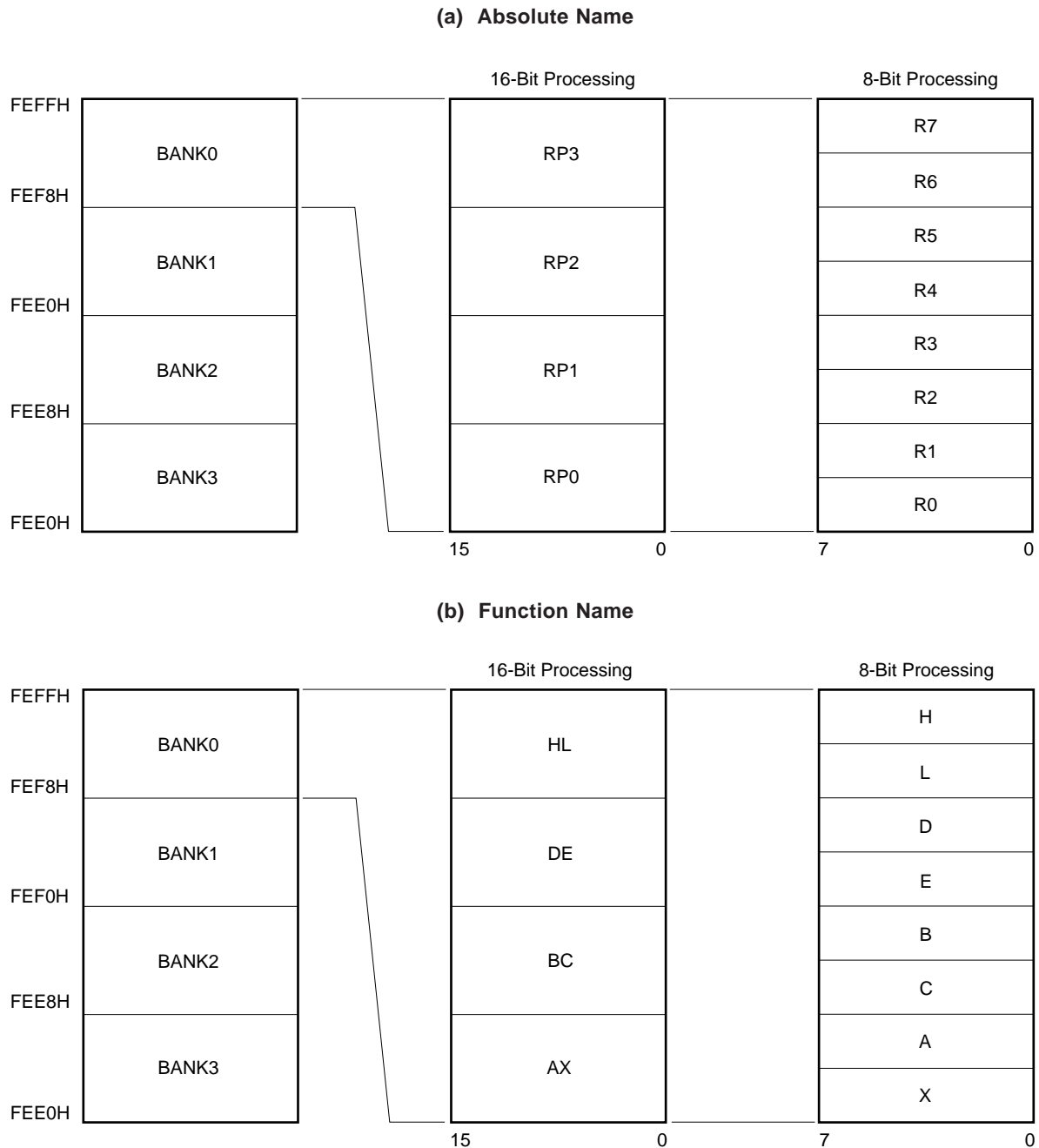
A general register is mapped at particular addresses (FEE0H to FEFFH) of the data memory. It consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L and H).

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 written 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 interrupt request for each bank.

Figure 3-10. General Register Configuration



3.2.3 Special Function Register (SFR)

Unlike a general register, each special function register has special functions.

It is allocated in the FF00H to FFFFH area.

The special function register can be manipulated like the general register, with the operation, transfer and bit manipulation instructions. Manipulatable bit units, 1, 8 and 16, depend on the special function register type.

Each manipulation bit unit can be specified as follows.

- **1-bit manipulation**

Use the symbol reserved in the assembler for the 1-bit manipulation instruction operand (sfr.bit).

This manipulation can also be specified with an address.

- **8-bit manipulation**

Use the symbol reserved in the assembler for the 8-bit manipulation instruction operand (sfr).

This manipulation can also be specified with an address.

- **16-bit manipulation**

Use the symbol reserved in the assembler for the 16-bit manipulation instruction operand (sfrp).

When addressing an address, use an even address.

Table 3-2 gives a list of special function registers. The meaning of items in the table is as follows.

- **Symbol**

This is a symbol to indicate an address of the special function register.

These symbols are reserved for the DF178018 and RA78K/0, and defined by header file sfrbit.h for the CC78K0.

They can be written as instruction operands when the RA78K0, ID78K0, ID78K0-NS, or SM78K/0 is used.

- **R/W**

Indicates whether the corresponding special function register can be read or written.

R/W : Read/write enable

R : Read only

R&Reset : Read only (reset to 0 when read)

W : Write only

- **Manipulatable bit units**

○ indicates manipulatable bit units 1, 8, and 16. – indicates the bit units that cannot be manipulated.

- **After reset**

Indicates each register status upon reset.

Table 3-2. Special Function Register List (1/2)

Address	Special Function Register (SFR) Name	Symbol	R/W	Manipulatable Bit Unit			After Reset
				1 bit	8 bits	16 bits	
FF00H	Port 0	P0	R/W	○	○	—	00H
FF01H	Port 1	P1		○	○	—	
FF02H	Port 2	P2		○	○	—	
FF03H	Port 3	P3		○	○	—	
FF04H	Port 4	P4		○	○	—	Undefined
FF05H	Port 5	P5		○	○	—	
FF06H	Port 6	P6		○	○	—	
FF0CH	Port 12	P12		○	○	—	00H
FF0DH	Port 13	P13		○	○	—	
FF16H	Compare register 10	CR10		—	○	—	Undefined
FF17H	Compare register 20	CR20		—	○	—	
FF18H	8-bit timer register 1	TMS	R	—	○	○	00H
FF19H	8-bit timer register 2			—	○		
FF1BH	Serial I/O shift register 1	SIO1	R/W	—	○	—	Undefined
FF1FH	A/D conversion result register	ADCR	R	—	○	—	
FF20H	Port mode register 0	PM0	R/W	○	○	—	FFH
FF21H	Port mode register 1	PM1		○	○	—	
FF22H	Port mode register 2	PM2		○	○	—	
FF23H	Port mode register 3	PM3		○	○	—	
FF25H	Port mode register 5	PM5		○	○	—	
FF26H	Port mode register 6	PM6		○	○	—	
FF2CH	Port mode register 12	PM12		○	○	—	
FF41H	Timer clock select register 1	TCL1		—	○	—	00H
FF42H	Timer clock select register 2	TCL2		—	○	—	
FF43H	Timer clock select register 3	TCL3		—	○	—	88H
FF47H	Sampling clock select register	SCS		—	○	—	00H
FF49H	8-bit timer mode control register	TMC1		○	○	—	
FF68H	Serial operating mode register 1	CSIM1		○	○	—	01H
FF80H	A/D converter mode register	ADM		○	○	—	
FF84H	A/D converter input select register	ADIS		—	○	—	00H
FFA0H	PLL mode select register	PLLMD		○	○	—	
FFA1H	PLL reference mode register	PLLRF		○	○	—	0FH
FFA2H	PLL unlock FF judge register	PLLUL	R&Reset	○	○	—	Held ^{Note}
FFA3H	PLL data transfer register	PLLNS	W	○	○	—	00H

Note The value of this register becomes undefined when reset is effected through power-ON clearing.

Caution Do not access the address allocated to the special function register.

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

Address	Special-Function Register (SFR) Name		Symbol		R/W	Manipulatable Bit Unit			After Reset	
						1 bit	8 bits	16 bits		
FFA6H	PLL data register	PLL data register L	PLLRL	PLLRLH	R/W	○	○	○	Undefined	
FFA7H		PLL data register H				○	○			
FFA8H	PLL data register 0		PLLRL0			○	○	—		00H
FFA9H	IF counter mode select register		IFCMD			○	○	—		
FFABH	IF counter gate judge register		IFCJG		R	○	○	—		
FFACH	IF counter control register		IFCR		W	○	○	—		
FFAEH	IF counter data register		IFC	IFR1	R	○	○	○	00H	
FFAFH				IFR2		○	○			
FFBBH	Note 1		CHECK ^{Note 1}		R/W	—	—	—		
FFBFH	POC status register		POCS		R&Reset	○	○	—	Held ^{Note}	
FFD0H to FFD7H	External access area ^{Note 3}				R/W	○	○	—	Undefined	
FFE0H	Interrupt request flag register 0L		IF0	IF0L		○	○	○	00H	
FFE1H	Interrupt request flag register 0H			IF0H		○	○			
FFE4H	Interrupt mask flag register 0L		MK0	MK0L		○	○	○	FFH	
FFE5H	Interrupt mask flag register 0H			MK0H		○	○			
FFE8H	Priority order specification flag register 0L		PR0	PR0L		○	○	○		
FFE9H	Priority order specification flag register 0H			PR0H		○	○			
FFECH	External interrupt mode register 0		INTM0			—	○	—	00H	
FFF2H	Oscillation mode selection register		OSMS			W	—	○		—
FFF6H	Key return mode register		KRM			R/W	○	○	—	02H
FFF8H	Port mode register 4		MM		○		○	—	10H	
FFFAH	Oscillation stabilizaiton time select register		OSTS		—		○	—	04H	
FFFBH	Processor clock control register		PCC		○		○	—		

Notes 1. This is a test register. Do not use the symbol of this register in software as a user-defined symbol. Do not use the symbols CHECK0, CHECK2, CHECK3, CPSTP, and PPSTP as user-defined symbols.

2. The value of this register is set to 01H only when reset is effected through power-ON clearing.

3. The external access area cannot be used in the μ PD178003 subseries.

Caution Do not access the address allocated to the special function register.

3.3 Instruction Address Addressing

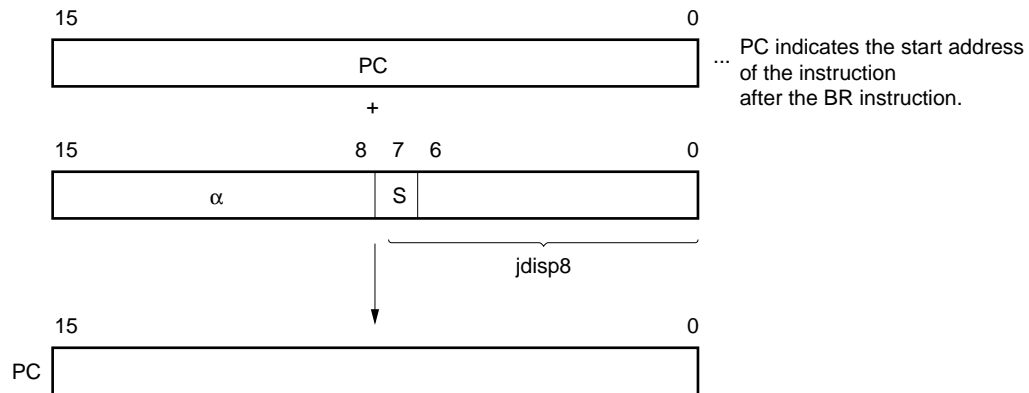
An instruction address is determined by program counter (PC) contents, and the contents is normally incremented (+1 for each byte) automatically according to the number of bytes of an instruction to be fetched each time another instruction is executed. When a branch instruction is executed, the branch destination information is set to the PC and branched by the following addressing. (For details of instructions, refer to **78K/0 User's Manual -Instruction (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. That is, using relative addressing, the program branches in the range −128 to +127 relative to the first address of the next instruction. 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

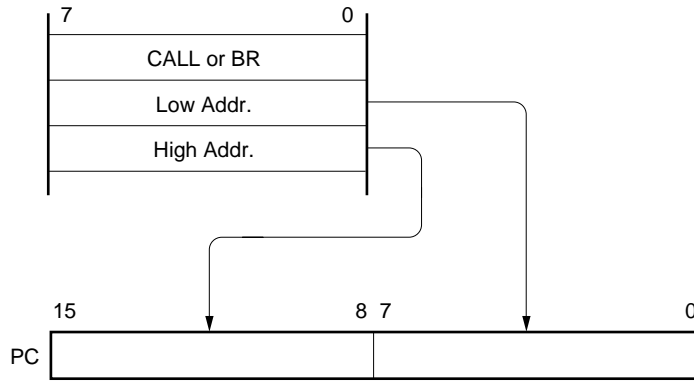
[Function]

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

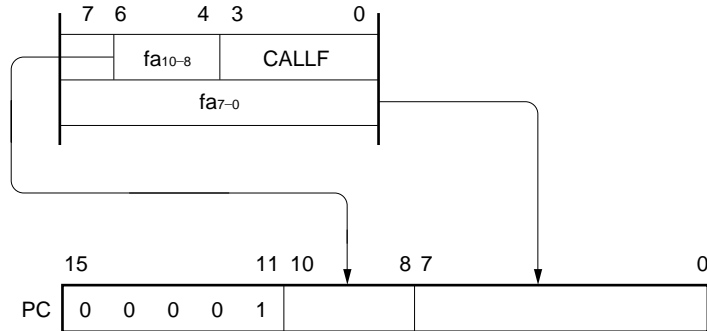
This function is carried out when the CALL !addr16 or BR !addr16 or CALLF !addr11 instruction is executed. The CALL !addr16 and BR !addr16 instructions can be used to branch to any location in the memory. The CALLF !addr11 instruction is used to branch to the area between 0800H through 0FFFH.

[Illustration]

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



In the case of CALLF !addr11 instruction



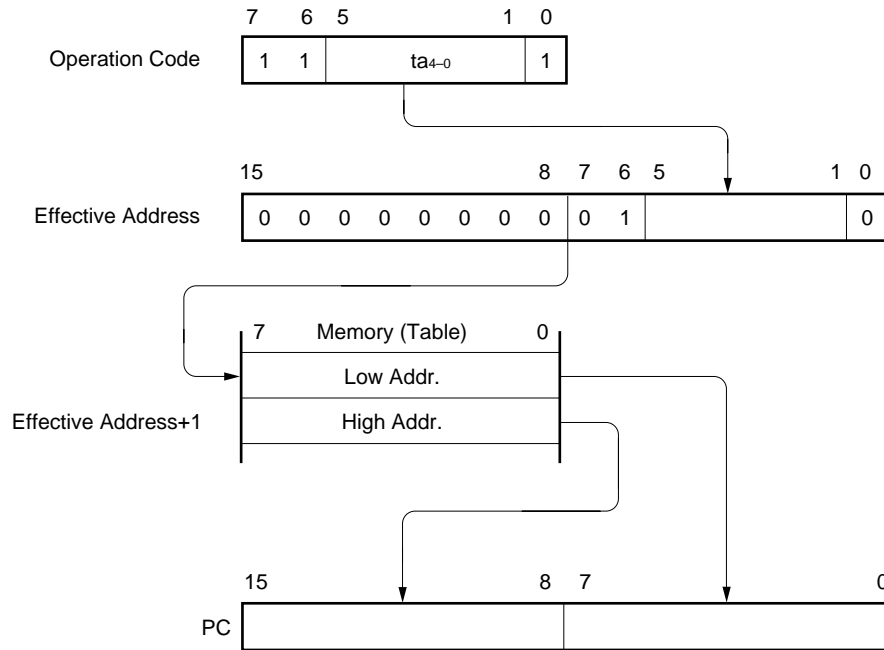
3.3.3 Table indirect addressing

[Function]

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

This addressing is used when the CALLT [addr5] instruction is executed. This instruction references an address stored in the memory table between 40H through 7FH, and can be used to branch to any location in the memory.

[Illustration]

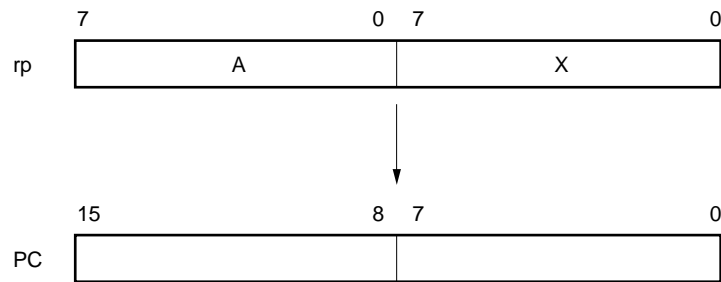


3.3.4 Register addressing

[Function]

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

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

[Illustration]

3.4 Operand Address Addressing

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

3.4.1 Implied addressing

[Function]

The register which functions as an accumulator (A and AX) in the general register is automatically addressed (implied). Of the μ PD178003 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.

[Example]

In the case of MULU X

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

3.4.2 Register addressing

[Function]

This addressing mode is used to access a general-purpose register as an operand. The register to be accessed is specified by the register bank select flags (RBS0 and RBS1) and the register specification code (Rn and RPN) in the operation code.

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

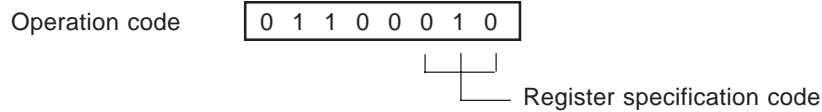
[Operand format]

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

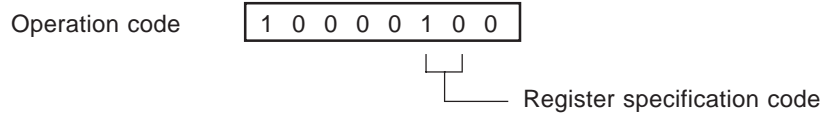
'r' and 'rp' can be written with function names (X, A, C, B, E, D, L, H, AX, BC, DE and HL) as well as absolute names (R0 to R7 and RP0 to RP3).

[Example]

MOV A, C; when selecting C register as r



INCW DE; when selecting DE register pair as rp



3.4.3 Direct addressing

[Function]

The memory with immediate data in an instruction word is directly addressed.

[Operand format]

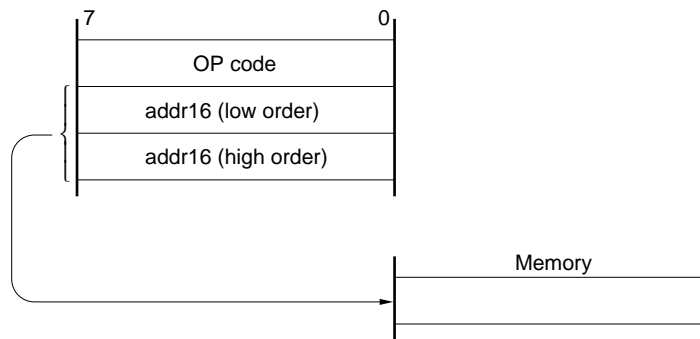
Symbol	Description
addr16	Label or 16-bit immediate data

[Example]

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

Operation code	1 0 0 0 1 1 1 0	Op code
	0 0 0 0 0 0 0 0	00H
	1 1 1 1 1 1 1 0	FEH

[Illustration]



3.4.4 Short direct addressing

[Function]

The memory to be manipulated in the fixed space is directly addressed with 8-bit data in an instruction word. This addressing is applied to the fixed 256-byte space FE20H to FF1FH. An internal RAM and a special function register (SFR) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

The SFR area (FF00H to FF1FH) where short direct addressing is applied is one part of all the SFR areas. In this area, ports which are frequently accessed in a program and a compare register of the timer/event counter is mapped and 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 **[Illustration]** on the next page.

[Operand format]

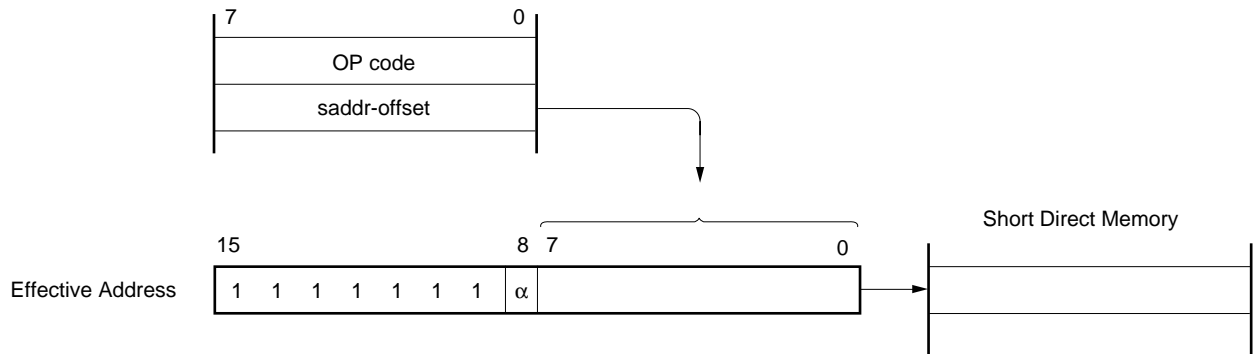
Symbol	Description
saddr	Label of FE20H to FF1FH immediate data
saddrp	Label of FE20H to FF1FH immediate data (even address only)

[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 register (SFR) is addressed with 8-bit immediate data in an instruction word.

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

[Operand format]

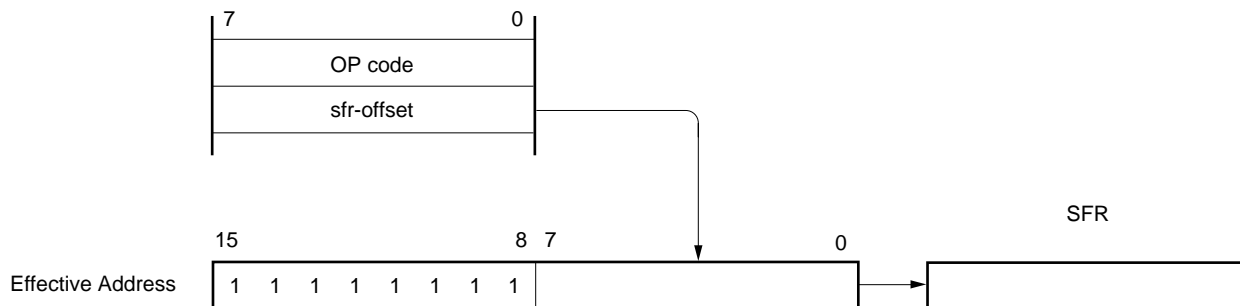
Symbol	Description
sfr	Special function register name
sfrp	16-bit manipulatable special function register name (even address only)

[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 mode is used to address the memory by using the contents of the register pair specified as the operand. The register pair to be accessed is specified by the register bank select flags (RBS0 and RBS1) and the register pair specification code in the operation code. This addressing mode can be used to access the entire memory space.

[Operand format]

Symbol	Description
—	[DE], [HL]

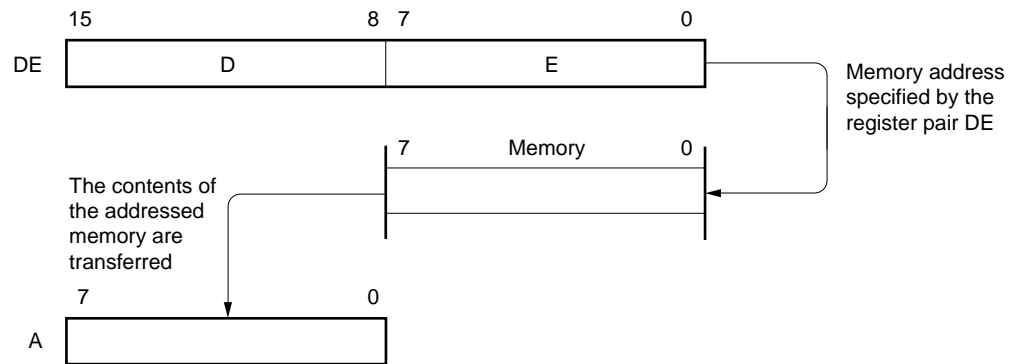
[Example]

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 mode is used to address a memory location specified by the result of adding the 8-bit immediate data to the contents of the HL register pair which is used as a base register. The HL register pair accessed is the register 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 all the memory spaces.

[Operand format]

Symbol	Description
—	[HL + byte]

[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 mode is used to address a memory location specified by the result of adding the contents of the B or C register specified in the instruction word to the contents of the HL register pair which is used as a base register. The HL, B, and C registers accessed are the registers 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 all the memory spaces.

[Operand format]

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

[Example]

In the case of 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/restored upon generation of an interrupt request.

Stack addressing enables to address the internal high-speed RAM area only.

[Example]

In the case of PUSH DE

Operation code

1	0	1	1	0	1	0	1
---	---	---	---	---	---	---	---

[MEMO]

CHAPTER 4 PORT FUNCTIONS

4.1 Port Functions

The μ PD178003 subseries units incorporate one input port, three output ports and 58 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. Port Types

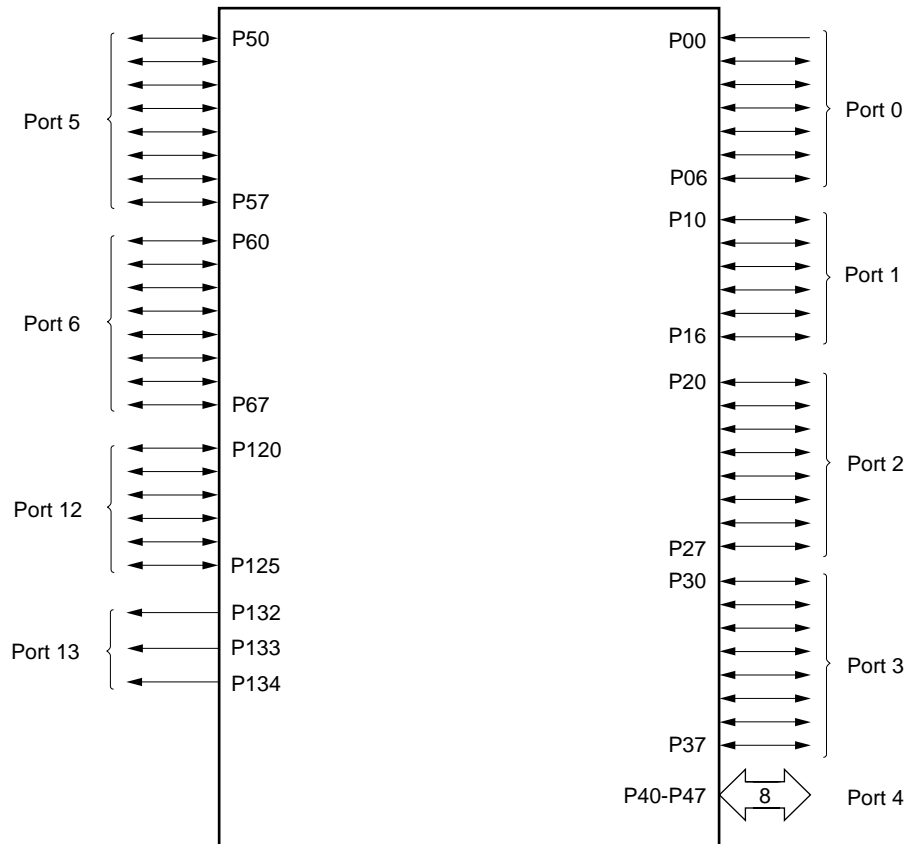


Table 4-1. Port Functions

Pin Name	I/O	Function		Alternate Function Pin
P00	Input	Port 0. 8-bit input/output port.	Input only	INTP0
P01	I/O		Input/output mode can be specified bit-wise.	INTP1
P02-P06				—
P10-P12	I/O	Port 1. 6-bit input/output port. Input/output mode can be specified bit-wise.	ANI0-ANI2	
P13-P15			—	
P20	I/O	Port 2. 8-bit input/output port. Input/output mode can be specified bit-wise.	SI1	
P21			SO1	
P22			$\overline{\text{SCK1}}$	
P23-P27			—	
P30-P32	I/O	Port 3. 8-bit input/output port. Input/output mode can be specified bit-wise.	—	
P33			TI1	
P34			TI2	
P35			—	
P36			BEEP	
P37			—	
P40-P47	I/O	Port 4. 8-bit input/output port. Input/output mode can be specified bit-wise. Test input flag (KRIF) is set to 1 by falling edge detection.	—	
P50-P57	I/O	Port 5. 8-bit input/output port. Input/output mode can be specified bit-wise.	—	
P60-P63	I/O	Port 6. 8-bit input/output port.	N-ch open drain input/output port. LEDs can be driven directly.	—
P64-P67		Input/output mode can be specified bit-wise.		
P120-P125	I/O	Port 12. 6-bit input/output port. Input/output mode can be specified bit-wise.	—	
P132-P134	Output	Port 13. 3-bit output port. N-ch open-drain output port.	—	

4.2 Port Configuration

A port consists of the following hardware:

Table 4-2. Port Configuration

Item	Configuration
Control register	Port mode register (PMm: m = 0 to 3, 5, 6, 12) Port mode register (MM) Note Key return mode register (KRM)
Port	Total: 62 ports (1 input, 3 outputs, 58 I/Os)

Note Port mode register 4 specifies port 4 input/output.

4.2.1 Port 0

Port 0 is a 7-bit input/output port with output latch. P01 to P06 pins can specify the input mode/output mode in 1-bit units with the port mode register 0 (PM0). P00 pin is input-only port.

Dual-functions include external interrupt request input.

Reset input sets port 0 to input mode.

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

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

Figure 4-2. P00 Block Diagram

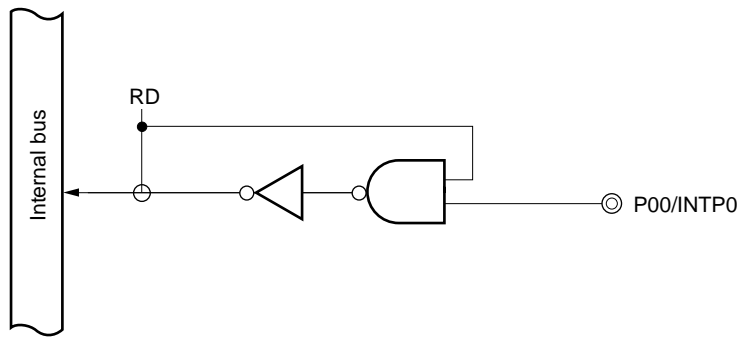
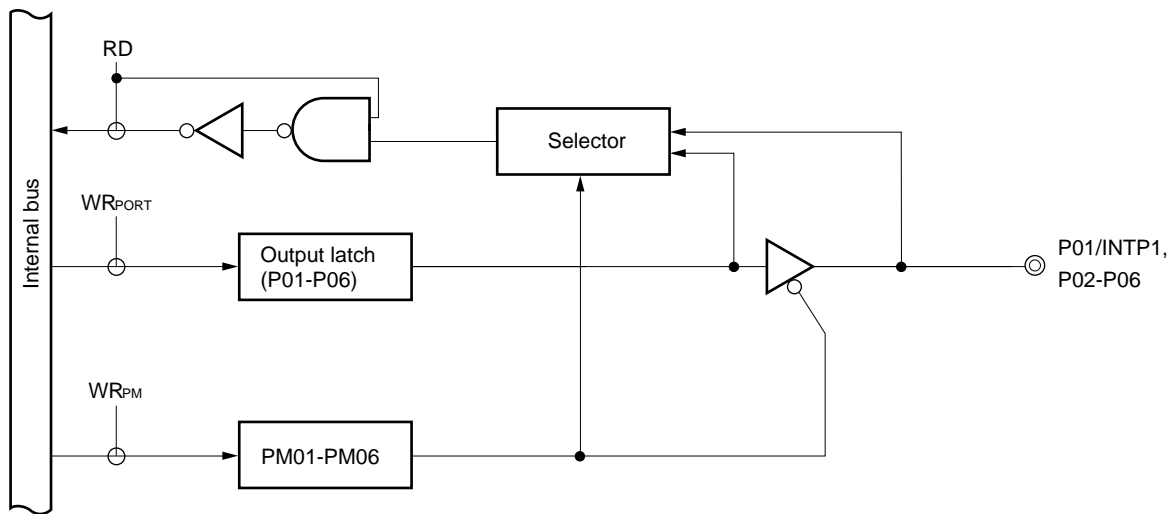


Figure 4-3. P01 to P06 Block Diagram



PM : Port mode register

RD : Port 0 read signal

WR : Port 0 write signal

4.2.2 Port 1

Port 1 is a 6-bit input/output port with output latch. It can specify the input mode/output mode in 1-bit units with a port mode register 1 (PM1).

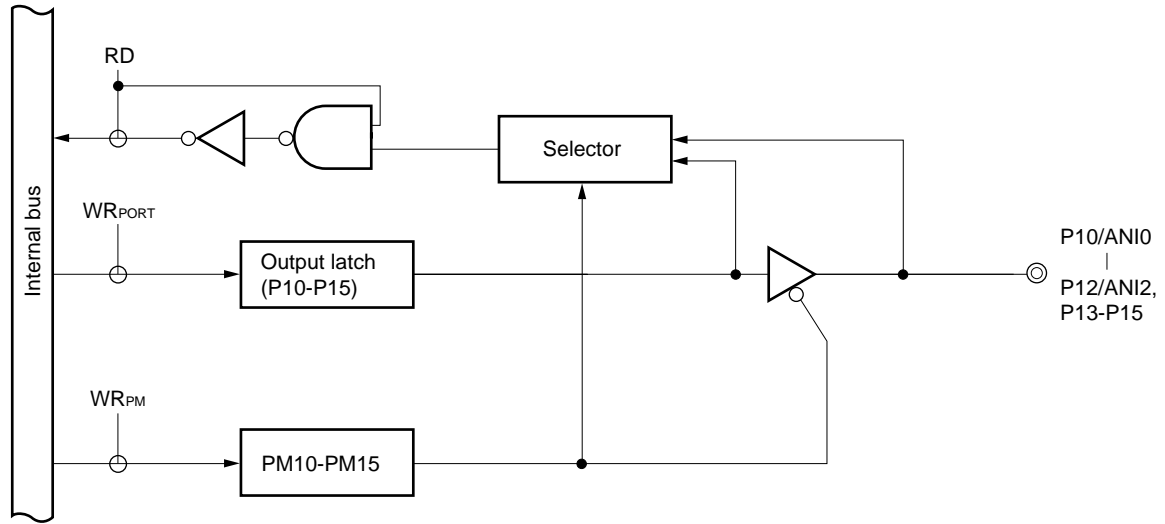
It becomes undefined when a pin specified by A/D converter input is read.

Dual-functions include an A/D converter analog input.

Reset input sets port 1 to input mode.

Figure 4-4 shows a block diagram of port 1.

Figure 4-4. P10 to P15 Block Diagram



PM : Port mode register

RD : Port 1 read signal

WR : Port 1 write signal

4.2.3 Port 2

Port 2 is an 8-bit input/output port with output latch. P20 to P27 pins can specify the input mode/output mode in 1-bit units with the port mode register 2 (PM2).

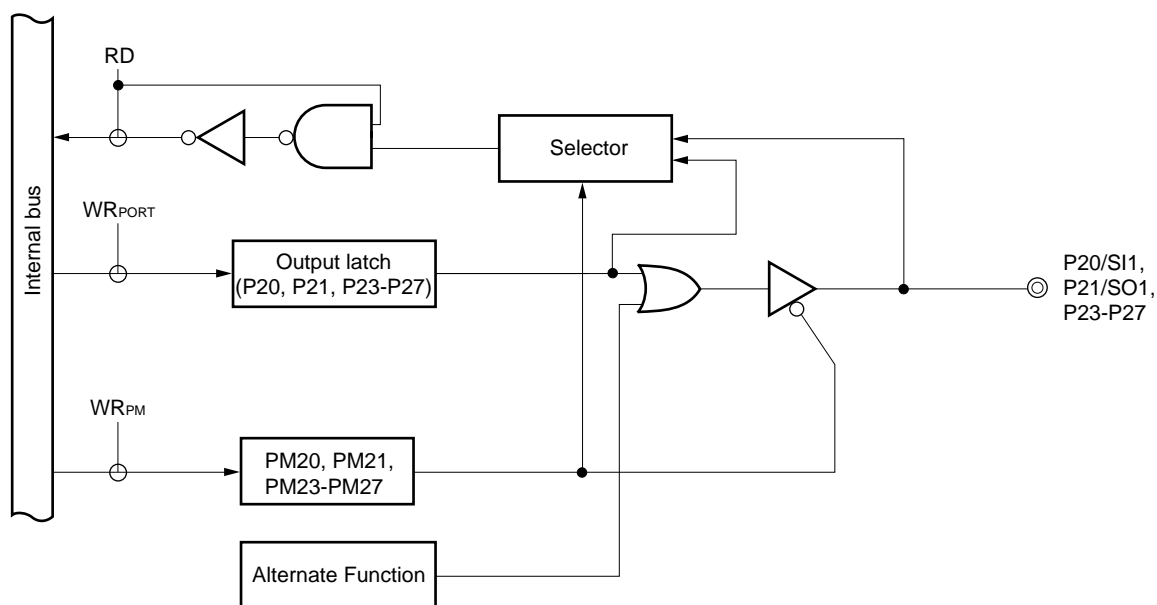
Dual-functions include serial interface data input/output and clock input/output.

Reset input sets port 2 to input mode.

Figures 4-5 and 4-6 show a block diagram of port 2.

Caution When P20 to P22 are used as serial interface pins, set the input/output and output latch according to its functions. For the setting method, refer to Figure 10-3 Serial Operating Mode Register 1 (CSIM1) Format.

Figure 4-5. P20, P21, P23 to P27 Block Diagram

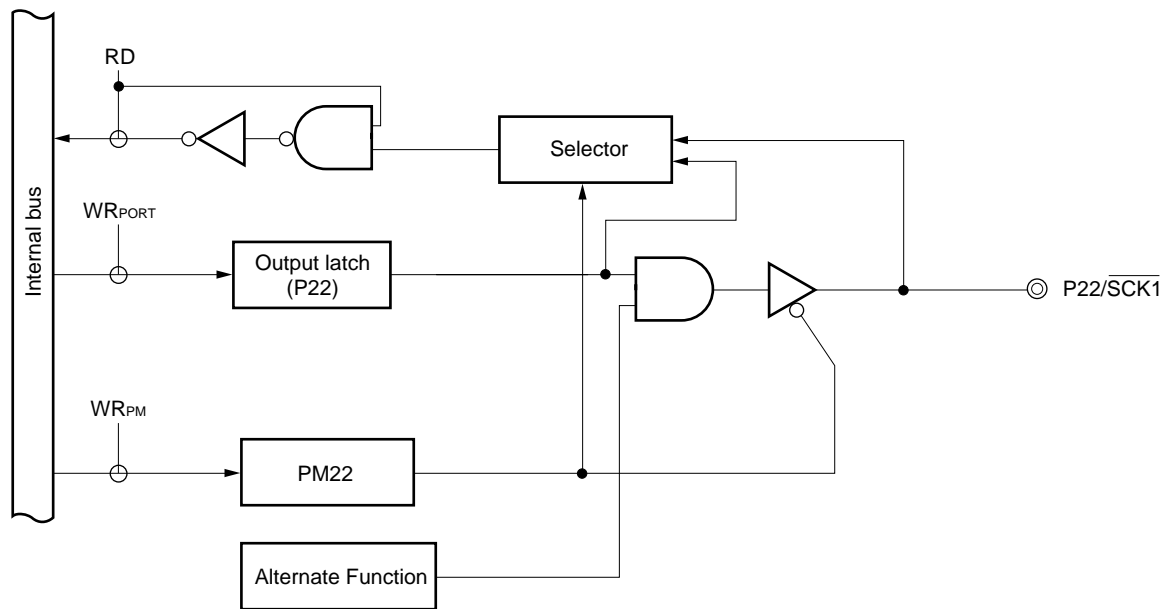


PM : Port mode register

RD : Port 2 read signal

WR : Port 2 write signal

Figure 4-6. P22 Block Diagram



PM : Port mode register

RD : Port 2 read signal

WR : Port 2 write signal

4.2.4 Port 3

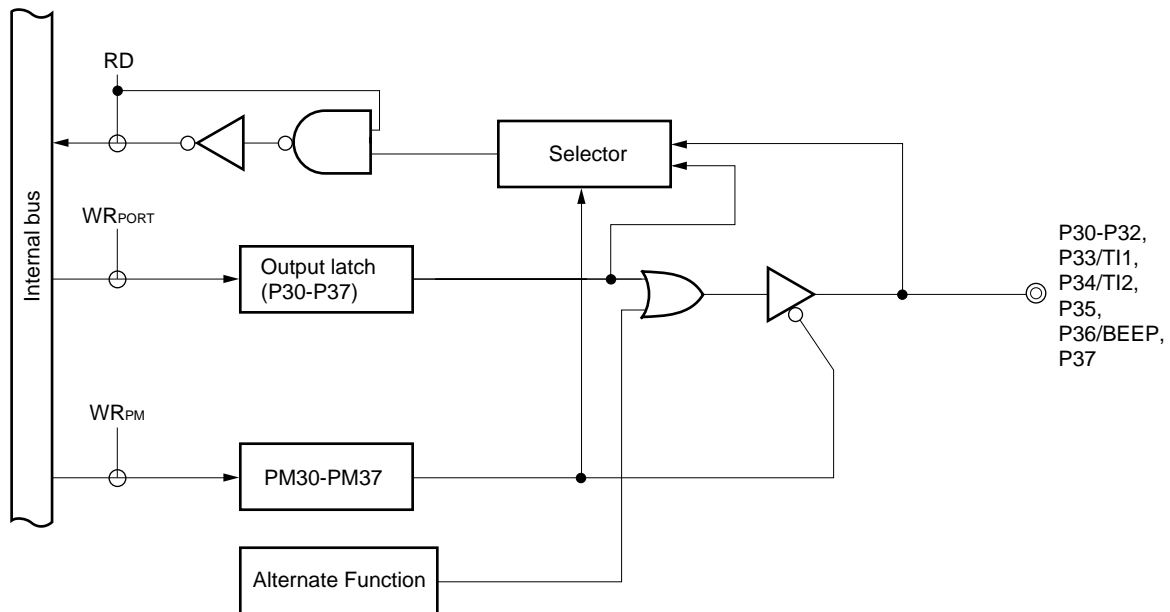
Port 3 is an 8-bit input/output port with output latch. P30 to P37 pins can specify the input mode/output mode in 1-bit units with the port mode register 3 (PM3).

Dual-functions include timer input and buzzer output.

Reset input sets port 3 to input mode.

Figure 4-7 shows a block diagram of port 3.

Figure 4-7. P30 to P37 Block Diagram



PM : Port mode register

RD : Port 3 read signal

WR : Port 3 write signal

4.2.5 Port 4

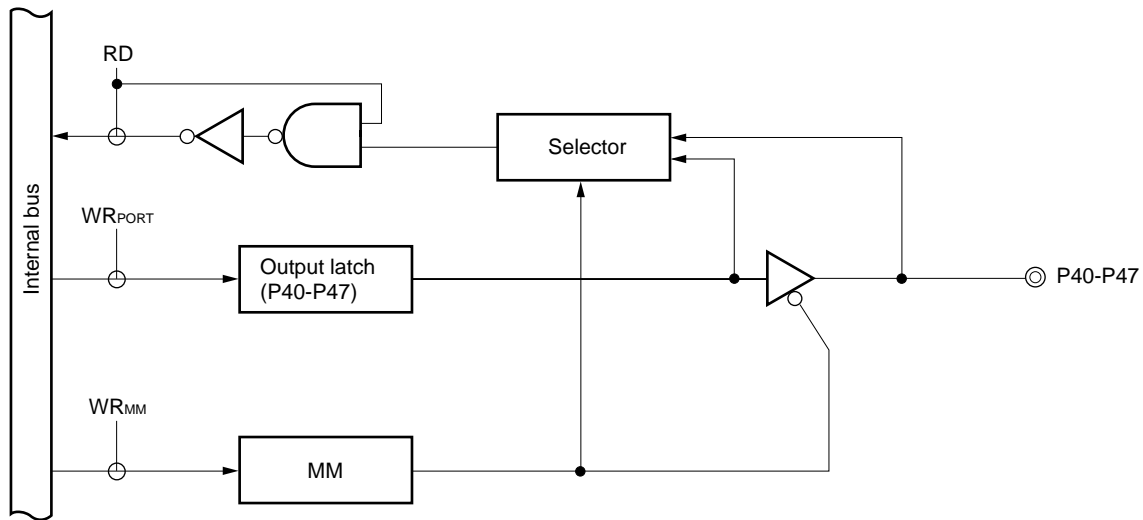
Port 4 is an 8-bit input/output port with output latch. P40 to P47 pins can specify the input mode/output mode in 8-bit units with the port mode register 4 (MM).

The test input flag (KRIF) can be set to 1 by detecting falling edges.

Reset input sets port 4 to input mode.

Figures 4-8 and 4-9 show a block diagram of port 4 and block diagram of falling edge detection circuit, respectively.

Figure 4-8. P40 to P47 Block Diagram

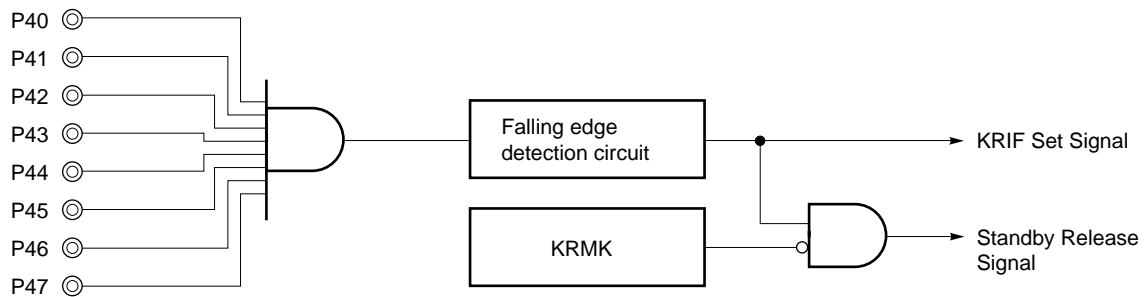


MM : Port mode register 4

RD : Port 4 read signal

WR : Port 4 write signal

Figure 4-9. Falling Edge Detection Circuit Block Diagram



KRIF : Test input flag

KRMK : Test mask flag

4.2.6 Port 5

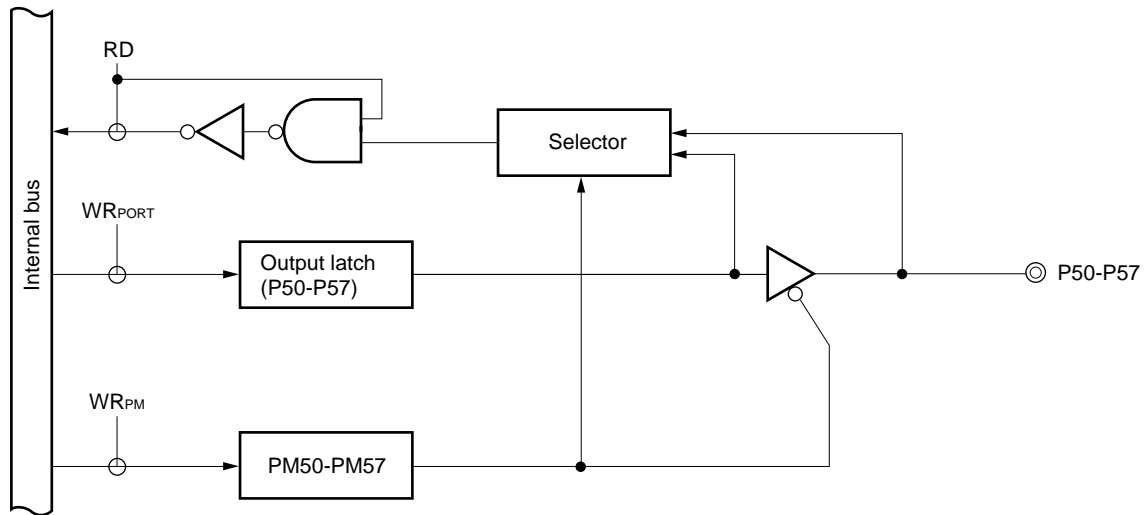
Port 5 is an 8-bit input/output port with output latch. P50 to P57 pins can specify the input mode/output mode in 1-bit units with the port mode register 5 (PM5).

Port 5 can drive LEDs directly.

Reset input sets port 5 to input mode.

Figure 4-10 shows a block diagram of port 5.

Figure 4-10. P50 to P57 Block Diagram



PM : Port mode register

RD : Port 5 read signal

WR : Port 5 write signal

4.2.7 Port 6

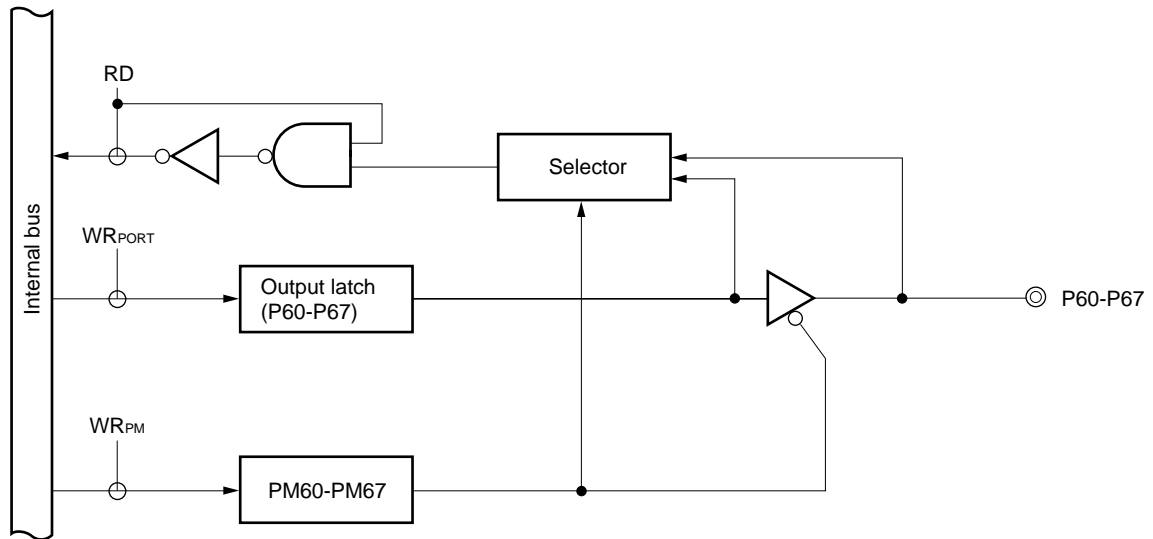
Port 6 is an 8-bit input/output port with output latch. P60 to P67 pins can specify the input mode/output mode in 1-bit units with the port mode register 6 (PM6).

Pins P60 to P63 can drive LEDs directly. This is an N-ch open-drain output port.

Reset input sets port 6 to input mode.

Figure 4-11 shows block diagram of port 6.

Figure 4-11. P60 to P67 Block Diagram



PM : Port mode register

RD : Port 6 read signal

WR : Port 6 write signal

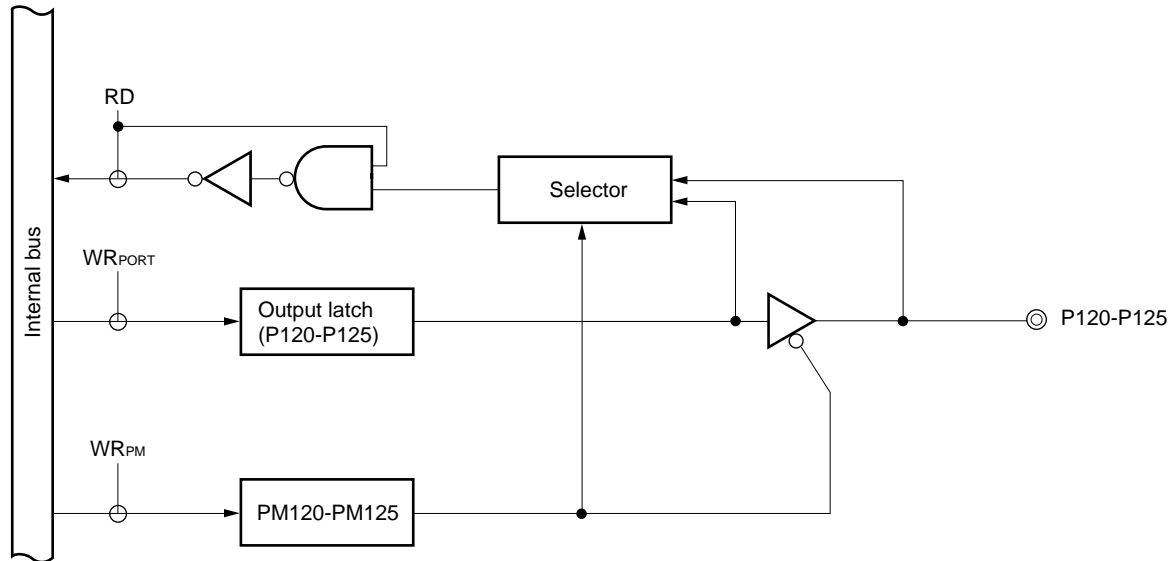
4.2.8 Port 12

This is a 6-bit input/output port with output latches. Input mode/output mode can be specified bit-wise by means of port mode register 12 (PM12).

Reset input sets the input mode.

The port 12 block diagram is shown in Figure 4-12.

Figure 4-12. P120 to P125 Block Diagram



PM : Port mode register

RD : Port 12 read signal

WR : Port 12 write signal

4.2.9 Port 13

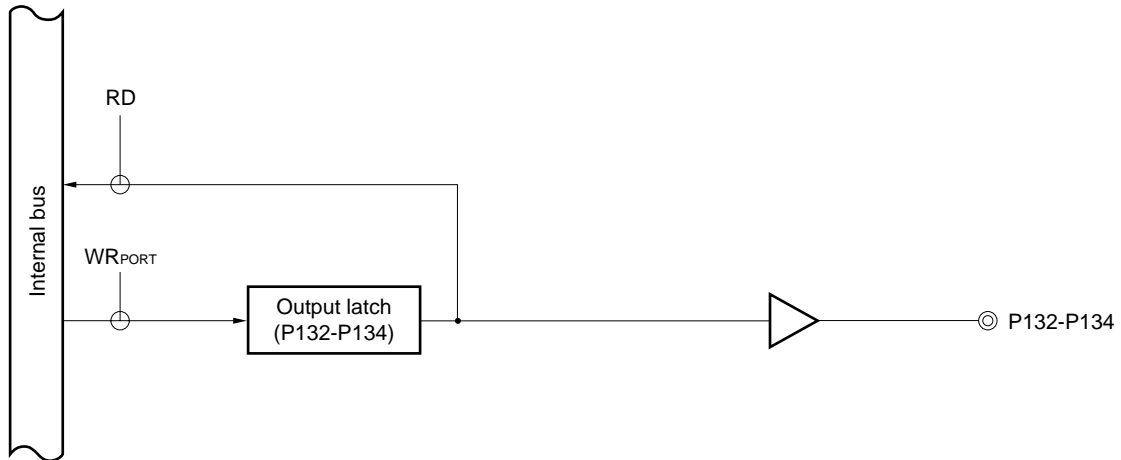
This is a 3-bit output port with an output latch.

This is an N-ch open-drain output port.

These pins are set in the general-purpose output port mode at reset.

The port 13 block diagram is shown in Figure 4-13.

Figure 4-13. P132 to P134 Block Diagram



RD : Port 13 read signal

WR : Port 13 write signal

4.3 Port Function Control Registers

The following three types of registers control the ports.

- Port mode registers (PM0 to PM3, PM5, PM6, PM12)
- Port mode register 4 (MM)
- Key return mode register (KRM)

(1) Port mode registers (PM0 to PM3, PM5, PM6, PM12)

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

PM0 to PM3, PM5, PM6, and PM12 are independently set with a 1-bit or 8-bit memory manipulation instruction. Reset input sets registers to FFH.

When port pins are used as the dual-function pins, set the port mode register and output latch according to Table 4-3.

Cautions 1. Pin P00 is input-only pin.

2. As port 0 has a dual function as external interrupt request 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.

3. The port mode register 4 (MM) specifies P40 to P47 as input/output pins.

Table 4-3. Port Mode Register and Output Latch Settings when Using Alternate Functions

Pin Name	Alternate Functions		PM _{xx}	P _{xx}
	Name	Input/Output		
P00	INTP0	Input	None	None
P01	INTP1	Input	1	×
P10-P12 ^{Note}	ANI0-ANI2	Input	1	×
P33	TI1	Input	1	×
P34	TI2	Input	1	×
P36	BEEP	Output	0	0

Note If these ports are read out when these pins are used in the alternate function mode, undefined values are read.

Caution When P20 to P22 of port 2 are used for serial interface, the I/O latch or output latch must be set according to its function. For the setting methods, refer to Figure 10-3 Serial Operating Mode Register 1 (CSIM1) Format.

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

Figure 4-14. Port Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM0	1	PM06	PM05	PM04	PM03	PM02	PM01	1	FF20H	FFH	R/W
PM1	1	1	PM15	PM14	PM13	PM12	PM11	PM10	FF21H	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	FF25H	FFH	R/W
PM6	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60	FF26H	FFH	R/W
PM12	1	1	PM125	PM124	PM123	PM122	PM121	PM120	FF2CH	FFH	R/W
PMmn	Pmn Pin Input/Output Mode Selection (m=0-3, 6, 12 : n=0-7)										
0	Output mode (output buffer ON)										
1	Input mode (output buffer OFF)										

(2) Port mode register 4 (MM)

This register is used to set input/output of port 4.

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

Reset input sets this register to 10H.

Figure 4-15. Port Mode Register 4 (MM) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MM	0	0	0 ^{Note}	1 ^{Note}	0	MM2	MM1	MM0	FFF8H	10H	R/W

MM2	MM1	MM0	I/O Mode Selection of P40-P47
0	0	0	Input mode
0	0	1	Output mode
Other than above			Setting prohibited

Note Be sure to set MM bit 4 to 1, and bit 5 to 0.

Caution Be sure to set MM2 and MM1 to 0.

(3) Key return mode register (KRM)

This register sets enabling/disabling of standby function release by a key return signal (falling edge detection of port 4).

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

Reset input sets KRM to 02H.

Figure 4-16. Key Return Mode Register (KRM) Format

Symbol	7	6	5	4	3	2	①	②	Address	After Reset	R/W
KRM	0	0	0	0	0	0	KRMK	KRIF	FFF6H	02H	R/W

KRIF	Key Return Signal Detection Flag
0	Not Detected
1	Detected (Falling edge detection of port 4)

KRMK	Standby Mode Control by Key Return Signal
0	Standby mode release enabled
1	Standby mode release disabled

Caution When falling edge detection of port4 is used, KRIF should be cleared to 0 (not cleared to 0 automatically).

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 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 except for 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 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. This system clock oscillator oscillates at frequencies of 4.5 MHz. Oscillation can be stopped by executing the STOP instruction or setting the processor clock control register (PCC).

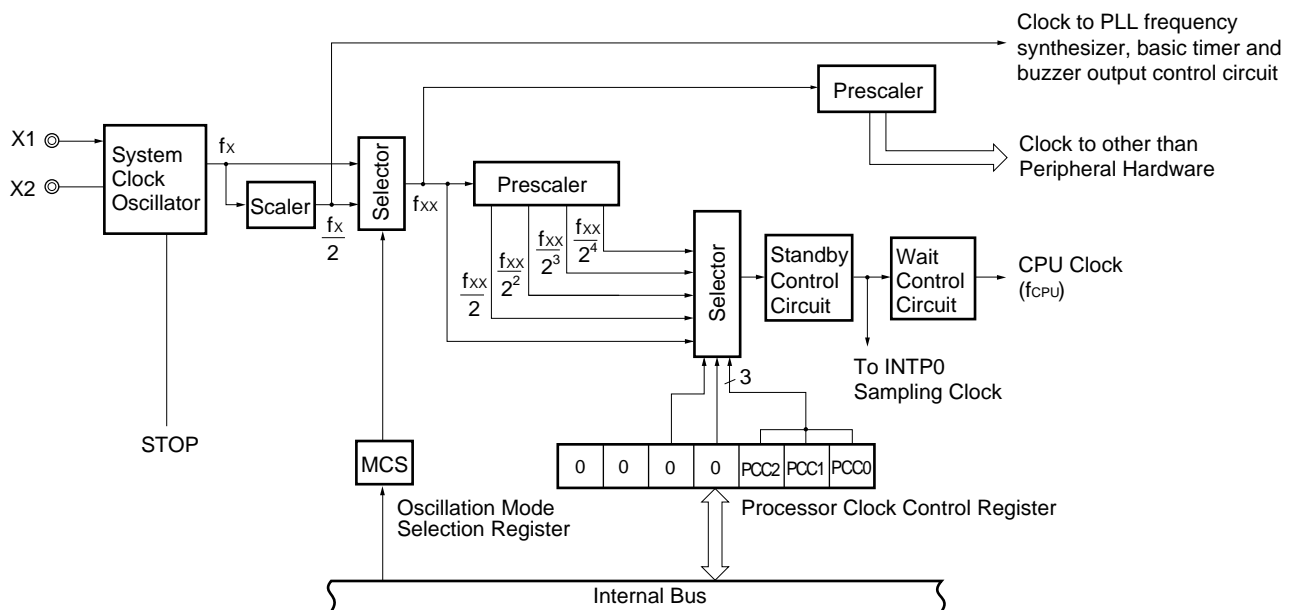
5.2 Clock Generator Configuration

The clock generator consists of the following hardware.

Table 5-1. Clock Generator Configuration

Item	Configuration
Control register	Processor clock control register (PCC) Oscillation mode selection register (OSMS)
Oscillator	System clock oscillator

Figure 5-1. Clock Generator Block Diagram



5.3 Clock Generator Control Register

The clock generator is controlled by the following two registers:

- Processor clock control register (PCC)
- Oscillation mode selection register (OSMS)

(1) Processor clock control register (PCC)

The PCC sets CPU clock.

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

Reset input sets the PCC to 04H.

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

Symbol	7	6	5	4	③	2	1	0	Address	After Reset	R/W
PCC	0	0	0	0	CMS	PCC2	PCC1	PCC0	FFFBH	04H	R/W ^{Note}

R/W	CMS	Be sure to set this bit to 0.
-----	-----	-------------------------------

R/W	PCC2	PCC1	PCC0	CPU Clock (f_{CPU}) Selection		
					MCS = 1	MCS = 0
	0	0	0	f_{xx}	f_x	$f_x/2$
	0	0	1	$f_{xx}/2$	$f_x/2$	$f_x/2^2$
	0	1	0	$f_{xx}/2^2$	$f_x/2^2$	$f_x/2^3$
	0	1	1	$f_{xx}/2^3$	$f_x/2^3$	$f_x/2^4$
	1	0	0	$f_{xx}/2^4$	$f_x/2^4$	$f_x/2^5$
	Other than above			Setting prohibited		

Note Bits 4 to 7 are Read Only.

Caution Be sure to set CMS (bit 3) to 0.

- Remarks**
1. f_{xx} : System clock frequency (f_x or $f_x/2$)
 2. f_x : System clock oscillator frequency
 3. MCS : Bit 0 of oscillation mode selection register (OSMS)

The fastest instruction of the μ PD178003 subseries is executed within CPU clocks. Therefore, the relation between the CPU clock (f_{CPU}) and minimum instruction execution time is as shown in Table 5-2.

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

CPU Clock (f_{CPU})	Minimum Instruction Execution Time: $2/f_{\text{CPU}}$
f_x	$0.44 \mu\text{s}$
$f_x/2$	$0.89 \mu\text{s}$
$f_x/2^2$	$1.78 \mu\text{s}$
$f_x/2^3$	$3.56 \mu\text{s}$
$f_x/2^4$	$7.11 \mu\text{s}$
$f_x/2^5$	$14.22 \mu\text{s}$

$f_x = 4.5 \text{ MHz}$

f_x : System clock oscillation frequency

(2) Oscillation mode selection register (OSMS)

This register specifies whether the clock output from the system clock oscillator without passing through the scaler is used as the system clock, or the clock output via the scaler is used as the main system clock.

OSMS is set with 8-bit memory manipulation instruction.

Reset input sets MCS flag to 0, but when MCS flag is read, it becomes undefined.

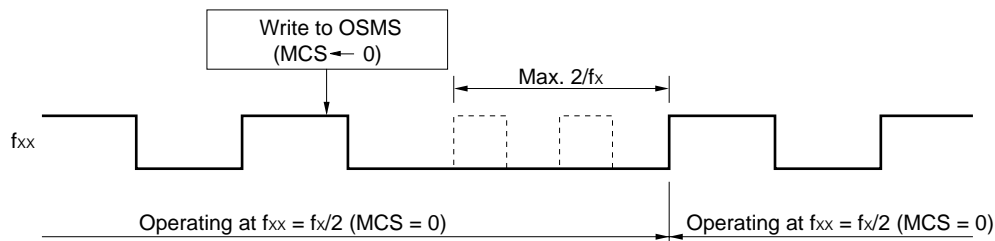
Figure 5-3. Oscillation Mode Selection Register (OSMS) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
OSMS	Un-defined	Un-defined	Un-defined	Un-defined	Un-defined	Un-defined	Un-defined	MCS	FFF2H	00H	W

MCS	System Clock Scaler Control
0	Scaler used
1	Scaler not used

- Caution**
1. As shown in Figure 5-4 below, writing data (including same data as previous) to OSMS cause delay of system clock cycle up to $2/f_x$ during the write operation. Therefore, if this register is written during the operation, in peripheral hardware which operates with the system clock, a temporary error occurs in the count clock cycle of timer, etc. In addition, because the oscillation mode is switched by this register, the clocks for peripheral hardware as well as that for the CPU are switched. However, the PLL synthesizer, basic timer and the clock supplied for buzzer output control circuit are not affected.
 2. When writing "1" to MCS, V_{DD} must be 4.5 V or higher before the write execution.

Figure 5-4. System Clock Waveform due to Writing to OSMS



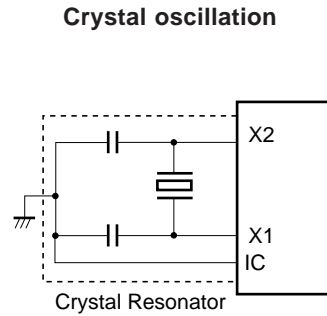
Remark f_{xx} : System clock frequency (f_x or $f_x/2$)
 f_x : System clock oscillation frequency

5.4 System Clock Oscillator

5.4.1 System clock oscillator

The system clock oscillator oscillates with a crystal resonator (4.5 MHz TYP.) connected to the X1 and X2 pins. Figure 5-5 shows an external circuit of the system clock oscillator.

Figure 5-5. External Circuit of System Clock Oscillator



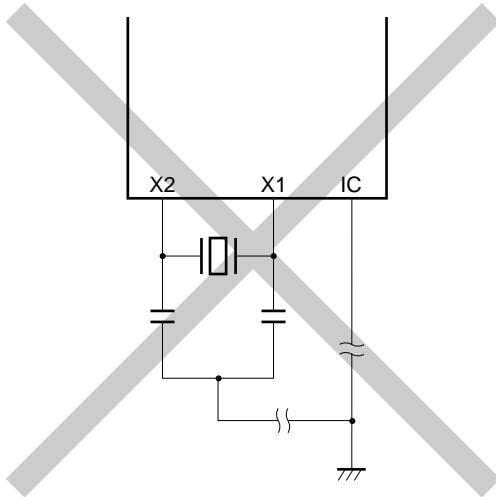
Caution When using a system clock oscillator, carry out wiring in the broken line area in Figure 5-5 to prevent any effects from wiring capacities.

- Minimize the wiring length.
- Do not allow wiring to intersect with other signal conductors. Do not allow wiring to come near changing high current.
- Set the potential of the grounding position of the oscillator capacitor to that of GND. Do not ground to any ground pattern where high current is present.
- Do not fetch signals from the oscillator.

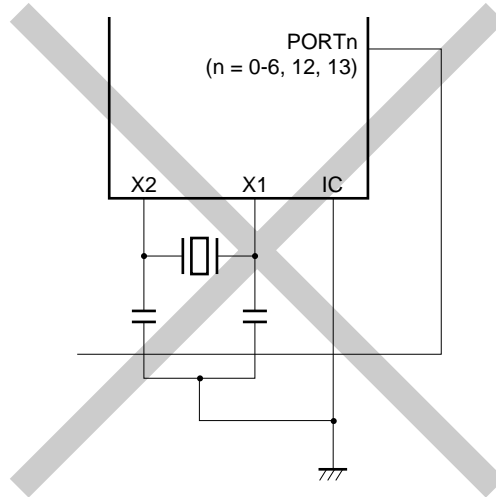
Figure 5-6 shows examples of resonator having bad connection.

Figure 5-6. Examples of Resonator with Bad Connection (1/2)

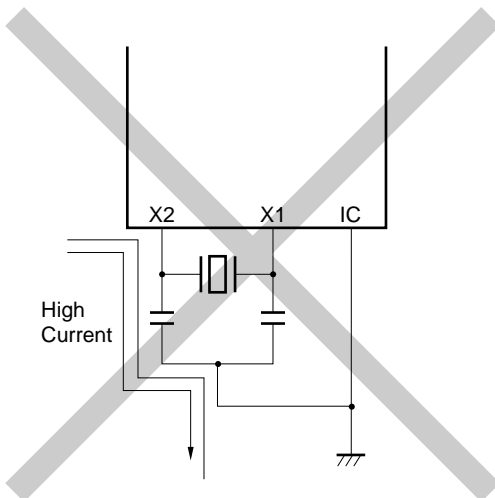
(a) Wiring of connection circuits is too long



(b) Signal conductors intersect each other



(c) Changing high current is too near a signal conductor



(d) Current flows through the grounding line of the oscillator (potential at points A, B, and C fluctuate)

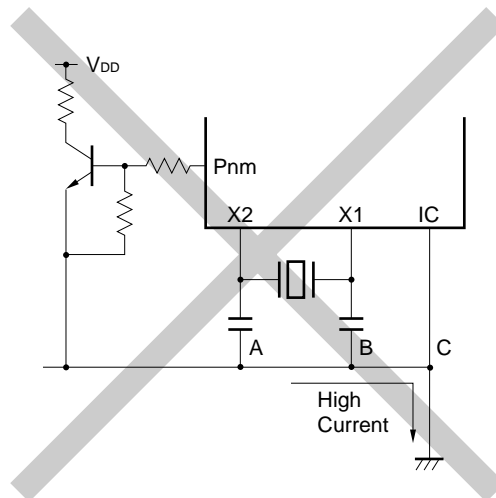
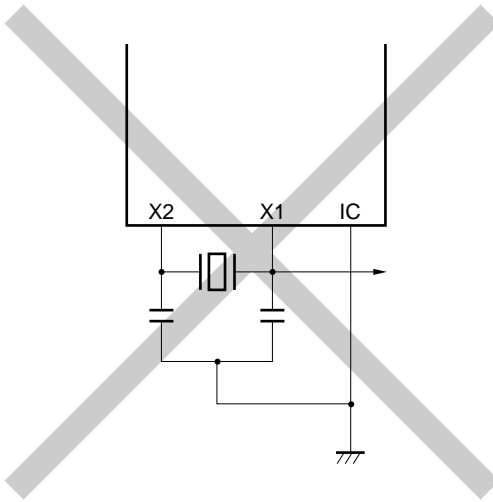


Figure 5-6. Examples of Resonator with Bad Connection (2/2)

(e) Signals are fetched



5.4.2 Scaler

The scaler divides the system clock oscillator output (f_{xx}) and generates various clocks.

5.5 Clock Generator Operations

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

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

The following clock generator functions and operations are determined with the processor clock control register (PCC) and the oscillation mode selection register (OSMS).

- Upon generation of reset signal, the lowest speed mode of the system clock ($14.22 \mu s$ when operated at 4.5 MHz) is selected. System clock oscillation stops while low level is applied to \overline{RESET} pin.
- One of the six CPU clock types (0.44, 0.89, 1.78, 3.56, 7.11, $14.22 \mu s$ at 4.5 MHz) can be selected by setting the PCC and OSMS.
- Two standby modes, the STOP and HALT modes, are available.
- The system clock is divided and supplied to the peripheral hardware. The peripheral hardware also stops if the system clock is stopped.

5.6 Changing System Clock and CPU Clock Settings

5.6.1 Time required for switchover between system clock and CPU clock

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

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

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

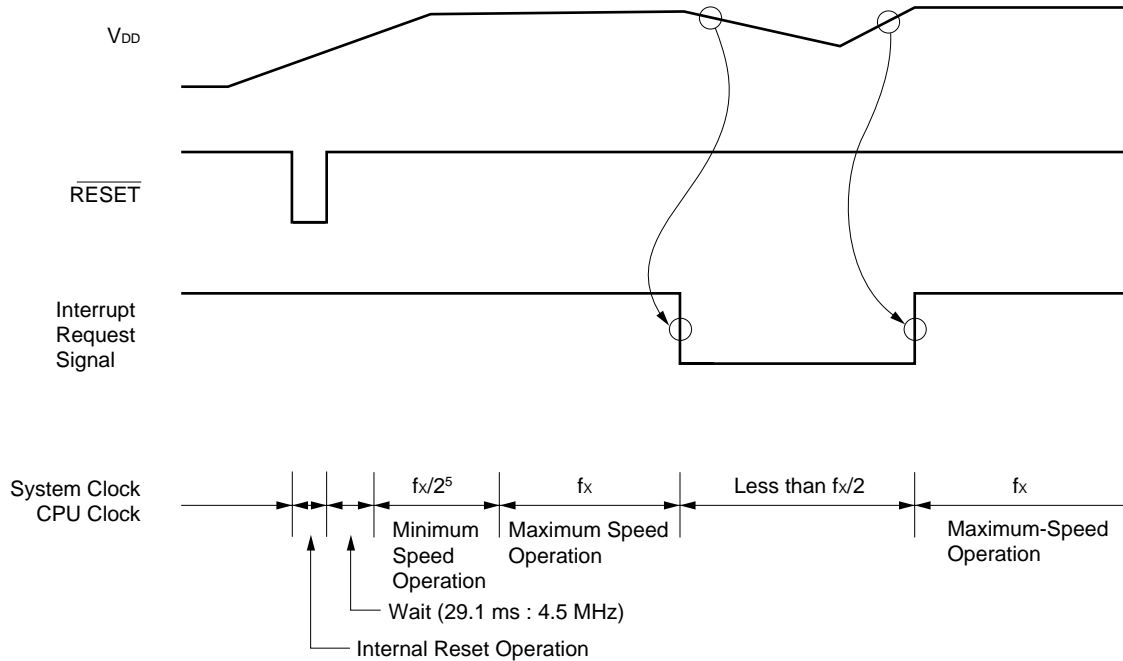
Set Values before Switchover			Set Values After Switchover														
PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0
			0	0	0	0	0	1	0	1	0	0	1	1	1	0	0
0	0	0	/			16 instructions			16 instructions			16 instructions			16 instructions		
0	0	1				8 instructions			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 System clock and CPU clock switching procedure

This section describes switching procedure between system clock and CPU clock.

Figure 5-7. System Clock and CPU Clock Switching



- (1) The CPU is reset by setting the $\overline{\text{RESET}}$ signal to low level, or V_{DD} becomes less than power-on clear voltage after power-on. After that, when reset is released by setting the $\overline{\text{RESET}}$ signal to high level, or V_{DD} become higher than power-on clear voltage, system clock starts oscillation. At this time, 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 ($14.22 \mu\text{s}$ when operated at 4.5 MHz).
- (2) After the lapse of a sufficient time for the V_{DD} voltage to increase to enable operation at maximum speeds, the processor clock control register (PCC) and oscillation mode select register (OSMS) are rewritten and the maximum-speed operation is carried out.
- (3) Upon detection of a decrease of the V_{DD} voltage due to an interrupt request, the PCC and OSMS are rewritten.
- (4) Upon detection of V_{DD} voltage restored due to an interrupt request, the PCC and OSMS are rewritten and the maximum-speed operation is resumed.

CHAPTER 6 8-BIT TIMER/EVENT COUNTERS 1 AND 2

6.1 8-Bit Timer/Event Counters 1 and 2 Functions

For the 8-bit timer/event counters 1 and 2, two modes are available. One is a mode for two-channel 8-bit timer/event counters to be used separately (the 8-bit timer/event counter mode) and the other is a mode for the 8-bit timer/event counter to be used as 16-bit timer/event counter (the 16-bit timer/event counter mode).

6.1.1 8-bit timer/event counter mode

The 8-bit timer/event counters 1 and 2 have the following functions.

- Interval timer
- External event counter

(1) 8-bit interval timer

Interrupt requests are generated at the preset time intervals.

Table 6-1. 8-Bit Timer/Event Counters 1 and 2 Interval Times

Minimum Interval Time		Maximum Interval Time		Resolution	
MCS = 1	MCS = 0	MCS = 1	MCS = 0	MCS = 1	MCS = 0
$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)	$2^9 \times 1/f_x$ (113.8 μ s)	$2^{10} \times 1/f_x$ (227.5 μ s)	$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)
$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μ s)	$2^{10} \times 1/f_x$ (227.5 μ s)	$2^{11} \times 1/f_x$ (455.1 μ s)	$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μ s)
$2^3 \times 1/f_x$ (1.78 μ s)	$2^4 \times 1/f_x$ (3.56 μ s)	$2^{11} \times 1/f_x$ (455.1 μ s)	$2^{12} \times 1/f_x$ (910.2 μ s)	$2^3 \times 1/f_x$ (1.78 μ s)	$2^4 \times 1/f_x$ (3.56 μ s)
$2^4 \times 1/f_x$ (3.56 μ s)	$2^5 \times 1/f_x$ (7.11 μ s)	$2^{12} \times 1/f_x$ (910.2 μ s)	$2^{13} \times 1/f_x$ (1.82 ms)	$2^4 \times 1/f_x$ (3.56 μ s)	$2^5 \times 1/f_x$ (7.11 μ s)
$2^5 \times 1/f_x$ (7.11 μ s)	$2^6 \times 1/f_x$ (14.2 μ s)	$2^{13} \times 1/f_x$ (1.82 ms)	$2^{14} \times 1/f_x$ (3.64 ms)	$2^5 \times 1/f_x$ (7.11 μ s)	$2^6 \times 1/f_x$ (14.2 μ s)
$2^6 \times 1/f_x$ (14.2 μ s)	$2^7 \times 1/f_x$ (28.4 μ s)	$2^{14} \times 1/f_x$ (3.64 ms)	$2^{15} \times 1/f_x$ (7.28 ms)	$2^6 \times 1/f_x$ (14.2 μ s)	$2^7 \times 1/f_x$ (28.4 μ s)
$2^7 \times 1/f_x$ (28.4 μ s)	$2^8 \times 1/f_x$ (56.9 μ s)	$2^{15} \times 1/f_x$ (7.28 ms)	$2^{16} \times 1/f_x$ (14.6 ms)	$2^7 \times 1/f_x$ (28.4 μ s)	$2^8 \times 1/f_x$ (56.9 μ s)
$2^8 \times 1/f_x$ (56.9 μ s)	$2^9 \times 1/f_x$ (113.8 μ s)	$2^{16} \times 1/f_x$ (14.6 ms)	$2^{17} \times 1/f_x$ (29.1 ms)	$2^8 \times 1/f_x$ (56.9 μ s)	$2^9 \times 1/f_x$ (113.8 μ s)
$2^9 \times 1/f_x$ (113.8 μ s)	$2^{10} \times 1/f_x$ (227.5 μ s)	$2^{17} \times 1/f_x$ (29.1 ms)	$2^{18} \times 1/f_x$ (58.3 ms)	$2^9 \times 1/f_x$ (113.8 μ s)	$2^{10} \times 1/f_x$ (227.5 μ s)
$2^{11} \times 1/f_x$ (455.1 μ s)	$2^{12} \times 1/f_x$ (910.2 μ s)	$2^{19} \times 1/f_x$ (116.5 ms)	$2^{20} \times 1/f_x$ (233.0 ms)	$2^{11} \times 1/f_x$ (455.1 μ s)	$2^{12} \times 1/f_x$ (910.2 μ s)

- Remarks**
1. f_x : System clock oscillation frequency
 2. MCS : Oscillation mode selection register (OSMS) bit 0
 3. Values in parentheses when operated at $f_x = 4.5$ MHz.

(2) External event counter

The number of pulses of an externally input signal can be measured.

6.1.2 16-bit timer/event counter mode

(1) 16-bit interval timer

Interrupt requests can be generated at the preset time intervals.

Table 6-2. Interval Times when 8-Bit Timer/Event Counters 1 and 2 are Used as 16-Bit Timer/Event Counters

Minimum Interval Time		Maximum Interval Time		Resolution	
MCS = 1	MCS = 0	MCS = 1	MCS = 0	MCS = 1	MCS = 0
$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)	$2^{17} \times 1/f_x$ (29.1 ms)	$2^{18} \times 1/f_x$ (58.3 ms)	$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)
$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μ s)	$2^{18} \times 1/f_x$ (58.3 ms)	$2^{19} \times 1/f_x$ (116.5 ms)	$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μ s)
$2^3 \times 1/f_x$ (1.78 μ s)	$2^4 \times 1/f_x$ (3.56 μ s)	$2^{19} \times 1/f_x$ (116.5 ms)	$2^{20} \times 1/f_x$ (233.0 ms)	$2^3 \times 1/f_x$ (1.78 μ s)	$2^4 \times 1/f_x$ (3.56 μ s)
$2^4 \times 1/f_x$ (3.56 μ s)	$2^5 \times 1/f_x$ (7.11 μ s)	$2^{20} \times 1/f_x$ (233.0 ms)	$2^{21} \times 1/f_x$ (466.0 ms)	$2^4 \times 1/f_x$ (3.56 μ s)	$2^5 \times 1/f_x$ (7.11 μ s)
$2^5 \times 1/f_x$ (7.11 μ s)	$2^6 \times 1/f_x$ (14.2 μ s)	$2^{21} \times 1/f_x$ (466.0 ms)	$2^{22} \times 1/f_x$ (932.0 ms)	$2^5 \times 1/f_x$ (7.11 μ s)	$2^6 \times 1/f_x$ (14.2 μ s)
$2^6 \times 1/f_x$ (14.2 μ s)	$2^7 \times 1/f_x$ (28.4 μ s)	$2^{22} \times 1/f_x$ (932.0 ms)	$2^{23} \times 1/f_x$ (1.9 s)	$2^6 \times 1/f_x$ (14.2 μ s)	$2^7 \times 1/f_x$ (28.4 μ s)
$2^7 \times 1/f_x$ (28.4 μ s)	$2^8 \times 1/f_x$ (56.9 μ s)	$2^{23} \times 1/f_x$ (1.9 s)	$2^{24} \times 1/f_x$ (3.7 s)	$2^7 \times 1/f_x$ (28.4 μ s)	$2^8 \times 1/f_x$ (56.9 μ s)
$2^8 \times 1/f_x$ (56.9 μ s)	$2^9 \times 1/f_x$ (113.8 μ s)	$2^{24} \times 1/f_x$ (3.7 s)	$2^{25} \times 1/f_x$ (7.5 s)	$2^8 \times 1/f_x$ (56.9 μ s)	$2^9 \times 1/f_x$ (113.8 μ s)
$2^9 \times 1/f_x$ (113.8 μ s)	$2^{10} \times 1/f_x$ (227.5 μ s)	$2^{25} \times 1/f_x$ (7.5 s)	$2^{26} \times 1/f_x$ (14.9 s)	$2^9 \times 1/f_x$ (113.8 μ s)	$2^{10} \times 1/f_x$ (227.5 μ s)
$2^{11} \times 1/f_x$ (455.1 μ s)	$2^{12} \times 1/f_x$ (910.2 μ s)	$2^{27} \times 1/f_x$ (29.8 s)	$2^{28} \times 1/f_x$ (59.7 s)	$2^{11} \times 1/f_x$ (455.1 μ s)	$2^{12} \times 1/f_x$ (910.2 μ s)

Remarks 1. f_x : System clock oscillation frequency

2. MCS : Oscillation mode selection register (OSMS) bit 0

3. Values in parentheses when operated at $f_x = 4.5$ MHz.

(2) External event counter

The number of pulses of an externally input signal can be measured.

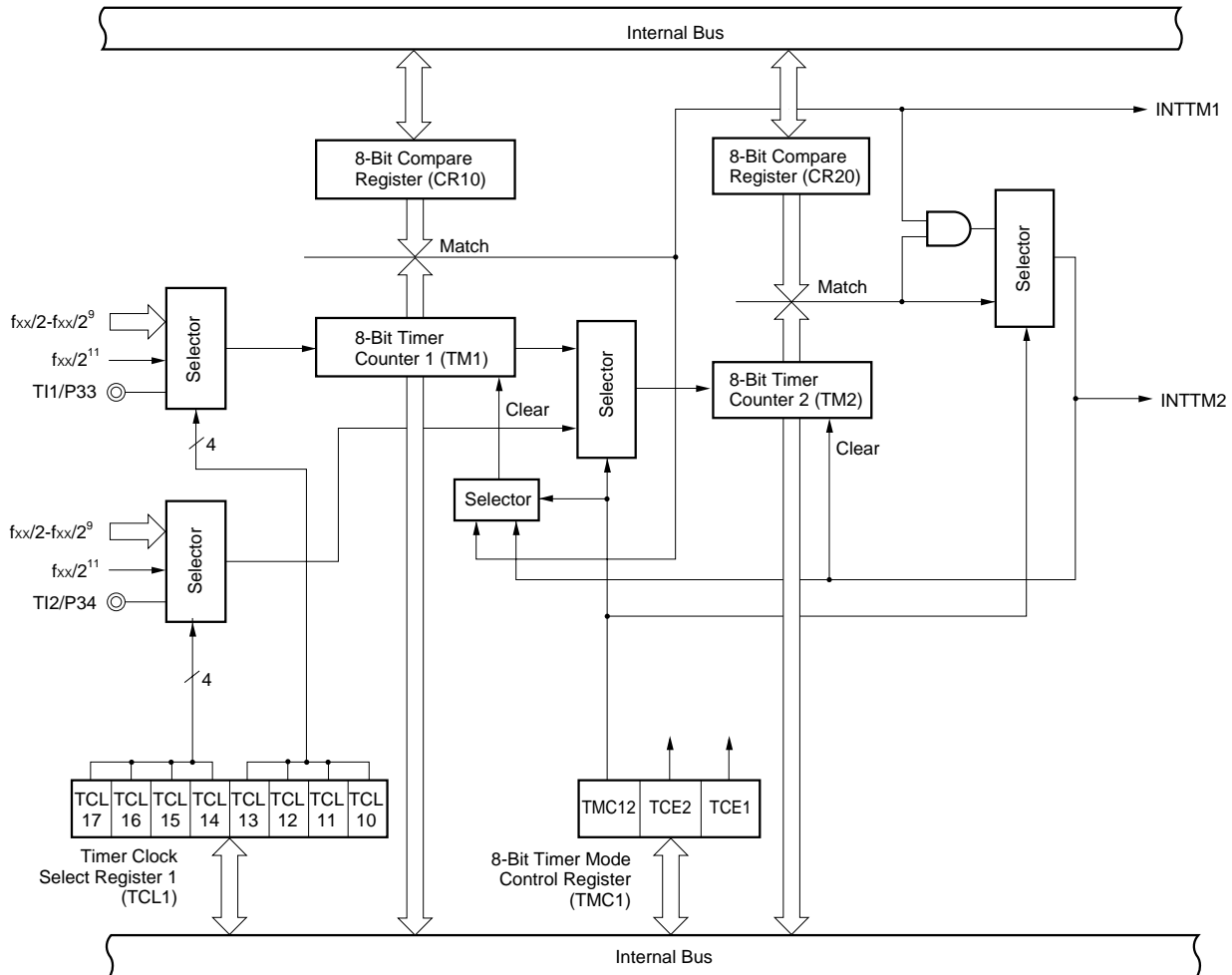
6.2 8-Bit Timer/Event Counters 1 and 2 Configurations

The 8-bit timer/event counters 1 and 2 consist of the following hardware.

Table 6-3. 8-Bit Timer/Event Counters 1 and 2 Configurations

Item	Configuration
Timer counter	8 bits × 2 (TM1, TM2)
Register	Compare register: 8 bits × 2 (CR10, CR20)
Control register	Timer clock select register 1 (TCL1) 8-bit timer mode control register 1 (TMC1)

Figure 6-1. 8-Bit Timer/Event Counters 1 and 2 Block Diagram



(1) Compare registers 10 and 20 (CR10, CR20)

These are 8-bit registers to compare the value set to CR10 to the 8-bit timer counter 1 (TM1) count value, and the value set to CR20 to the 8-bit timer counter 2 (TM2) count value, and, if they match, generate an interrupt request (INTTM1 and INTTM2, respectively).

CR10 and CR20 are set with an 8-bit memory manipulation instruction. They cannot be set with a 16-bit memory manipulation instruction. When the compare register is used as 8-bit timer/event counter, the 00H to FFH values can be set. When the compare register is used as 16-bit timer/event counter, the 0000H to FFFFH values can be set.

Reset input makes CR10 and CR20 undefined.

Cautions 1. When using the compare register as 16-bit timer/event counter, be sure to set data after stopping timer operation.

2. If the new values of CR10 and CR20 are less than the value of the 8-bit timer counters (TM1, TM2), TM1 and TM2 continue counting, overflows and count again from 0. If the new values of CR10 and CR20 are less than the previous values, it is necessary to restart the timer.

(2) 8-bit timer counters 1, 2 (TM1, TM2)

These are 8-bit registers to count count pulses.

When TM1 and TM2 are used in the 8-bit timer \times 2 channels mode, they are read with an 8-bit memory manipulation instruction. When TM1 and TM2 are used as 16-bit timer \times 1 channels mode, 16-bit timer (TMS) is read with a 16-bit memory manipulation instruction.

Reset input sets TM1 and TM2 to 00H.

6.3 8-Bit Timer/Event Counters 1 and 2 Control Registers

The following two types of registers are used to control the 8-bit timer/event counter.

- Timer clock select register 1 (TCL1)
- 8-bit timer mode control register 1 (TMC1)

(1) Timer clock select register 1 (TCL1)

This register sets count clocks of 8-bit timer counters 1 and 2.

TCL1 is set with an 8-bit memory manipulation instruction.

Reset input sets TCL1 to 00H.

Figure 6-2. Timer Clock Select Register 1 (TCL1) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCL1	TCL17	TCL16	TCL15	TCL14	TCL13	TCL12	TCL11	TCL10	FF41H	00H	R/W

TCL13	TCL12	TCL11	TCL10	8-Bit Timer Counter 1 Count Clock Selection		
				MCS = 1		MCS = 0
0	0	0	0	TI1 falling edge		
0	0	0	1	TI1 rising edge		
0	1	1	0	$f_{xx}/2$	$f_x/2$ (2.25 MHz)	$f_x/2^2$ (1.13 MHz)
0	1	1	1	$f_{xx}/2^2$	$f_x/2^2$ (1.13 MHz)	$f_x/2^3$ (563 kHz)
1	0	0	0	$f_{xx}/2^3$	$f_x/2^3$ (563 kHz)	$f_x/2^4$ (281 kHz)
1	0	0	1	$f_{xx}/2^4$	$f_x/2^4$ (281 kHz)	$f_x/2^5$ (141 kHz)
1	0	1	0	$f_{xx}/2^5$	$f_x/2^5$ (141 kHz)	$f_x/2^6$ (70.3 kHz)
1	0	1	1	$f_{xx}/2^6$	$f_x/2^6$ (70.3 kHz)	$f_x/2^7$ (35.2 kHz)
1	1	0	0	$f_{xx}/2^7$	$f_x/2^7$ (35.2 kHz)	$f_x/2^8$ (17.6 kHz)
1	1	0	1	$f_{xx}/2^8$	$f_x/2^8$ (17.6 kHz)	$f_x/2^9$ (8.8 kHz)
1	1	1	0	$f_{xx}/2^9$	$f_x/2^9$ (8.8 kHz)	$f_x/2^{10}$ (4.4 kHz)
1	1	1	1	$f_{xx}/2^{11}$	$f_x/2^{11}$ (2.2 kHz)	$f_x/2^{12}$ (1.1 kHz)
Other than above				Setting prohibited		

TCL17	TCL16	TCL15	TCL14	8-Bit Timer Counter 2 Count Clock Selection		
				MCS = 1		MCS = 0
0	0	0	0	TI2 falling edge		
0	0	0	1	TI2 rising edge		
0	1	1	0	$f_{xx}/2$	$f_x/2$ (2.25 MHz)	$f_x/2^2$ (1.13 MHz)
0	1	1	1	$f_{xx}/2^2$	$f_x/2^2$ (1.13 MHz)	$f_x/2^3$ (563 kHz)
1	0	0	0	$f_{xx}/2^3$	$f_x/2^3$ (563 kHz)	$f_x/2^4$ (281 kHz)
1	0	0	1	$f_{xx}/2^4$	$f_x/2^4$ (281 kHz)	$f_x/2^5$ (141 kHz)
1	0	1	0	$f_{xx}/2^5$	$f_x/2^5$ (141 kHz)	$f_x/2^6$ (70.3 kHz)
1	0	1	1	$f_{xx}/2^6$	$f_x/2^6$ (70.3 kHz)	$f_x/2^7$ (35.2 kHz)
1	1	0	0	$f_{xx}/2^7$	$f_x/2^7$ (35.2 kHz)	$f_x/2^8$ (17.6 kHz)
1	1	0	1	$f_{xx}/2^8$	$f_x/2^8$ (17.6 kHz)	$f_x/2^9$ (8.8 kHz)
1	1	1	0	$f_{xx}/2^9$	$f_x/2^9$ (8.8 kHz)	$f_x/2^{10}$ (4.4 kHz)
1	1	1	1	$f_{xx}/2^{11}$	$f_x/2^{11}$ (2.2 kHz)	$f_x/2^{12}$ (1.1 kHz)
Other than above				Setting prohibited		

Caution When rewriting TCL1 to other data, stop the timer operation beforehand.

- Remarks**
1. f_{xx} : System clock frequency (f_x or $f_x/2$)
 2. f_x : System clock oscillation frequency
 3. TI1 : 8-bit timer counter 1 input pin
 4. TI2 : 8-bit timer counter 2 input pin
 5. MCS : Oscillation mode selection register (OSMS) bit 0
 6. Figures in parentheses apply to operation with $f_x = 4.5$ MHz

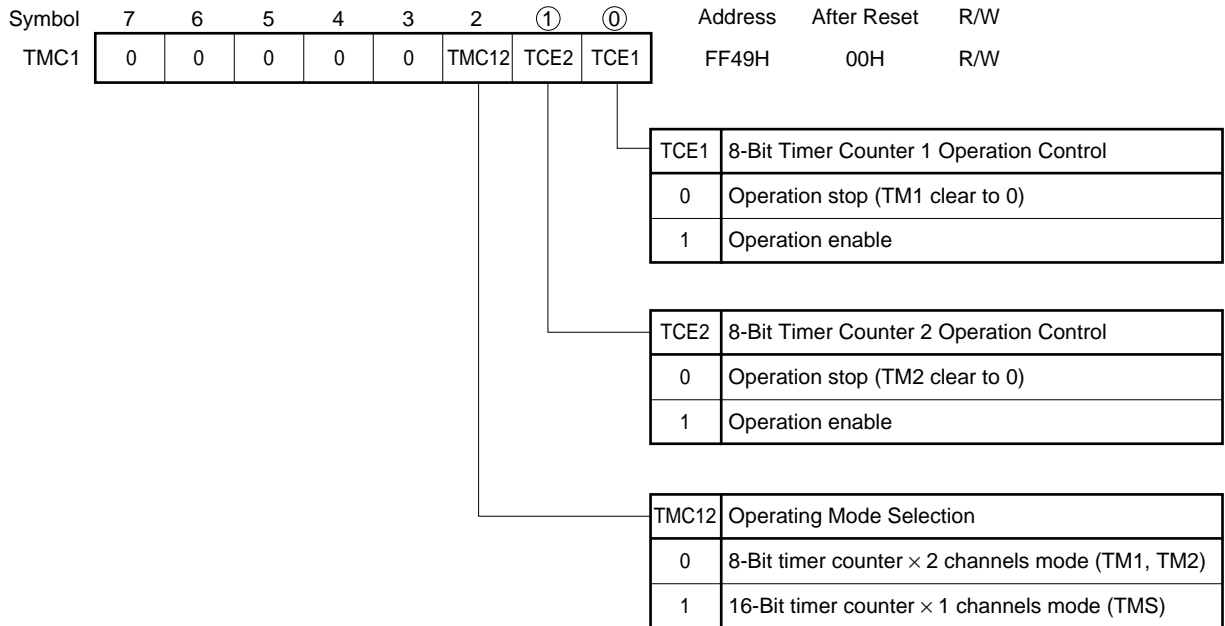
(2) 8-bit timer mode control register (TMC1)

This register enables/stops operation of 8-bit timer counters 1 and 2 and sets the operating mode of 8-bit timer counter 2.

TMC1 is set with a 1-bit or 8-bit memory manipulation instruction.

Reset input sets TMC1 to 00H.

Figure 6-3. 8-Bit Timer Mode Control Register (TMC1) Format



Cautions 1. Switch the operating mode after stopping timer operation.

2. When used as 16-bit timer counter, TCE1 should be used for operation enable/stop.

6.4 8-Bit Timer/Event Counters 1 and 2 Operations

6.4.1 8-bit timer/event counter mode

(1) Interval timer operations

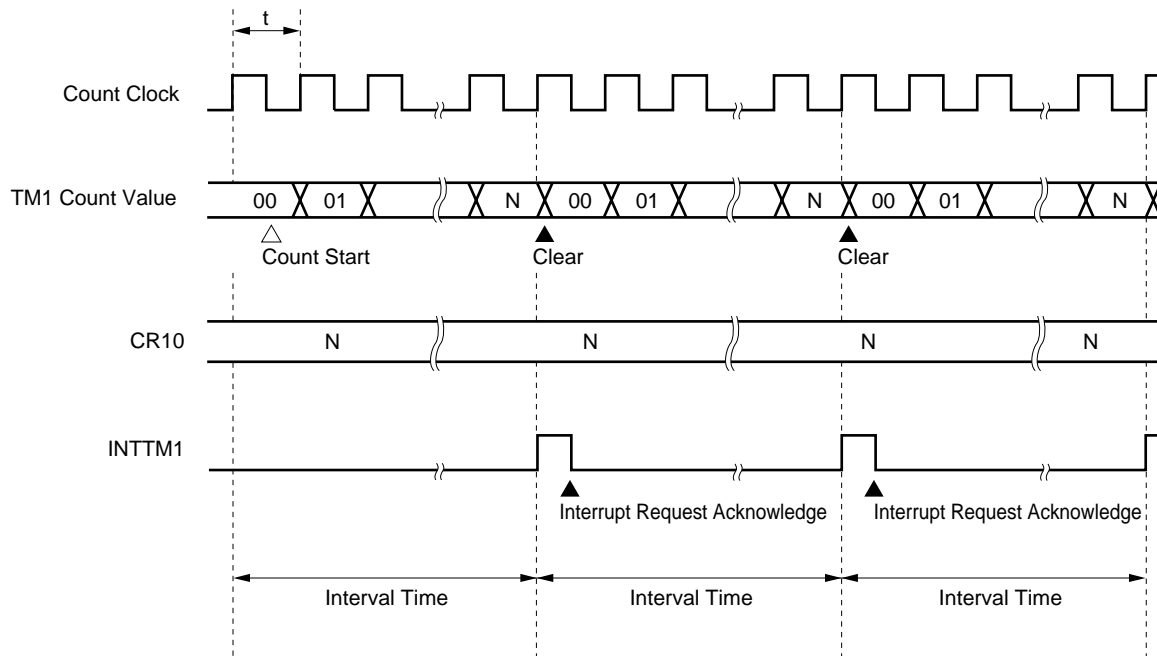
The 8-bit timer/event counters 1 and 2 operate as interval timers which generate interrupt requests repeatedly at intervals of the count value preset to 8-bit compare registers 10 and 20 (CR10 and CR20).

When the count values of the 8-bit timer counters 1 and 2 (TM1 and TM2) match the values set to CR10 and CR20, counting continues with the TM1 and TM2 values cleared to 0 and the interrupt request signals (INTTM1 and INTTM2) are generated.

Count clock of the 8-bit timer counter 1 (TM1) can be selected with bits 0 to 3 (TCL10 to TCL13) of the timer clock select register 1 (TCL1). Count clock of the 8-bit timer counter 2 (TM2) can be selected with bits 4 to 7 (TCL14 to TCL17) of the timer clock select register 1 (TCL1).

For the operation when the value of the compare register is changed during timer count operation, refer to **6.5 Cautions on 8-Bit Timer/Event Counters (3)**.

Figure 6-4. Interval Timer Operation Timings



Remark Interval time = $(N + 1) \times t$: N = 00H to FFH

Table 6-4. 8-Bit Timer/Event Counter 1 Interval Time

TCL13	TCL12	TCL11	TCL10	Minimum Interval Time		Maximum Interval Time		Resolution	
				MCS = 1	MCS = 0	MCS = 1	MCS = 0	MCS = 1	MCS = 0
0	0	0	0	Tl1 input cycle		$2^8 \times \text{Tl1 input cycle}$		Tl1 input edge cycle	
0	0	0	1	Tl1 input cycle		$2^8 \times \text{Tl1 input cycle}$		Tl1 input edge cycle	
0	1	1	0	$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)	$2^9 \times 1/f_x$ (113.8 μs)	$2^{10} \times 1/f_x$ (227.5 μs)	$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)
0	1	1	1	$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μs)	$2^{10} \times 1/f_x$ (227.5 μs)	$2^{11} \times 1/f_x$ (455.1 μs)	$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μs)
1	0	0	0	$2^3 \times 1/f_x$ (1.78 μs)	$2^4 \times 1/f_x$ (3.56 μs)	$2^{11} \times 1/f_x$ (455.1 μs)	$2^{12} \times 1/f_x$ (910.2 μs)	$2^3 \times 1/f_x$ (1.78 μs)	$2^4 \times 1/f_x$ (3.56 μs)
1	0	0	1	$2^4 \times 1/f_x$ (3.56 μs)	$2^5 \times 1/f_x$ (7.11 μs)	$2^{12} \times 1/f_x$ (910.2 μs)	$2^{13} \times 1/f_x$ (1.82 ms)	$2^4 \times 1/f_x$ (3.56 μs)	$2^5 \times 1/f_x$ (7.11 μs)
1	0	1	0	$2^5 \times 1/f_x$ (7.11 μs)	$2^6 \times 1/f_x$ (14.2 μs)	$2^{13} \times 1/f_x$ (1.82 ms)	$2^{14} \times 1/f_x$ (3.64 ms)	$2^5 \times 1/f_x$ (7.11 μs)	$2^6 \times 1/f_x$ (14.2 μs)
1	0	1	1	$2^6 \times 1/f_x$ (14.2 μs)	$2^7 \times 1/f_x$ (28.4 μs)	$2^{14} \times 1/f_x$ (3.64 ms)	$2^{15} \times 1/f_x$ (7.28 ms)	$2^6 \times 1/f_x$ (14.2 μs)	$2^7 \times 1/f_x$ (28.4 μs)
1	1	0	0	$2^7 \times 1/f_x$ (28.4 μs)	$2^8 \times 1/f_x$ (56.9 μs)	$2^{15} \times 1/f_x$ (7.28 ms)	$2^{16} \times 1/f_x$ (14.6 ms)	$2^7 \times 1/f_x$ (28.4 μs)	$2^8 \times 1/f_x$ (56.9 μs)
1	1	0	1	$2^8 \times 1/f_x$ (56.9 μs)	$2^9 \times 1/f_x$ (113.8 μs)	$2^{16} \times 1/f_x$ (14.6 ms)	$2^{17} \times 1/f_x$ (29.1 ms)	$2^8 \times 1/f_x$ (56.9 μs)	$2^9 \times 1/f_x$ (113.8 μs)
1	1	1	0	$2^9 \times 1/f_x$ (113.8 μs)	$2^{10} \times 1/f_x$ (227.5 μs)	$2^{17} \times 1/f_x$ (29.1 ms)	$2^{18} \times 1/f_x$ (58.3 ms)	$2^9 \times 1/f_x$ (113.8 μs)	$2^{10} \times 1/f_x$ (227.5 μs)
1	1	1	1	$2^{11} \times 1/f_x$ (455.1 μs)	$2^{12} \times 1/f_x$ (910.2 μs)	$2^{19} \times 1/f_x$ (116.5 ms)	$2^{20} \times 1/f_x$ (233.0 ms)	$2^{11} \times 1/f_x$ (455.1 μs)	$2^{12} \times 1/f_x$ (910.2 μs)
Other than above				Setting prohibited					

- Remarks**
1. f_x : System clock oscillation frequency
 2. MCS : Oscillation mode selection register (OSMS) bit 0
 3. TCL10-TCL13: Timer clock select register 1 (TCL1) bits 0-3
 4. Values in parentheses when operated at $f_x = 4.5 \text{ MHz}$.

Table 6-5. 8-Bit Timer/Event Counter 2 Interval Time

TCL17	TCL16	TCL15	TCL14	Minimum Interval Time		Maximum Interval Time		Resolution	
				MCS = 1	MCS = 0	MCS = 1	MCS = 0	MCS = 1	MCS = 0
0	0	0	0	TI2 input cycle		$2^8 \times \text{TI2 input cycle}$		TI2 input edge cycle	
0	0	0	1	TI2 input cycle		$2^8 \times \text{TI2 input cycle}$		TI2 input edge cycle	
0	1	1	0	$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)	$2^9 \times 1/f_x$ (113.8 μs)	$2^{10} \times 1/f_x$ (227.5 μs)	$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)
0	1	1	1	$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μs)	$2^{10} \times 1/f_x$ (227.5 μs)	$2^{11} \times 1/f_x$ (455.1 μs)	$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μs)
1	0	0	0	$2^3 \times 1/f_x$ (1.78 μs)	$2^4 \times 1/f_x$ (3.56 μs)	$2^{11} \times 1/f_x$ (455.1 μs)	$2^{12} \times 1/f_x$ (910.2 μs)	$2^3 \times 1/f_x$ (1.78 μs)	$2^4 \times 1/f_x$ (3.56 μs)
1	0	0	1	$2^4 \times 1/f_x$ (3.56 μs)	$2^5 \times 1/f_x$ (7.11 μs)	$2^{12} \times 1/f_x$ (910.2 μs)	$2^{13} \times 1/f_x$ (1.82 ms)	$2^4 \times 1/f_x$ (3.56 μs)	$2^5 \times 1/f_x$ (7.11 μs)
1	0	1	0	$2^5 \times 1/f_x$ (7.11 μs)	$2^6 \times 1/f_x$ (14.2 μs)	$2^{13} \times 1/f_x$ (1.82 ms)	$2^{14} \times 1/f_x$ (3.64 ms)	$2^5 \times 1/f_x$ (7.11 μs)	$2^6 \times 1/f_x$ (14.2 μs)
1	0	1	1	$2^6 \times 1/f_x$ (14.2 μs)	$2^7 \times 1/f_x$ (28.4 μs)	$2^{14} \times 1/f_x$ (3.64 ms)	$2^{15} \times 1/f_x$ (7.28 ms)	$2^6 \times 1/f_x$ (14.2 μs)	$2^7 \times 1/f_x$ (28.4 μs)
1	1	0	0	$2^7 \times 1/f_x$ (28.4 μs)	$2^8 \times 1/f_x$ (56.9 μs)	$2^{15} \times 1/f_x$ (7.28 ms)	$2^{16} \times 1/f_x$ (14.6 ms)	$2^7 \times 1/f_x$ (28.4 μs)	$2^8 \times 1/f_x$ (56.9 μs)
1	1	0	1	$2^8 \times 1/f_x$ (56.9 μs)	$2^9 \times 1/f_x$ (113.8 μs)	$2^{16} \times 1/f_x$ (14.6 ms)	$2^{17} \times 1/f_x$ (29.1 ms)	$2^8 \times 1/f_x$ (56.9 μs)	$2^9 \times 1/f_x$ (113.8 μs)
1	1	1	0	$2^9 \times 1/f_x$ (113.8 μs)	$2^{10} \times 1/f_x$ (227.5 μs)	$2^{17} \times 1/f_x$ (29.1 ms)	$2^{18} \times 1/f_x$ (58.3 ms)	$2^9 \times 1/f_x$ (113.8 μs)	$2^{10} \times 1/f_x$ (227.5 μs)
1	1	1	1	$2^{11} \times 1/f_x$ (455.1 μs)	$2^{12} \times 1/f_x$ (910.2 μs)	$2^{19} \times 1/f_x$ (116.5 ms)	$2^{20} \times 1/f_x$ (233.0 ms)	$2^{11} \times 1/f_x$ (455.1 μs)	$2^{12} \times 1/f_x$ (910.2 μs)
Other than above				Setting prohibited					

- Remarks**
1. f_x : System clock oscillation frequency
 2. MCS : Oscillation mode selection register (OSMS) bit 0
 3. TCL14-TCL17: Timer clock select register (TCL1) bits 4-7
 4. Values in parentheses when operated at $f_x = 4.5 \text{ MHz}$

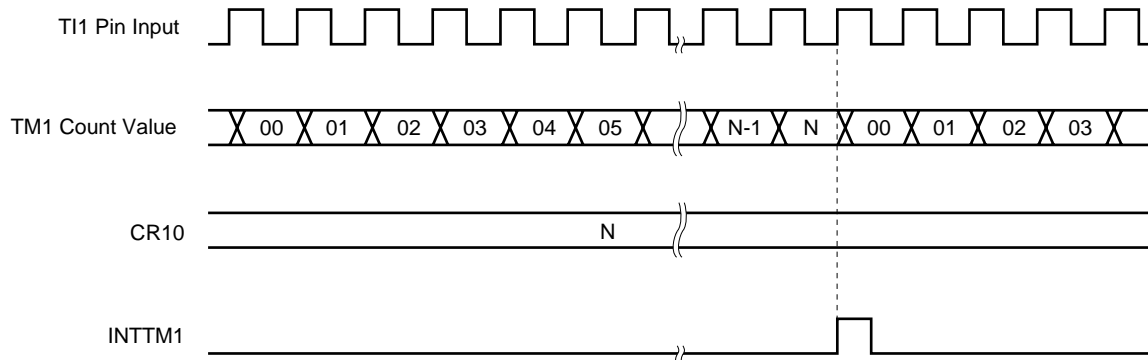
(2) External event counter operation

The external event counter counts the number of external clock pulses to be input to the T11/P33 and T12/P34 pins with 8-bit timer counters 1 and 2 (TM1 and TM2).

TM1 and TM2 are incremented each time the valid edge specified with the timer clock select register (TCL1) is input. Either the rising or falling edge can be selected.

When the TM1 and TM2 counted values match the values of 8-bit compare registers (CR10 and CR20), TM1 and TM2 are cleared to 0 and the interrupt request signals (INTTM1 and INTTM2) are generated.

Figure 6-5. External Event Counter Operation Timings (with Rising Edge Specified)



Remark N = 00H to FFH

6.4.2 16-bit timer/event counter mode

The 16-bit timer/event counter mode is selected if bit 2 (TMC12) of the 8-bit timer mode control register (TMC1) is set to 1.

In this mode, the count clock is selected by using bit 0 through 3 (TCL10 through TCL13) of the timer clock select register (TCL1). The overflow signal of 8-bit timer/event counter 1 (TM1) is used as the count clock to 8-bit timer/event counter 2 (TM2).

In this mode, counting is enabled or disabled by bit 0 (TCE1) of TMC1.

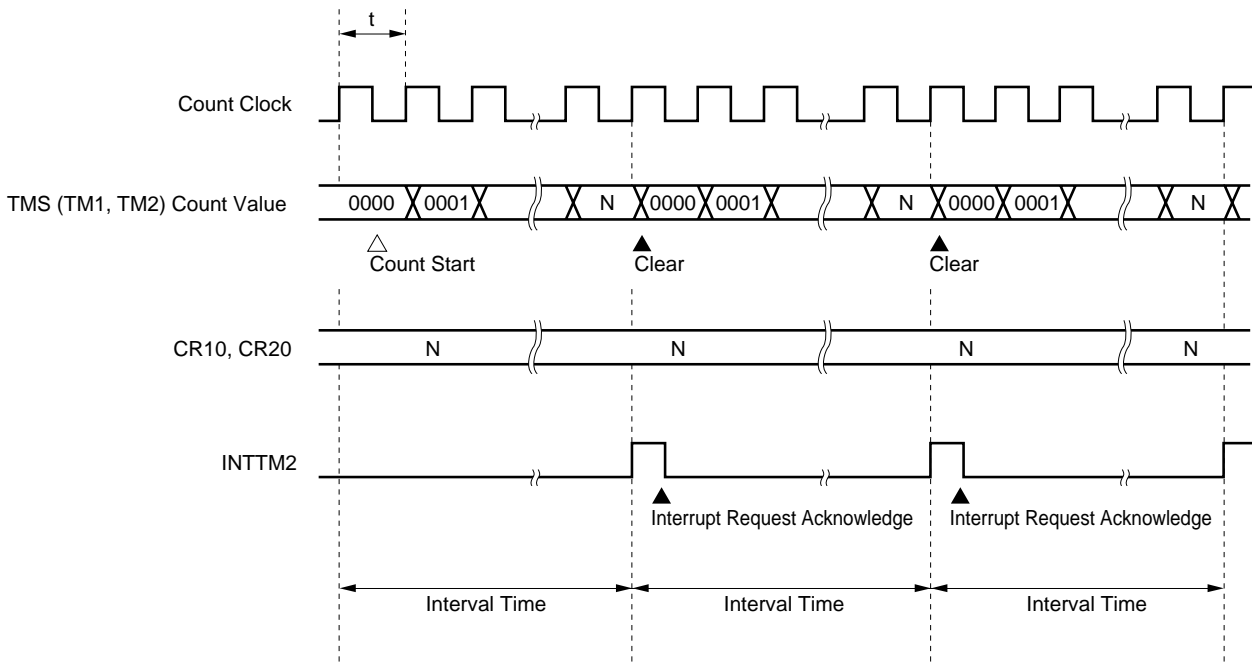
(1) Operation as interval timer

The 16-bit timer/event counter operates as an interval timer that repeatedly generates an interrupt request at interval specified by the count value assigned in advance to the 8-bit compare registers (CR10 and CR20) of the two channels. To set a count value, assigned the value of high-order 8 bits to CR20, and the value of the low-order 8 bits to CR10. For details of the count values (interval time) that can be set, refer to **Table 6-6**.

When the value of 8-bit timer counter 1 (TM1) coincides with the value of CR10, and when the value of 8-bit timer counter 2 (TM2) coincides with the value of CR20, the values of TM1 and TM2 are cleared to 0, and TM1 and TM2 continue counting. At the same time, an interrupt request signal is generated (INTTM2). For details of the operation timing of the interval timer, refer to **Figure 6-6**.

Bits 0 through 3 (TCL10 through TCL13) select the count clock of timer clock select register 1. The overflow signal of TM1 is used as the count clock to TM2.

Figure 6-6. Interval Timer Operation Timing



Remark Interval time = $(N + 1) \times t$: $N = 0000H$ to $FFFFH$

Caution Even if the 16-bit timer/event counter mode is used, when the TM1 count value matches the CR10 value, interrupt request signal (INTTM1) is generated. Thus, when using 8-bit timer/event counter as 16-bit interval timer, set the INTTM1 mask flag TMMK1 to 1 to disable INTTM1 acknowledgment.

When reading the 16-bit timer (TMS) count value, use the 16-bit memory manipulation instruction.

Table 6-6. Interval Times when 2-Channel 8-Bit Timer/Event Counters (TM1 and TM2) are Used as 16-Bit Timer/Event Counter

TCL13	TCL12	TCL11	TCL10	Minimum Interval Time		Maximum Interval Time		Resolution	
				MCS = 1	MCS = 0	MCS = 1	MCS = 0	MCS = 1	MCS = 0
0	0	0	0	Tl1 input cycle		$2^8 \times \text{Tl1 input cycle}$		Tl1 input edge cycle	
0	0	0	1	Tl1 input cycle		$2^8 \times \text{Tl1 input cycle}$		Tl1 input edge cycle	
0	1	1	0	$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)	$2^{17} \times 1/f_x$ (29.1 ms)	$2^{18} \times 1/f_x$ (58.3 ms)	$2 \times 1/f_x$ (444 ns)	$2^2 \times 1/f_x$ (888 ns)
0	1	1	1	$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μs)	$2^{18} \times 1/f_x$ (58.3 ms)	$2^{19} \times 1/f_x$ (116.5 ms)	$2^2 \times 1/f_x$ (888 ns)	$2^3 \times 1/f_x$ (1.78 μs)
1	0	0	0	$2^3 \times 1/f_x$ (1.78 μs)	$2^4 \times 1/f_x$ (3.56 μs)	$2^{19} \times 1/f_x$ (116.5 ms)	$2^{20} \times 1/f_x$ (233.0 ms)	$2^3 \times 1/f_x$ (1.78 μs)	$2^4 \times 1/f_x$ (3.56 μs)
1	0	0	1	$2^4 \times 1/f_x$ (3.56 μs)	$2^5 \times 1/f_x$ (7.11 μs)	$2^{20} \times 1/f_x$ (233.0 ms)	$2^{21} \times 1/f_x$ (466.0 ms)	$2^4 \times 1/f_x$ (3.56 μs)	$2^5 \times 1/f_x$ (7.11 μs)
1	0	1	0	$2^5 \times 1/f_x$ (7.11 μs)	$2^6 \times 1/f_x$ (14.2 μs)	$2^{21} \times 1/f_x$ (466.0 ms)	$2^{22} \times 1/f_x$ (932.0 ms)	$2^5 \times 1/f_x$ (7.11 μs)	$2^6 \times 1/f_x$ (14.2 μs)
1	0	1	1	$2^6 \times 1/f_x$ (14.2 μs)	$2^7 \times 1/f_x$ (28.4 μs)	$2^{22} \times 1/f_x$ (932.0 ms)	$2^{23} \times 1/f_x$ (1.9 s)	$2^6 \times 1/f_x$ (14.2 μs)	$2^7 \times 1/f_x$ (28.4 μs)
1	1	0	0	$2^7 \times 1/f_x$ (28.4 μs)	$2^8 \times 1/f_x$ (56.9 μs)	$2^{23} \times 1/f_x$ (1.9 s)	$2^{24} \times 1/f_x$ (3.7 s)	$2^7 \times 1/f_x$ (28.4 μs)	$2^8 \times 1/f_x$ (56.9 μs)
1	1	0	1	$2^8 \times 1/f_x$ (56.9 μs)	$2^9 \times 1/f_x$ (113.8 μs)	$2^{24} \times 1/f_x$ (3.7 s)	$2^{25} \times 1/f_x$ (7.5 s)	$2^8 \times 1/f_x$ (56.9 μs)	$2^9 \times 1/f_x$ (113.8 μs)
1	1	1	0	$2^9 \times 1/f_x$ (113.8 μs)	$2^{10} \times 1/f_x$ (227.5 μs)	$2^{25} \times 1/f_x$ (7.5 s)	$2^{26} \times 1/f_x$ (14.9 s)	$2^9 \times 1/f_x$ (113.8 μs)	$2^{10} \times 1/f_x$ (227.5 μs)
1	1	1	1	$2^{11} \times 1/f_x$ (455.1 μs)	$2^{12} \times 1/f_x$ (910.2 μs)	$2^{27} \times 1/f_x$ (29.8 s)	$2^{28} \times 1/f_x$ (59.7 s)	$2^{11} \times 1/f_x$ (455.1 μs)	$2^{12} \times 1/f_x$ (910.2 μs)
Other than above				Setting prohibited					

- Remarks**
1. f_x : System clock oscillation frequency
 2. MCS : Oscillation mode selection register (OSMS) bit 0
 3. TCL10-TCL13: Timer clock select register 1 (TCL1) bits 0-3
 4. Values in parentheses when operated at $f_x = 4.5 \text{ MHz}$.

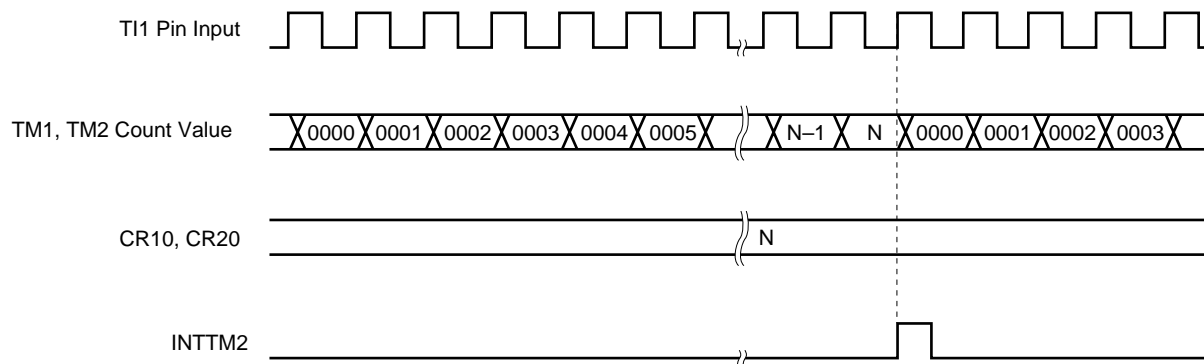
(2) External event counter operations

The external event counter counts the number of external clock pulses to be input to the TI1/P33 pin with 2-channel 8-bit timer counters 1 and 2 (TM1 and TM2).

Each time the valid edge specified by the timer clock select register 1 (TCL1) is input, TM1 is incremented. When TM1 overflows, its overflow signal is used as a count clock and TM2 is incremented. Either rising or falling edge can be specified as the valid edge.

When the TM1 and TM2 counted values match the values of 8-bit compare registers 10 and 20 (CR10 and CR20), TM1 and TM2 are cleared to 0 and the interrupt request signal (INTTM2) is generated.

Figure 6-7. External Event Counter Operation Timings (with Rising Edge Specified)



Caution Even if the 16-bit timer/event counter mode is used, when the TM1 count value matches the CR10 value, interrupt request signal (INTTM1) is generated. Thus, when using 8-bit timer/event counter as 16-bit interval timer, set the INTTM1 mask flag TMMK1 to 1 to disable INTTM1 acknowledgment.

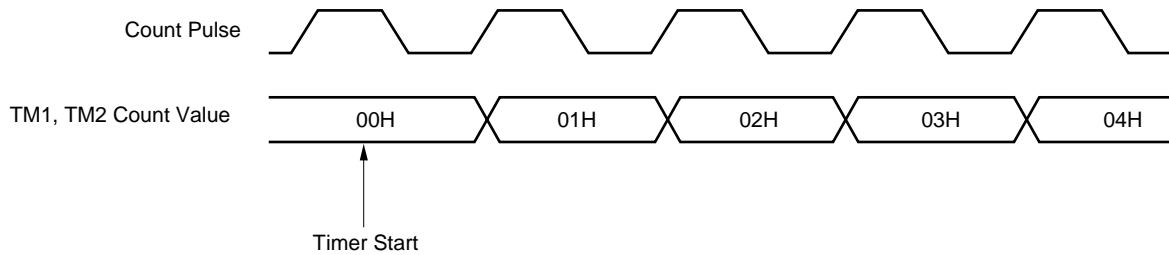
When reading the 16-bit timer (TMS) count value, use the 16-bit memory manipulation instruction.

6.5 Cautions on 8-Bit Timer/Event Counters

(1) Timer start errors

An error with a maximum of one clock may occur concerning the time required for a match signal to be generated after timer start. This is because 8-bit timer counters 1 and 2 (TM1 and TM2) are started asynchronously with the count pulse.

Figure 6-8. 8-Bit Timer Registers 1 and 2 Start Timing



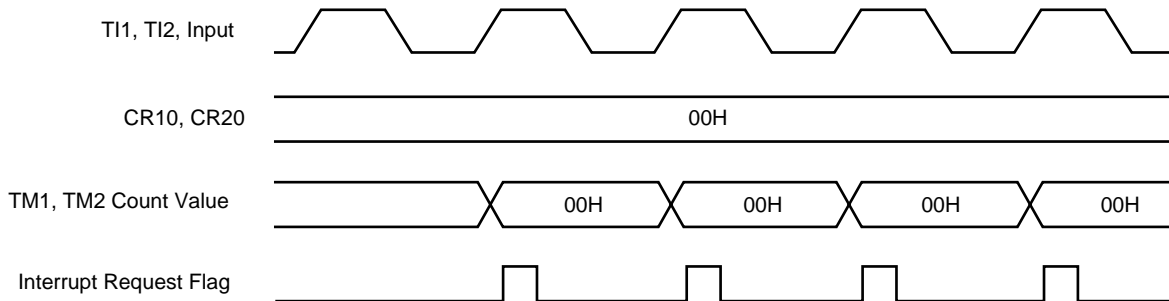
(2) Compare register 10 and 20 setting

The 8-bit compare registers 10 and 20 (CR10 and CR20) can be set to 00H.

Thus, when these 8-bit compare registers are used as event counters, one-pulse count operation can be carried out.

When the 8-bit compare register is used as 16-bit timer/event counter, write data to CR10 and CR20 after setting bit 0 (TCE1) of the 8-bit timer mode control register 1 (TMC1) to 0 and stopping timer operation.

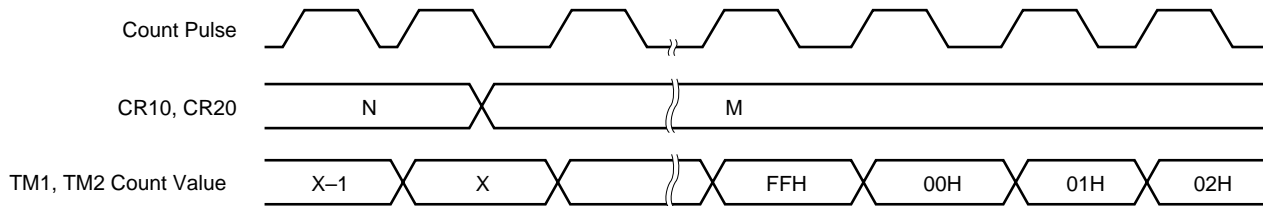
Figure 6-9. Event Counter Operation Timing



(3) Operation after compare register change during timer count operation

If the values after the 8-bit compare registers 10 and 20 (CR10 and CR20) are changed are smaller than those of 8-bit timer counters (TM1 and TM2), TM1 and TM2 continue counting, overflow and then restart counting from 0. Thus, if the value (M) after CR10 and CR20 change is smaller than value (N) before the change, it is necessary to restart the timer after changing CR10 and CR20.

Figure 6-10. Timing after Compare Register Change during Timer Count Operation



Remark $N > X > M$

CHAPTER 7 BASIC TIMER

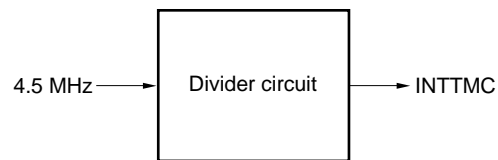
The basic timer is used for time management during program execution.

7.1 Function of Basic Timer

The basic timer generates an interrupt request signal (INTTMC) at time intervals of 100 ms.

7.2 Configuration of Basic Timer

Figure 7-1. Basic Timer Block Diagram



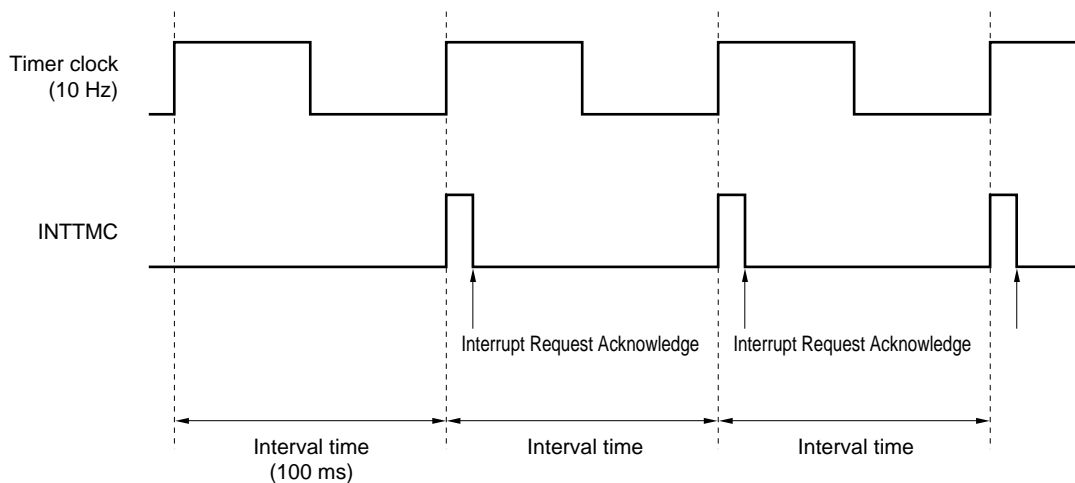
7.3 Operation of Basic Timer

An example of the operation of the basic timer is shown below.

In this example, the basic timer operates as an interval timer that repeatedly generates an interrupt at time intervals of 100 ms. Interrupt request signal INTTMC is generated every 100 ms.

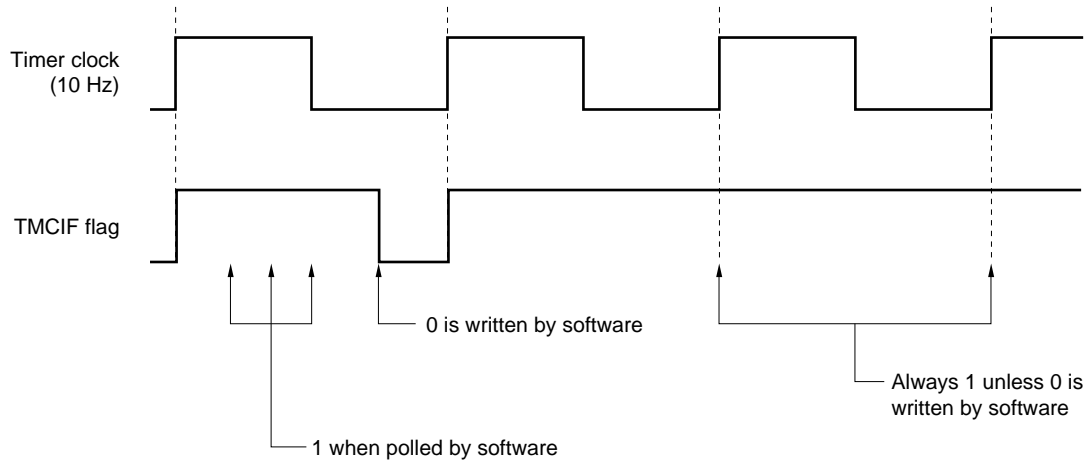
The timer clock frequency is 10 Hz.

Figure 7-2. Basic Timer Operation Timing



By polling the interrupt request flag (TMCIF) of this basic timer by software, time management can be carried out. Note that TMCIF is not a Read & Reset flag.

Figure 7-3. Operating Timing to Poll TMCIF Flag



CHAPTER 8 BUZZER OUTPUT CONTROL CIRCUIT

8.1 Buzzer Output Control Circuit Functions

The buzzer output control circuit outputs 1.5 kHz, 3 kHz, or 6 kHz frequency square waves. The buzzer frequency selected with timer clock select register 2 (TCL2) is output from the BEEP/P36 pin.

Follow the procedure below to output the buzzer frequency.

- (1) Select the buzzer output frequency with bits 5 to 7 (TCL25 to TCL27) of TCL2.
- (2) Set the P36 output latch to 0.
- (3) Set bit 6 (PM36) of port mode register 3 to 0 (Set to output mode).

Caution Buzzer output cannot be used when setting P36 output latch to 1.

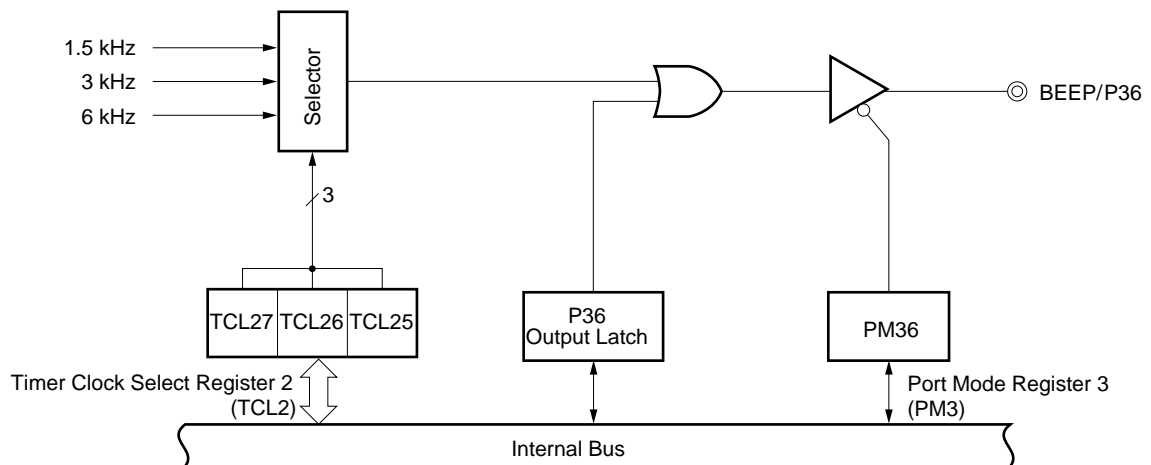
8.2 Buzzer Output Control Circuit Configuration

The buzzer output control circuit consists of the following hardware.

Table 8-1. Buzzer Output Control Circuit Configuration

Item	Configuration
Control register	Timer clock select register 2 (TCL2) Port mode register 3 (PM3)

Figure 8-1. Buzzer Output Control Circuit Block Diagram



8.3 Buzzer Output Function Control Registers

The following two types of registers are used to control the buzzer output function.

- Timer clock select register 2 (TCL2)
- Port mode register 3 (PM3)

(1) Timer clock select register 2 (TCL2)

This register sets the buzzer output frequency.

TCL2 is set with an 8-bit memory manipulation instruction.

Reset input sets TCL2 to 00H.

Figure 8-2. Timer Clock Select Register 2 (TCL2) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCL2	TCL27	TCL26	TCL25	TCL24	0	0	0	0	FF42H	00H	R/W

TCL27	TCL26	TCL25	Buzzer Output Frequency Selection
			MCS = 0 or 1
0	×	×	Buzzer output disable (port output)
1	0	0	6 kHz
1	0	1	3 kHz
1	1	0	1.5 kHz
1	1	1	Setting prohibited

Caution When rewriting TCL2 to other data, stop the timer operation beforehand.

Remarks 1. × : Don't care

2. MCS : Oscillation mode selection register (OSMS) bit 0

(2) Port mode register 3 (PM3)

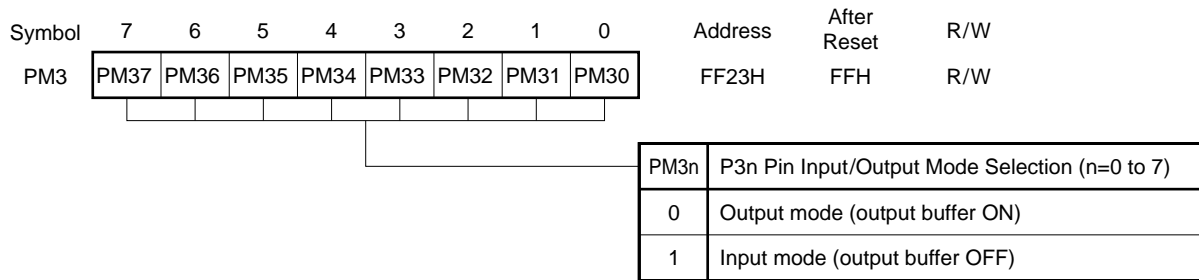
This register sets port 3 input/output in 1-bit units.

When using the P36/BEEP pin for buzzer output function, set PM36 and output latch of P36 to 0.

PM3 is set with a 1-bit or 8-bit memory manipulation instruction.

Reset input sets PM3 to FFH.

Figure 8-3. Port Mode Register 3 (PM3) Format



[MEMO]

CHAPTER 9 A/D CONVERTER

9.1 A/D Converter Functions

- ★ The A/D converter converts an analog input into a digital value. It consists of 3 channels (ANI0 to ANI2) with an 8-bit resolution.

The conversion method is based on successive approximation and the conversion result is held in the 8-bit A/D conversion result register (ADCR).

A/D conversion is started (software start) by setting the A/D converter mode register (ADM).

- ★ Select one channel of analog input from ANI0 to ANI2 and carry out A/D conversion. Each time an A/D conversion operation ends, the next conversion starts immediately, and an interrupt request (INTAD) is generated.

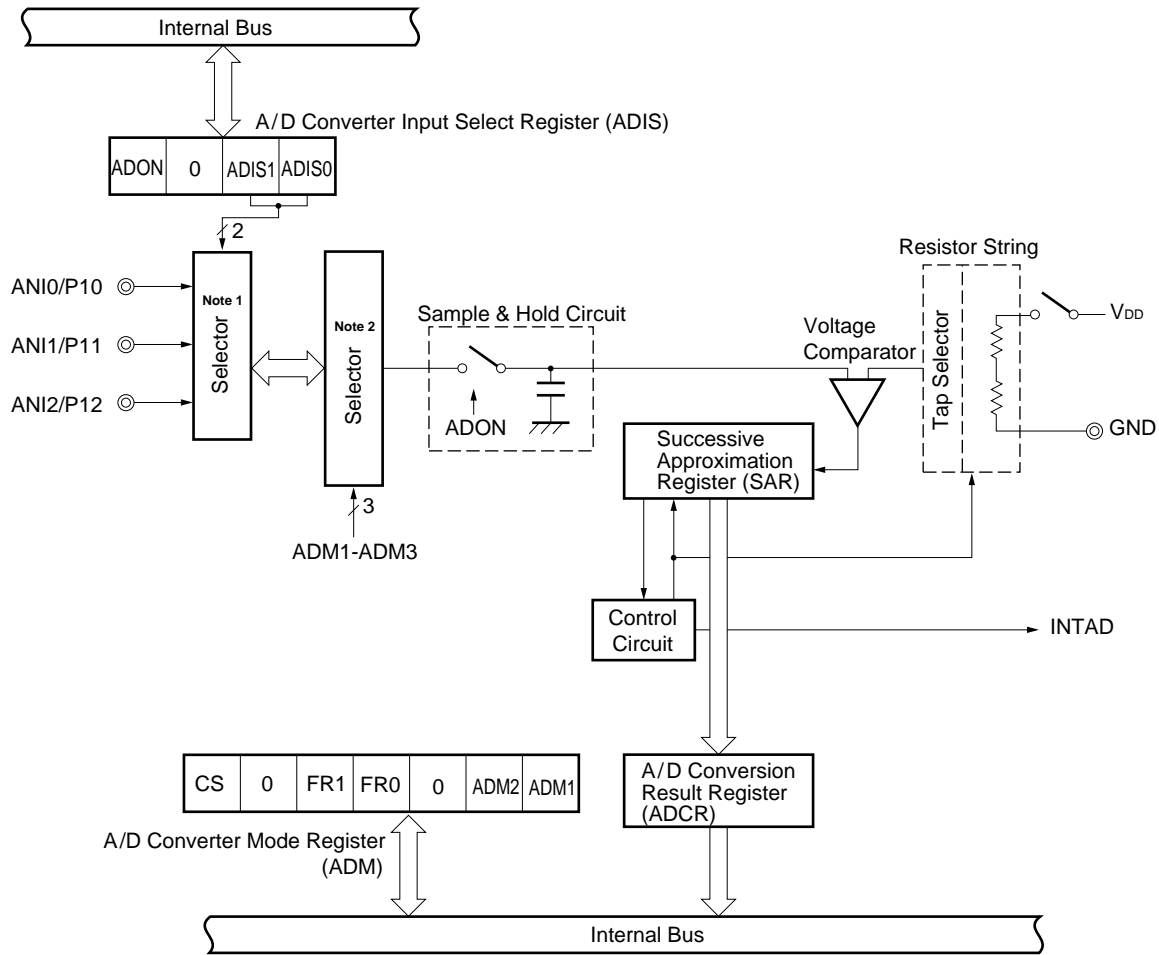
9.2 A/D Converter Configuration

The A/D converter consists of the following hardware.

Table 9-1. A/D Converter Configuration

Item	Configuration
Analog input	3 channels (ANI0 to ANI2)
Control register	A/D converter mode register (ADM) A/D converter input select register (ADIS)
Register	Successive approximation register (SAR) A/D conversion result register (ADCR)

Figure 9-1. A/D Converter Block Diagram



- Notes**
1. Selector to select the number of channels to be used for analog input.
 2. Selector to select the channel for A/D conversion.

(1) Successive approximation register (SAR)

This register compares the analog input voltage value to the voltage tap (compare voltage) value applied from the series resistor string and holds the result from the most significant bit (MSB).

When up to the least significant bit (LSB) is set (termination of A/D conversion), the SAR contents are transferred to the A/D conversion result register (ADCR).

(2) A/D conversion result register (ADCR)

This register holds the A/D conversion result. Each time A/D conversion terminates, the conversion result is loaded from the successive approximation register (SAR).

ADCR is read with an 8-bit memory manipulation instruction.

Reset input makes ADCR undefined.

(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 to the series resistor string output voltage.

(5) Resistor string

The resistor string is connected between V_{DD} and GND, and generates a voltage to be compared to the analog input.

★

(6) ANI0 to ANI2 pins

These are 3-channel analog input pins to input analog signals to undergo A/D conversion to the A/D converter. Pins other than those selected as analog input by the A/D converter input select register (ADIS) can be used as input/output ports.

Caution Use ANI0 to ANI2 input voltages within the specified range. If a voltage higher than V_{DD} or lower than GND is applied (even if within the absolute maximum ratings), the converted value of the corresponding channel becomes indeterminate and may adversely affect the converted values of other channels.

9.3 A/D Converter Control Registers

The following two types of registers are used to control the A/D converter.

- A/D converter mode register (ADM)
- A/D converter input select register (ADIS)

(1) A/D converter mode register (ADM)

This register sets the analog input channel for A/D conversion, conversion time, and conversion start/stop. ADM is set with a 1-bit or 8-bit memory manipulation instruction.

Reset input sets ADM to 01H.

★

Figure 9-2. A/D Converter Mode Register (ADM) Format

Symbol	⑦	6	5	4	3	2	1	0	Address	After Reset	R/W
ADM	CS	0	FR1	FR0	0	ADM2	ADM1	HSC	FF80H	01H	R/W

CS	A/D Conversion Operation Control
0	Operation stop
1	Operation start

FR1	FR0	HSC	A/D Conversion Time Selection ^{Note 1}
			$f_x = 4.5 \text{ MHz}$ Operation
			MCS = 0 or 1
0	0	1	$160/f_x$ (35.6 μs)
0	1	1	$80/f_x$ (Setting prohibited ^{Note 2})
1	0	0	$100/f_x$ (22.2 μs)
1	0	1	$200/f_x$ (44.4 μs)
Other than above			Setting prohibited

ADM2	ADM1	Analog Input Channel Selection
0	0	ANI0
0	1	ANI1
1	0	ANI2
Other than above		Setting prohibited

- Notes**
1. Set so that the A/D conversion time is 19.1 μs or more.
 2. Setting prohibited because A/D conversion time is less than 19.1 μs .

- Cautions**
1. The following sequence is recommended for power consumption reduction of A/D converter when the standby function is used: Clear bit 7 (CS) to 0 first to stop the A/D conversion operation, and then execute the HALT or STOP instruction.
 2. When restarting the stopped A/D conversion operation, start the A/D conversion operation after clearing the interrupt request flag (ADIF) to 0.
 3. Be sure to set bit 3 of ADM to 0.

- Remarks**
1. f_x : System clock oscillation frequency
 2. MCS : Oscillation mode selection register (OSMS) bit 0

(2) A/D converter input select register (ADIS)

This register determines whether the ANI0/P10 to ANI2/P12 pins should be used for analog input channels or ports. Pins other than those selected as analog input can be used as input/output ports.

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

Reset input sets ADIS to 00H.

Cautions 1. Set the analog input channel in the following order.

(1) Set the number of analog input channels with ADIS.

(2) Using A/D converter mode register (ADM), select one channel to undergo A/D conversion from among the channels set for analog input with ADIS.

2. When bit 3 (ADON) of ADIS is set to 1, the current consumption of the A/D converter increases in the standby mode. Clear ADON to 0 while A/D conversion is stopped.

★ **Figure 9-3. A/D Converter Input Select Register (ADIS) Format**

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
ADIS	0	0	0	0	ADON	0	ADIS1	ADIS0	FF84H	00H	R/W

ADON	Control of Resistor String Power Supply
0	Power OFF
1	Power ON

ADIS1	ADIS0	Number of Analog Input Channel Selection
0	0	No analog input channel (P10-P12)
0	1	1 channel (ANI0, P11, P12)
1	0	2 channel (ANI0, ANI1, P12)
1	1	3 channel (ANI0-ANI2)
Other than above		Setting prohibited

Caution Be sure to set bit 2 of ADIS to 0.

9.4 A/D Converter Operations

9.4.1 Basic operations of A/D converter

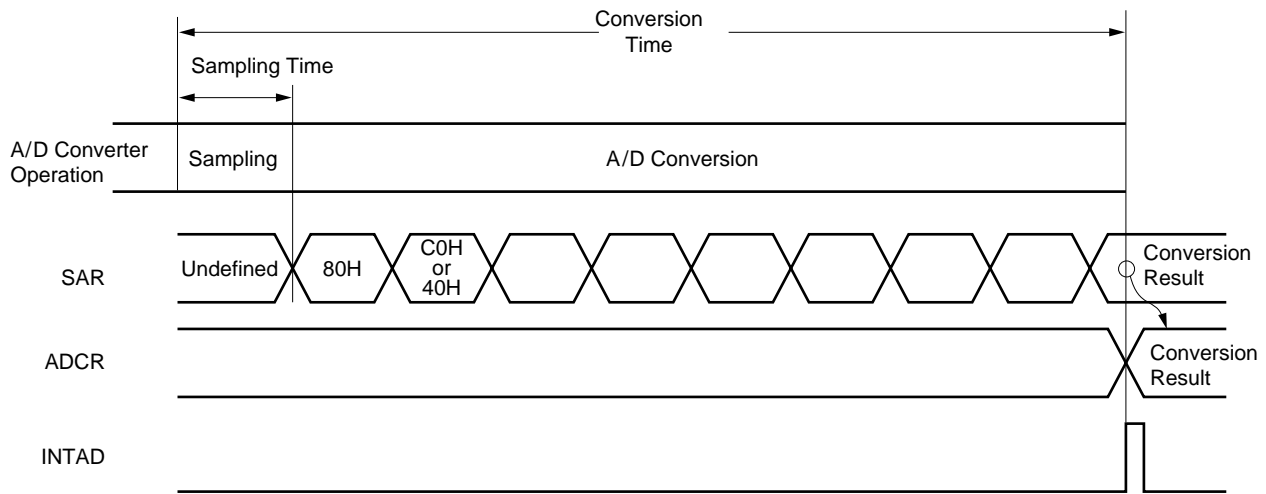
- (1) Set the number of analog input channels with A/D converter input select register (ADIS) and turn A/D converter power supply on.
- (2) From among the analog input channels set with ADIS, select one channel for A/D conversion with A/D converter mode register (ADM).
- (3) Sample the voltage input to the selected analog input channel with the sample & hold circuit.
- (4) Sampling for the specified period of time sets the sample & hold circuit to the hold state so that the circuit holds the input analog voltage until termination of A/D conversion.
- (5) Bit 7 of the successive approximation register (SAR) is set and the tap selector sets the resistor string voltage tap to $(1/2) V_{DD}$.
- (6) The voltage difference between the series resistor string voltage tap and analog input is compared with a voltage comparator. If the analog input is greater than $(1/2) V_{DD}$, the MSB of SAR remains set. If the input is smaller than $(1/2) V_{DD}$, the MSB is reset.
- (7) Next, bit 6 of SAR is automatically set and the operation proceeds to the next comparison. In this case, the resistor string voltage tap is selected according to the preset value of bit 7 as described below.
 - Bit 7 = 1 : $(3/4) V_{DD}$
 - Bit 7 = 0 : $(1/4) V_{DD}$

The voltage tap and analog input voltage are compared and bit 6 of SAR is manipulated with the result as follows.

- Analog input voltage \geq Voltage tap : Bit 6 = 1
 - Analog input voltage $<$ Voltage tap : Bit 6 = 0
- (8) Comparison of this sort continues up to bit 0 of SAR.
 - (9) Upon completion of the comparison of 8 bits, any effective digital resultant value remains in SAR and the resultant value is transferred to and latched in the A/D conversion result register (ADCR).

At the same time, the A/D conversion termination interrupt request signal (INTAD) can also be generated.

Figure 9-4. A/D Converter Basic Operation



A/D conversion operations are performed continuously until the bit 7 (CS) of the A/D converter mode register (ADM) is cleared (0) by software.

If a write to the ADM is performed during an A/D conversion operation, the conversion operation is initialized, and if the CS is set (1), conversion starts again from the beginning.

After reset input, the value of ADCR is undefined.

9.4.2 Input voltage and conversion results

- ★ The relation between the analog input voltage input to the analog input pins (ANI0 to ANI2) and the A/D conversion result (the value stored in ADCR) is shown by the following expression.

$$\text{ADCR} = \text{INT} \left(\frac{V_{\text{IN}}}{V_{\text{DD}}} \times 256 + 0.5 \right)$$

or

$$(\text{ADCR} - 0.5) \times \frac{V_{\text{DD}}}{256} \leq V_{\text{IN}} < (\text{ADCR} + 0.5) \times \frac{V_{\text{DD}}}{256}$$

Where, INT () : Function which returns integer parts of value in parentheses.

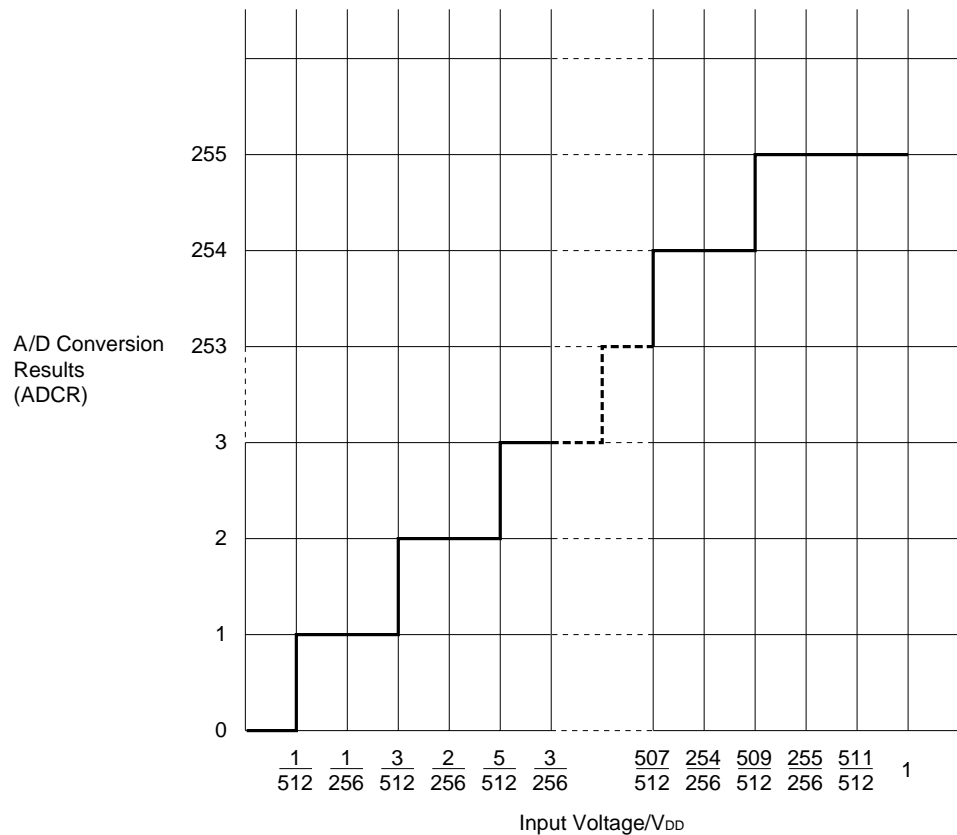
V_{IN} : Analog input voltage

V_{DD} : Supply voltage

ADCR : ADCR register value

Figure 9-5 shows the relation between the analog input voltage and the A/D conversion result.

Figure 9-5. Relations between Analog Input Voltage and A/D Conversion Result



9.4.3 A/D converter operating mode

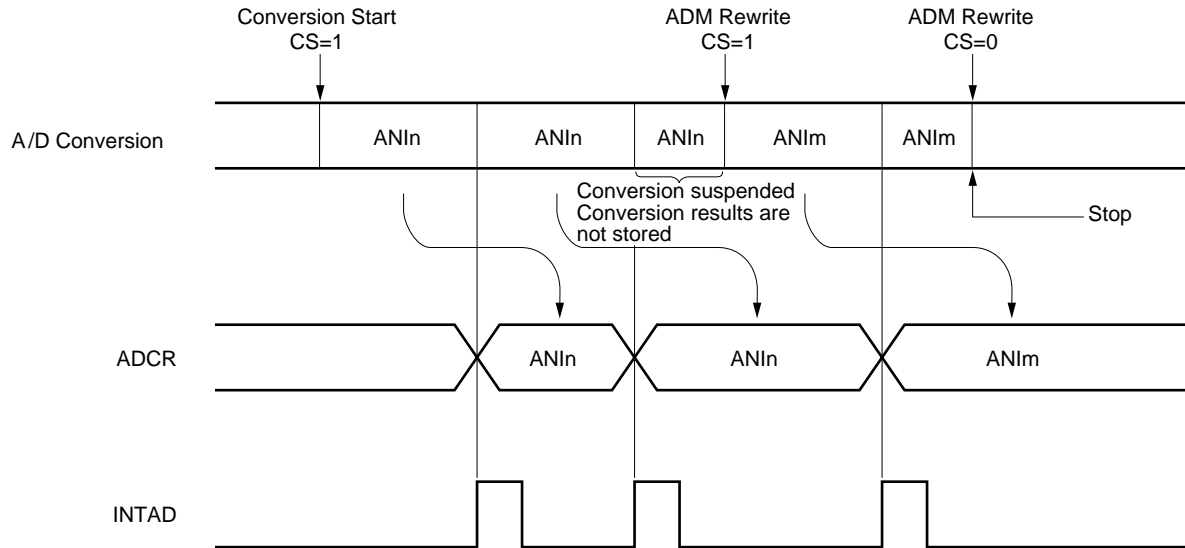
★ When bit 7 (CS) of A/D converter mode register (ADM) is set to 1, respectively, the A/D conversion starts on the voltage applied to the analog input pins specified with bits 1 to 2 (ADM1 to ADM2) of ADM.

Upon termination of the A/D conversion, the conversion result is stored in the A/D conversion result register (ADCR) and the interrupt request signal (INTAD) is generated. After one A/D conversion operation is started and terminated, the next A/D conversion operation starts immediately. The A/D conversion operation continues repeatedly until new data is written to ADM.

If data with CS set to 1 is written to ADM again during A/D conversion, the converter suspends its A/D conversion operation and starts A/D conversion on the newly written data.

If data with CS set to 0 is written to ADM during A/D conversion, the A/D conversion operation stops immediately.

Figure 9-6. A/D Conversion



- Remarks**
1. $n = 0, 1, 2$
 2. $m = 0, 1, 2$

9.5 A/D Converter Cautions

(1) Current consumption in standby mode

The A/D converter operates on the system clock. Therefore, its operation stops in STOP mode, but a current still flows in the resistor string.

To reduce the current consumption, clear bit 3 (ADON) of the A/D converter input select register (ADIS) to 0 before executing the HALT or STOP instruction. To reduce the power consumption, clear bit 7 (CS) of the A/D converter mode register (ADM) to 0 and stop A/D conversion before executing the HALT or STOP instruction.

★

(2) Input range of ANI0 to ANI2

The input voltages of ANI0 to ANI2 should be within the specification range. In particular, if a voltage above V_{DD} or below GND is input (even if within the absolute maximum rating range), the conversion value for that channel will be indeterminate. The conversion values of the other channels may also be affected.

★

(3) Pins ANI0/P10 to ANI2/P12

The analog input pins ANI0 to ANI2 also function as input/output port (PORT1) pins. When A/D conversion is performed with any of pins ANI0 to ANI2 selected, be sure not to execute a PORT1 input instruction while conversion is in progress, as this may reduce the conversion resolution.

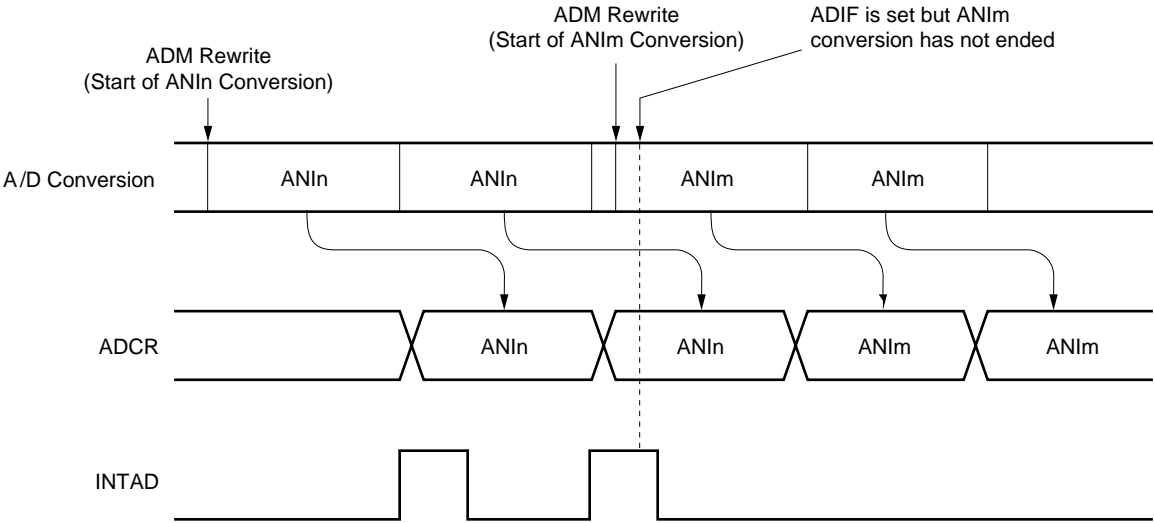
Also, if digital pulses are applied to a pin adjacent to the pin in the process of A/D conversion, the expected A/D conversion value may not be obtainable due to coupling noise. Therefore, avoid applying pulses to pins adjacent to the pin undergoing A/D conversion.

(4) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if the A/D converter mode register (ADM) is changed. Caution is therefore required since, if a change of analog input pin is performed during A/D conversion, the A/D conversion result and ADIF for the pre-change analog input may be set just before the ADM rewrite, and when ADIF is read immediately after the ADM rewrite, ADIF may be set despite the fact that the A/D conversion for the post-change analog input has not ended.

When the A/D conversion is stopped and then resumed, clear the interrupt request flag (ADIF) before it is resumed.

Figure 9-7. A/D Conversion End Interrupt Request Generation Timing



Remarks 1. $n = 0, 1, 2$
2. $m = 0, 1, 2$

CHAPTER 10 SERIAL INTERFACE CHANNEL 1

10.1 Serial Interface Channel 1 Functions

Serial interface channel 1 employs the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

(1) Operation stop mode

This mode is used when serial transfer is not carried out to reduce power consumption.

(2) 3-wire serial I/O mode (MSB first)

This mode is used for 8-bit data transfer using three lines, each for serial clock ($\overline{\text{SCK1}}$), serial output (SO1) and serial input (SI1).

The 3-wire serial I/O mode enables simultaneous transmission/reception and so decreases the data transfer processing time.

The start bit of 8-bit data to undergo serial transfer is MSB.

The 3-wire serial I/O mode is valid for connection of peripheral I/O units and display controllers which incorporate a conventional synchronous serial interface such as the 75X/XL, 78K and 17K series.

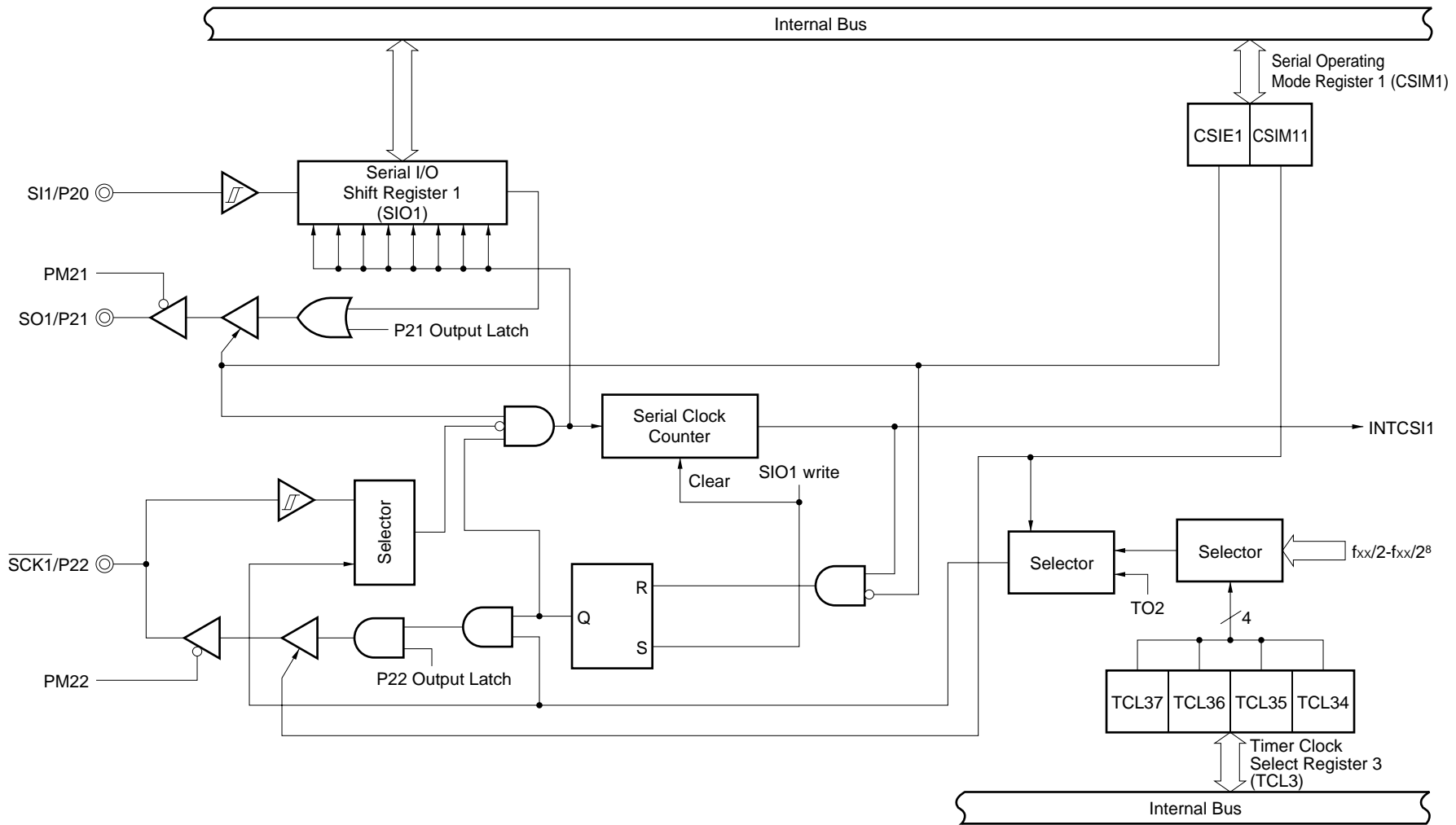
10.2 Serial Interface Channel 1 Configuration

Serial interface channel 1 consists of the following hardware.

Table 10-1. Serial Interface Channel 1 Configuration

Item	Configuration
Register	Serial I/O shift register 1 (SIO1)
Control register	Timer clock select register 3 (TCL3) Serial operating mode register 1 (CSIM1) Port mode register 2 (PM2)

Figure 10-1. Serial Interface Channel 1 Block Diagram



(1) Serial I/O shift register 1 (SIO1)

This is an 8-bit register to carry out parallel/serial conversion and to carry out serial transmission/reception (shift operation) in synchronization with the serial clock.

SIO1 is set with an 8-bit memory manipulation instruction.

When the value in bit 7 (CSIE1) of serial operating mode register 1 (CSIM1) is 1, writing data to SIO1 starts serial operation.

In transmission, data written to SIO1 is output to the serial output (SO1). In reception, data is read from the serial input (SI1) to SIO1.

Reset input makes SIO1 undefined.

(2) Serial clock counter

This counter counts the serial clocks to be output and input during transmission/reception to check whether 8-bit data has been transmitted/received.

10.3 Serial Interface Channel 1 Control Registers

The following two types of registers are used to control serial interface channel 1.

- Timer clock select register 3 (TCL3)
- Serial operating mode register 1 (CSIM1)

(1) Timer clock select register 3 (TCL3)

This register sets the serial clock of serial interface channel 1.

TCL3 is set with an 8-bit memory manipulation instruction.

Reset input sets TCL3 to 88H.

Remark Besides setting the serial clock of serial interface channel 1, TCL3 sets the serial clock of serial interface channel 0.

Figure 10-2. Timer Clock Select Register 3 (TCL3) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCL3	TCL37	TCL36	TCL35	TCL34	1	0	0	0	FF43H	88H	R/W

TCL37	TCL36	TCL35	TCL34	Serial Interface Channel 1 Serial Clock Selection		
					MCS = 1	MCS = 0
0	1	1	0	$f_{xx}/2$	Setting prohibited	$f_x/2^2$ (1.13 MHz)
0	1	1	1	$f_{xx}/2^2$	$f_x/2^2$ (1.13 MHz)	$f_x/2^3$ (563 kHz)
1	0	0	0	$f_{xx}/2^3$	$f_x/2^3$ (563 kHz)	$f_x/2^4$ (281 kHz)
1	0	0	1	$f_{xx}/2^4$	$f_x/2^4$ (281 kHz)	$f_x/2^5$ (141 kHz)
1	0	1	0	$f_{xx}/2^5$	$f_x/2^5$ (141 kHz)	$f_x/2^6$ (70.3 kHz)
1	0	1	1	$f_{xx}/2^6$	$f_x/2^6$ (70.3 kHz)	$f_x/2^7$ (35.2 kHz)
1	1	0	0	$f_{xx}/2^7$	$f_x/2^7$ (35.2 kHz)	$f_x/2^8$ (17.6 kHz)
1	1	0	1	$f_{xx}/2^8$	$f_x/2^8$ (17.6 kHz)	$f_x/2^9$ (8.8 kHz)
Other than above				Setting prohibited		

Caution When rewriting other data to TCL3, stop the serial transfer operation beforehand.

- Remarks**
1. f_{xx} : System clock frequency (f_x or $f_x/2$)
 2. f_x : System clock oscillation frequency
 3. MCS : Oscillation mode selection register (OSMS) bit 0
 4. Figures in parentheses apply to operation with $f_x = 4.5$ MHz

(2) Serial operating mode register 1 (CSIM1)

This register sets serial interface channel 1 serial clock and operation enable/stop.

CSIM1 is set with a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM1 to 01H.

Figure 10-3. Serial Operating Mode Register 1 (CSIM1) Format

Symbol	⑦	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM1	CSIE1	0	0	0	0	0	CSIM11	1	FF68H	01H	R/W

CSIM11	Serial Interface Channel 1 Clock Selection
0	Clock externally input to SCK1 pin
1	Clock specified with bits 4 to 7 of timer clock select register 3 (TCL3)

CSIE 1	CSIM 11	PM20	P20	PM21	P21	PM22	P22	Shift Register 1 Operation	Serial Clock Counter Operation Control	SI1/P20 Pin Function	SO1/P21 Pin Function	SCK1/P22 Pin Function
0	×	Note 1 ×	Note 1 ×	Note 1 ×	Note 1 ×	Note 1 ×	Note 1 ×	Operation stop	Clear	P20 (CMOS I/O)	P21 (CMOS I/O)	P22 (CMOS I/O)
1	0	Note 2 1	Note 2 ×	0	0	1	×	Operation enable	Count operation	SI1 ^{Note 2} (input)	SO1 (CMOS output)	SCK1 (Input)
	1					0	1					SCK1 (CMOS output)

Notes 1. Can be used freely as port function.

2. P20 (CMOS input/output) when only transmitter is used.

Remark × : Don't care

PMxx: Port mode register

Pxx : Output latch of port

10.4 Serial Interface Channel 1 Operations

The following two operating modes are available to the serial interface channel 1.

- Operation stop mode
- 3-wire serial I/O mode

10.4.1 Operation stop mode

Serial transfer is not carried out in the operation stop mode. Thus, power consumption can be reduced. The serial I/O shift register 1 (SIO1) does not carry out shift operation either, and thus it can be used as an ordinary 8-bit register.

In the operation stop mode, the P20/SI1, P21/SO1, and P22/ $\overline{\text{SCK1}}$ pins can be used as ordinary input/output ports.

(1) Register setting

The operation stop mode is set with the serial operating mode register 1 (CSIM1).

CSIM1 is set with a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM1 to 01H.

Symbol	⑦	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM1	CSIE1	0	0	0	0	0	CSIM11	1	FF68H	01H	R/W

CSIE1	CSIM11	PM20	P20	PM21	P21	PM22	P22	Shift Register 1 Operation	Serial Clock Counter Operation Control	SI1/P20 Pin Function	SO1/P21 Pin Function	$\overline{\text{SCK1}}$ /P22 Pin Function
0	×	Note 1	Note 1	Note 1	Note 1	Note 1	Note 1	Operation stop	Clear	P20 (CMOS I/O)	P21 (CMOS I/O)	P22 (CMOS I/O)
1	0	Note 2	Note 2	0	0	1	×	Operation enable	Count operation	SI1 ^{Note 2} (Input)	SO1 (CMOS output)	$\overline{\text{SCK1}}$ (Input)
	1					0	1					$\overline{\text{SCK1}}$ (CMOS output)

Notes 1. Can be used freely as port function.

2. P20 (CMOS input/output) when only transmitter is used.

Remark × : Don't care

PM×× : Port mode register

P×× : Output latch of port

10.4.2 3-wire serial I/O mode operation

The 3-wire serial I/O mode is valid for connection of peripheral I/O units and display controllers which incorporate a conventional synchronous serial interface such as the 75X/XL, 78K and 17K series.

Communication is carried out with three lines of serial clock ($\overline{\text{SCK1}}$), serial output (SO1) and serial input (SI1).

(1) Register setting

The 3-wire serial I/O mode is set with the serial operating mode register 1 (CSIM1).

CSIM1 is set with a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM1 to 01H.

Symbol	⑦	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM1	CSIE1	0	0	0	0	0	CSIM11	1	FF68H	01H	R/W

CSIM11	Serial Interface Channel 1 Clock Selection
0	Clock externally input to SCK1 pin
1	Clock specified with bits 4 to 7 of timer clock select register 3 (TCL3)

CSIE	CSIM	PM20	P20	PM21	P21	PM22	P22	Shift Register 1 Operation	Serial Clock Counter Operation Control	SI1/P20 Pin Function	SO1/P21 Pin Function	$\overline{\text{SCK1}}$ /P22 Pin Function
1	11											
0	×	Note 1	Note 1	Note 1	Note 1	Note 1	Note 1	Operation stop	Clear	P20 (CMOS I/O)	P21 (CMOS I/O)	P22 (CMOS I/O)
1	0	Note 2	Note 2			1	×	Operation enable	Count operation	SI1 ^{Note 2} (Input)	SO1 (CMOS output)	$\overline{\text{SCK1}}$ (Input)
	1	1	×	0	0	0	1					$\overline{\text{SCK1}}$ (CMOS output)

Notes 1. Can be used freely as port function.

2. P20 (CMOS input/output) when only transmitter is used (set bit 7 (RE) of the automatic data transmit/receive control register (ADTC) to 0).

Remark × : Don't care
 PM×× : Port mode register
 P×× : Output latch of port

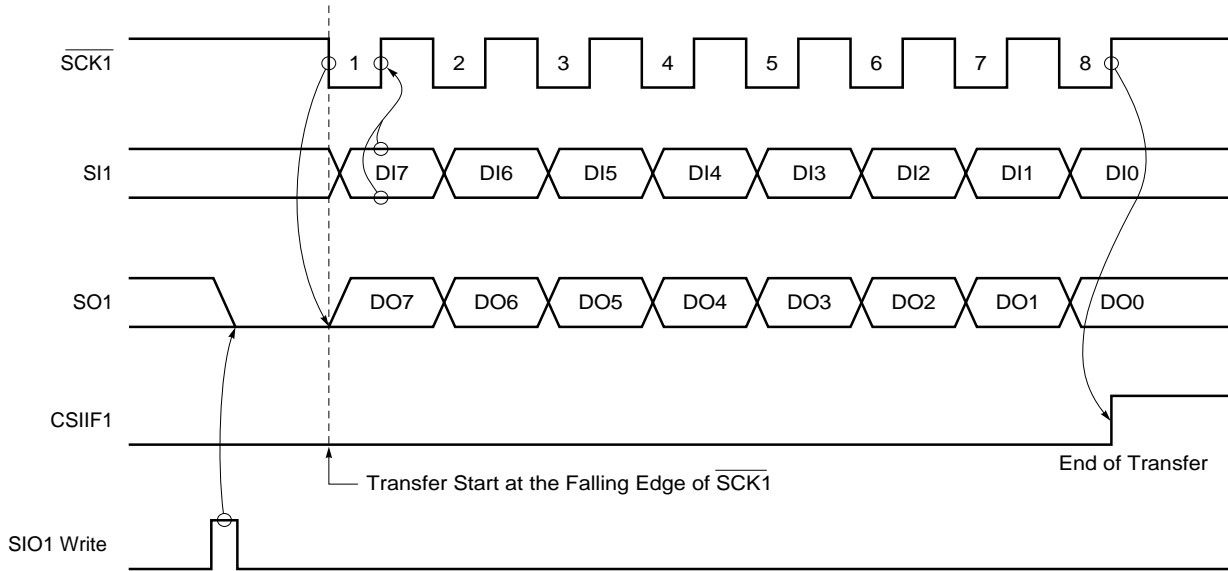
(2) Communication operation

The 3-wire serial I/O mode is used for data transmission/reception in 8-bit units. The start bit of data is the MSB. Bit-wise data transmission/reception is carried out in synchronization with the serial clock.

Shift operation of the serial I/O shift register 1 (SIO1) is carried out at the falling edge of the serial clock ($\overline{SCK1}$). The transmit data is held in the SO1 latch and is output from the SO1 pin. The receive data input to the SI1 pin is latched into SIO1 at the rising edge of $\overline{SCK1}$.

Upon termination of 8-bit transfer, the SIO1 operation stops automatically and the interrupt request flag (CSIF1) is set.

Figure 10-4. 3-Wire Serial I/O Mode Timings



Caution SO1 pin becomes low level by SIO1 write.

(3) Transfer start

Serial transfer is started by setting transfer data to the serial I/O shift register 1 (SIO1) when the following two conditions are satisfied.

- Serial interface channel 1 operation control bit (CSIE1) = 1
- Internal serial clock is stopped or $\overline{SCK1}$ is a high level after 8-bit serial transfer.

Caution If CSIE1 is set to "1" after data write to SIO1, transfer does not start.

Upon termination of 8-bit transfer, serial transfer automatically stops and the interrupt request flag (CSIF1) is set.

CHAPTER 11 INTERRUPT AND TEST FUNCTIONS

11.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Maskable interrupts

These interrupts undergo mask control. Maskable interrupts can be divided into a high interrupt priority group and a low interrupt priority group by setting the priority specification flag register (PR0L, PR0L).

Multiple high priority interrupts can be applied to low priority interrupts. If two or more interrupts with the same priority are simultaneously generated, each interrupts has a predetermined priority (refer to **Table 11-1**). A standby release signal is generated.

Two external interrupt sources and five internal interrupt sources are incorporated as maskable interrupts.

(2) Software interrupt

This is a vectored interrupt to be generated by executing the BRK instruction. It is acknowledged even in a disabled state. The software interrupt does not undergo interrupt priority control.

11.2 Interrupt Sources and Configuration

A total of eight maskable and software interrupt sources are incorporated in the interrupt sources (refer to **Table 11-1**).

Table 11-1. Interrupt Source List

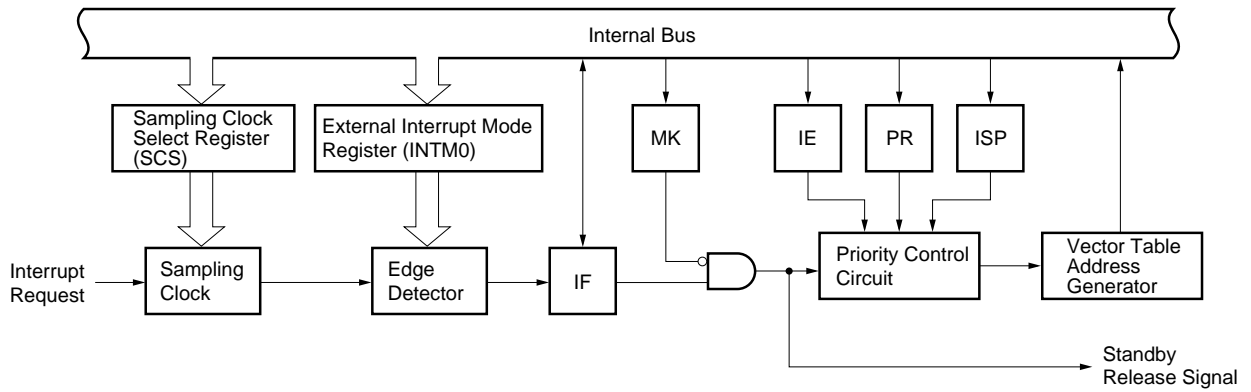
Interrupt Type	Note 1 Default Priority	Interrupt Source		Internal/ External	Vector Table Address	Note 2 Basic Configuration Type
	Name	Trigger				
Maskable	0	INTP0	Pin input edge detection	External	0006H	(A)
	1	INTP1			0008H	(B)
	2	INTCSI1	End of serial interface channel 1 transfer	Internal	0016H	(C)
	3	INTTMC	Generation of basic timer match signal		001CH	
	4	INTTM1	Generation of 8-bit timer/event counter 1 match signal			
	5	INTTM2	Generation of 8 bit timer/event counter 2 match signal			
	6	INTAD	End of A/D converter conversion		0020H	
Software	–	BRK	BRK instruction execution	Internal	003EH	(D)

Notes 1. Default priorities are intended for two or more simultaneously generated maskable interrupt requests. 0 is the highest priority and 6 is the lowest priority.

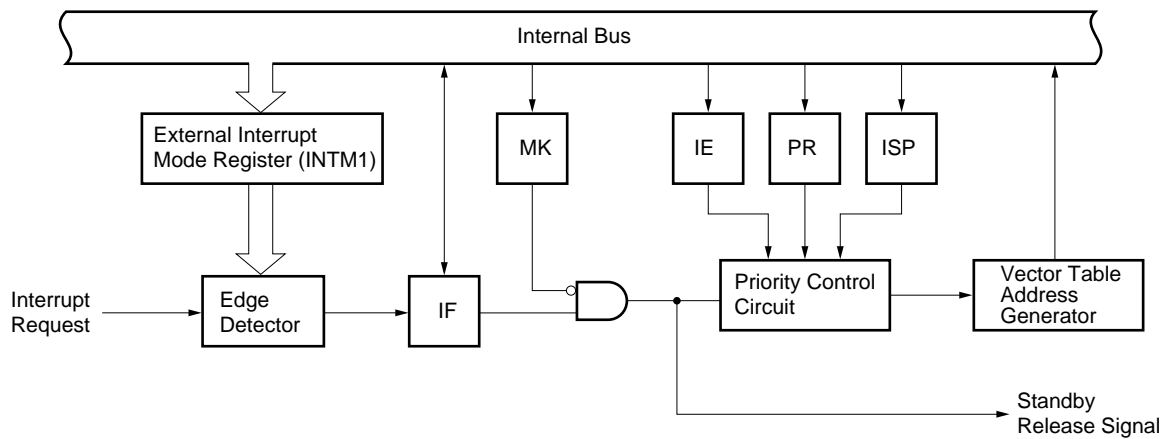
2. Basic configuration types (A) to (D) correspond to (A) to (D) of Figure 11-1.

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

(A) External maskable interrupt (INTP0)



(B) External maskable interrupt (INTP1)



(C) Internal maskable interrupt

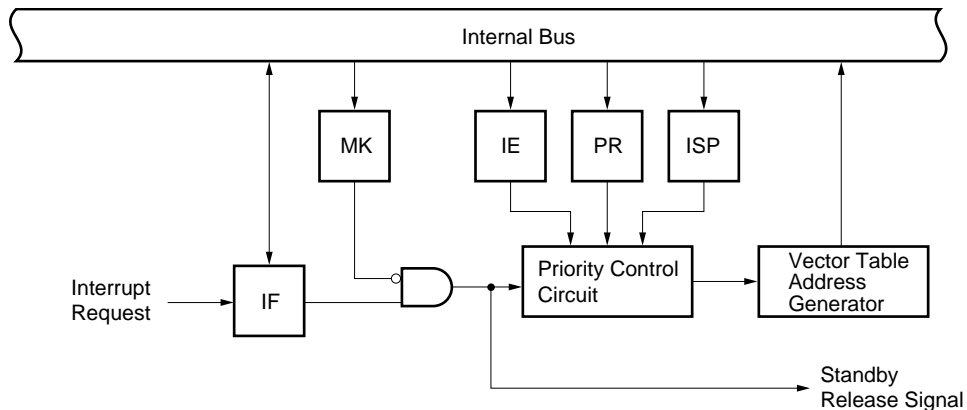
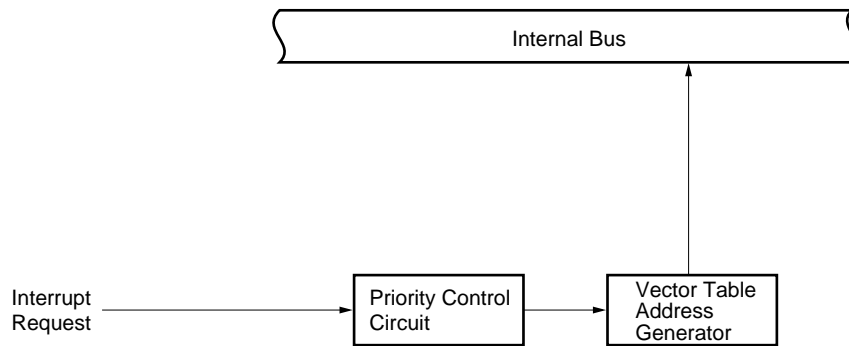


Figure 11-1. Basic Configuration of Interrupt Function (2/2)

(D) Software interrupt



Remark

IF	:	Interrupt request flag
IE	:	Interrupt enable flag
ISP	:	Inservice priority flag
MK	:	Interrupt mask flag
PR	:	Priority specification flag

11.3 Interrupt Function Control Registers

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

- Interrupt request flag register (IF0L, IF0H)
- Interrupt mask flag register (MK0L, MK0H)
- Priority specification flag register (PR0L, PR0H)
- External interrupt mode register (INTM0)
- Sampling clock select register (SCS)
- Program status word (PSW)

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

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

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specification Flag	
		Register		Register		Register
INTP0	PIF0	IF0L	PMK0	MK0L	PPR0	PR0L
INTP1	PIF1		PMK1		PPR1	
INTTM1	TMIF1	IF0H	TMMK1	MK0H	TMPR1	PR0H
INTTM2	TMIF2		TMMK2		TMPR2	
INTCSI1	CSIIF1		CSIMK1		CSIPR1	
INTAD	ADIF		ADMK		ADPR	
INTTMC	TMCIF		TMCMK		TMCPR	

(1) Interrupt request flag registers (IF0L and IF0H)

The interrupt request flag is set to 1 when the corresponding interrupt request is generated or an instruction is executed. It is cleared to 0 when an instruction is executed upon acknowledgment of an interrupt request or upon application of reset input.

IF0L and IF0H are set with a 1-bit or 8-bit memory manipulation instruction. If IF0L and IF0H are used as a 16-bit register IF0 use a 16-bit memory manipulation instruction for the setting.

Reset input sets these registers to 00H.

Figure 11-2. Interrupt Request Flag Register (IF0L, IF0H) Format

Symbol	7	6	5	4	3	②	①	0	Address	After Reset	R/W
IF0L	0	0	0	0	0	PIF1	PIF0	0	FFE0H	00H	R/W
IF0H	7	⑥	⑤	④	3	②	①	0	FFE1H	00H	R/W
	0	ADIF	TMIF2	TMIF1	0	TMCIF	CSIIF1	0			

× × IF ×	Interrupt Request Flag
0	No interrupt request signal
1	Interrupt request signal is generated; Interrupt request state

Caution Be sure to set bits 0, 3 through 7 of IF0L and bits 0, 3, and 7 of IF0H to 0.

(2) Interrupt mask flag registers (MK0L and MK0H)

The interrupt mask flag is used to enable/disable the corresponding maskable interrupt service and to set standby clear enable/disable.

MK0L and MK0H are set with a 1-bit or 8-bit memory manipulation instruction. If MK0L and MK0H are used as a 16-bit register MK0, use a 16-bit memory manipulation instruction for the setting.

Reset input sets these registers to FFH.

Figure 11-3. Interrupt Mask Flag Register (MK0L, MK0H) Format

Symbol	7	6	5	4	3	②	①	0	Address	After Reset	R/W
MK0L	1	1	1	1	1	PMK	PMK	1	FFE4H	FFH	R/W
MK0H	7	⑥	⑤	④	3	②	①	0	FFE5H	FFH	R/W
	1	ADMK	TMMK2	TMMK1	1	TMCMK	CSIMK1	1			

× × MK ×	Interrupt Service Control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Cautions 1. Because P00 and P01 of port 0 have dual functions as the 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, 1 should be set in the interrupt mask flag before using the output mode.

2. Be sure to set bits 0, 3 through 7 of MK0L and bits 0, 3, and 7 of MK0H to 1.

(3) Priority specification flag registers (PR0L and PR0H)

The priority specification flag is used to set the corresponding maskable interrupt priority orders.

PR0L and PR0H are set with a 1-bit or 8-bit memory manipulation instruction. If PR0L and PR0H are used as a 16-bit register PR0, use a 16-bit memory manipulation instruction for the setting.

Reset input sets these registers to FFH.

Figure 11-4. Priority Specification Flag Register (PR0L, PR0H) Format



Caution Be sure to set bits 0, 3 through 7 of PR0L and bits 0, 3, and 7 of PR0H to 1.

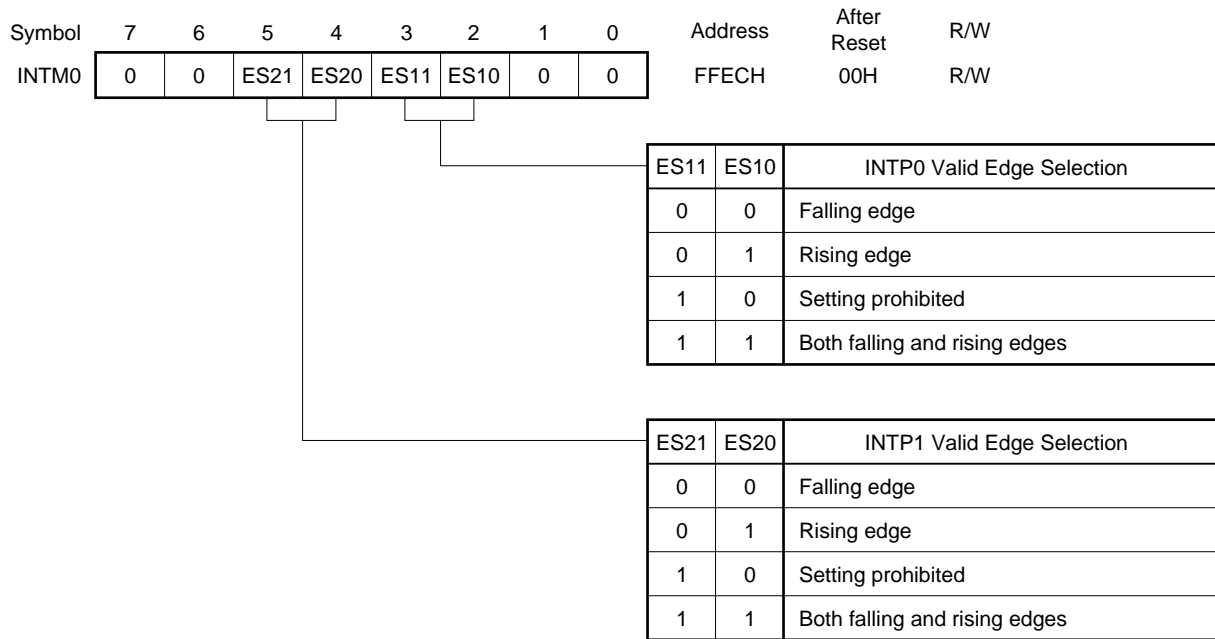
(4) External interrupt mode register 0 (INTM0)

These registers set the valid edge for INTP0 and INTP1.

INTM0 is set by 8-bit memory manipulation instruction.

Reset input sets this register to 00H.

Figure 11-5. External Interrupt Mode Register 0 (INTM0) Format



(5) Sampling clock select register (SCS)

This register is used to set the valid edge clock sampling clock to be input to INTP0. When remote controlled data reception is carried out using INTP0, digital noise is removed with sampling clocks.

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

Reset input sets SCS to 00H.

Figure 11-6. Sampling Clock Select Register (SCS) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SCS	0	0	0	0	0	0	SCS1	SCS0	FF47H	00H	R/W

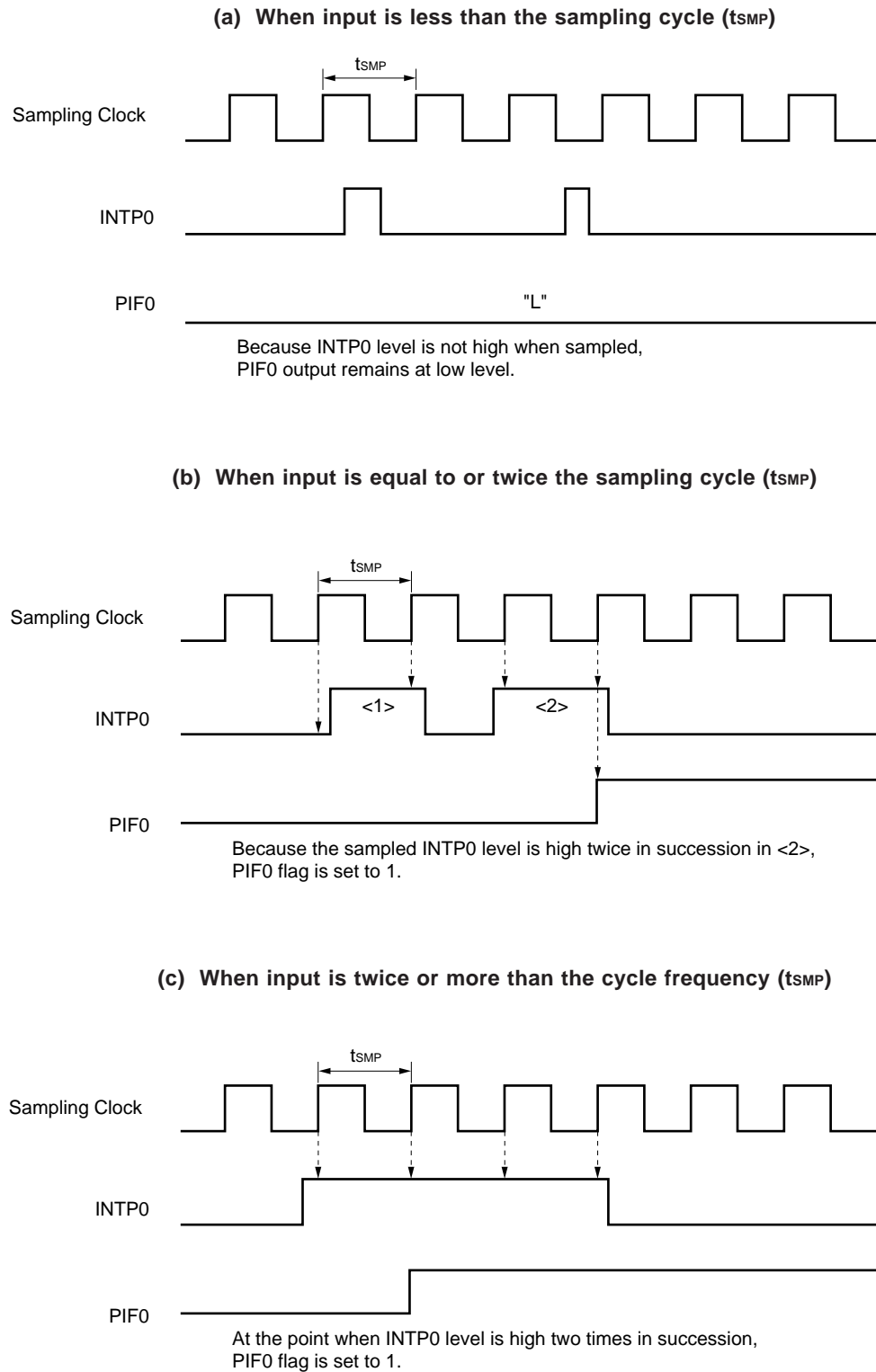
SCS1	SCS0	INTP0 Sampling Clock Selection		
			MCS = 1	MCS = 0
0	0	$f_{xx}/2^N$		
0	1	$f_{xx}/2^7$	$f_x/2^7$ (35.2 kHz)	$f_x/2^8$ (17.6 kHz)
1	0	$f_{xx}/2^5$	$f_x/2^5$ (140.6 kHz)	$f_x/2^6$ (70.3 kHz)
1	1	$f_{xx}/2^6$	$f_x/2^6$ (70.3 kHz)	$f_x/2^7$ (35.2 kHz)

Caution $f_{xx}/2^N$ is a clock to be supplied to the CPU and $f_{xx}/2^5$, $f_{xx}/2^6$ and $f_{xx}/2^7$ are clocks to be supplied to the peripheral hardware. $f_{xx}/2^N$ stops in the HALT mode.

- Remarks**
1. N : Value (N=0 to 4) at bits 0 to 2 (PCC0 to PCC2) of processor clock control register (PCC)
 2. f_{xx} : System clock frequency (f_x or $f_x/2$)
 3. f_x : System clock oscillation frequency
 4. MCS : Oscillation mode selection register (OSMS) bit 0
 5. Values in parentheses when operated with $f_x = 4.5$ MHz.

When the sampled INTP0 input level is active twice in succession, the noise eliminator sets interrupt request flag (PIF0) to 1.

Figure 11-7. Noise Eliminator Input/Output Timing (during rising edge detection)



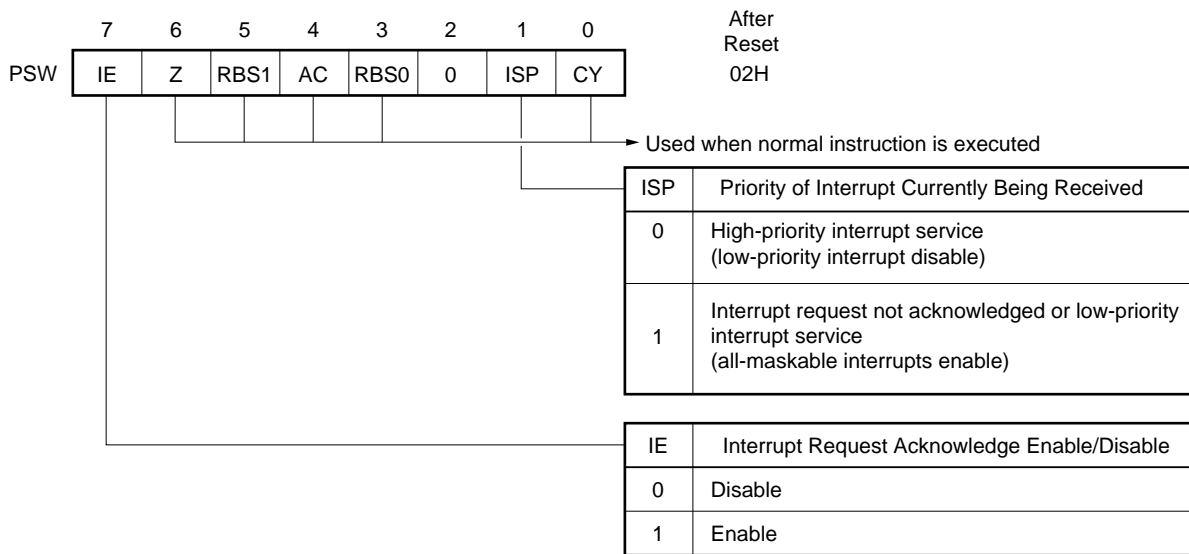
(6) Program status word (PSW)

The program status word is a register to hold the instruction execution result and the current status for interrupt request. The IE flag to set maskable interrupt enable/disable and the ISP flag to control nesting interrupt are mapped.

Besides 8-bit unit read/write, this register can carry out operations with a bit manipulation instruction and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, if 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 contents of the priority specification flag of the acknowledged interrupt are transferred to the ISP flag. The acknowledged interrupt is also saved into the stack with the PUSH PSW instruction. It is restored from the stack with the RETI, RETB, and POP PSW instructions.

Reset input sets PSW to 02H.

Figure 11-8. Program Status Word (PSW) Configuration



11.4 Interrupt Service Operations

11.4.1 Maskable interrupt request acknowledge operation

A maskable interrupt request becomes acknowledgeable when an interrupt request flag is set to 1 and the mask (MK) flag of the interrupt request is cleared to 0. A vectored interrupt request is acknowledged in an interrupt enable state (with IE flag set to 1). However, a low-priority interrupt request is not acknowledged during high-priority interrupt request service (with ISP flag reset to 0).

Wait times maskable interrupt request generation to interrupt request service are as follows.

Table 11-3. Times from Maskable Interrupt Request Generation to Interrupt Service

	Minimum Time	Maximum Time ^{Note}
When $\times\times PR\times = 0$	7 clocks	32 clocks
When $\times\times PR\times = 1$	8 clocks	33 clocks

Note If an interrupt request is generated just before a divide instruction, the wait time is maximized.

Remark 1 clock : $\frac{1}{f_{CPU}}$ (f_{CPU} : CPU clock)

If two or more maskable interrupt requests are generated simultaneously, the request specified for higher priority with the priority specification flag is acknowledged first. Two or more requests specified for the same priority by the priority specification flag, the default priorities apply.

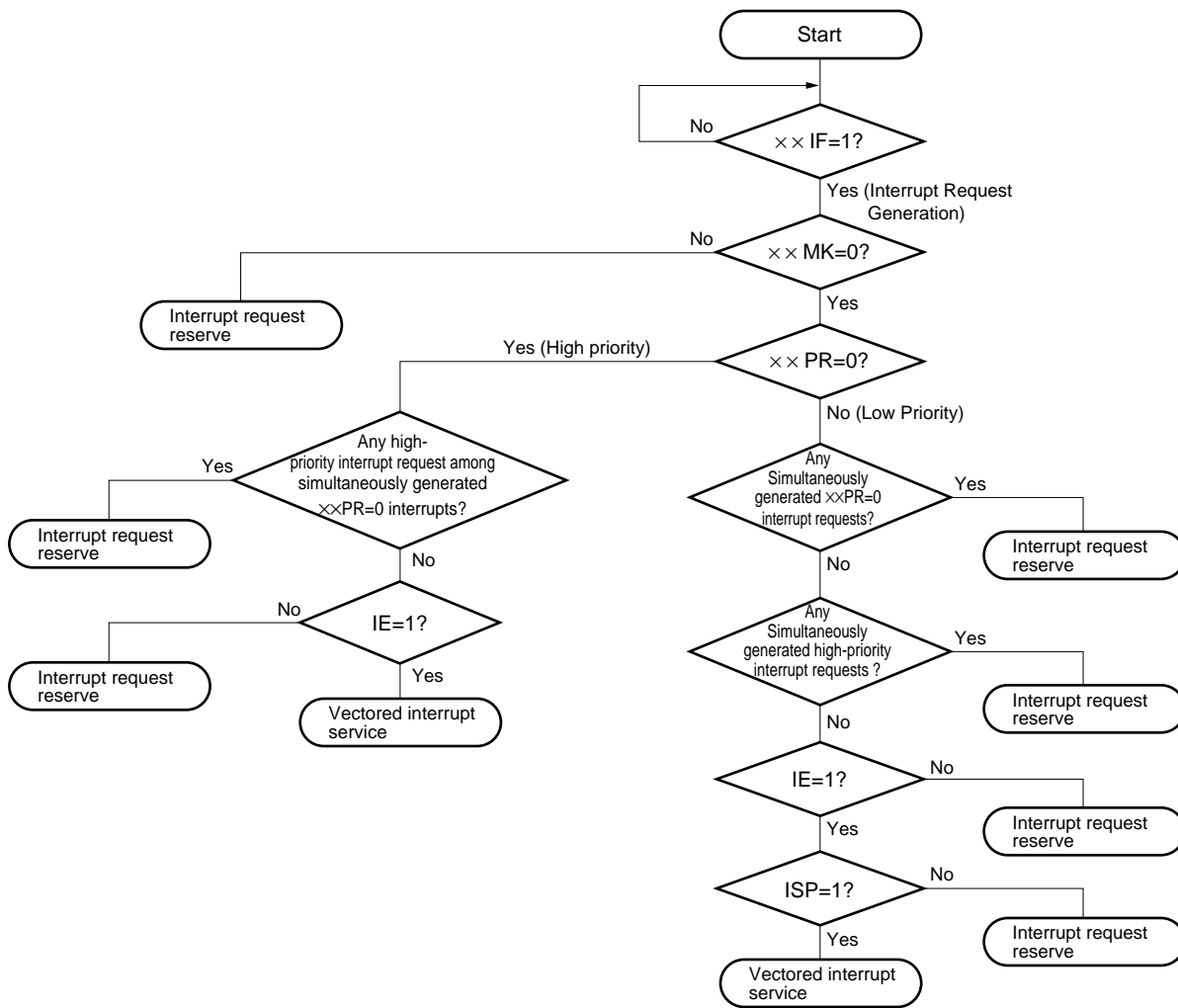
Any reserved interrupt requests are acknowledged when they become acknowledgeable.

Figure 11-9 shows interrupt request acknowledge algorithms.

If a maskable interrupt request is acknowledged, the acknowledged interrupt request is saved in the stacks, the program status word (PSW) and the program counter (PC), in that order, the IE flag is reset to 0, and the acknowledged interrupt priority specification flag contents are transferred to the ISP flag. Further, the vector table data determined for each interrupt request is loaded into PC and branched.

Return from the interrupt is possible with the RETI instruction.

Figure 11-9. Interrupt Request Acknowledge Processing Algorithm



xxIF : Interrupt request flag

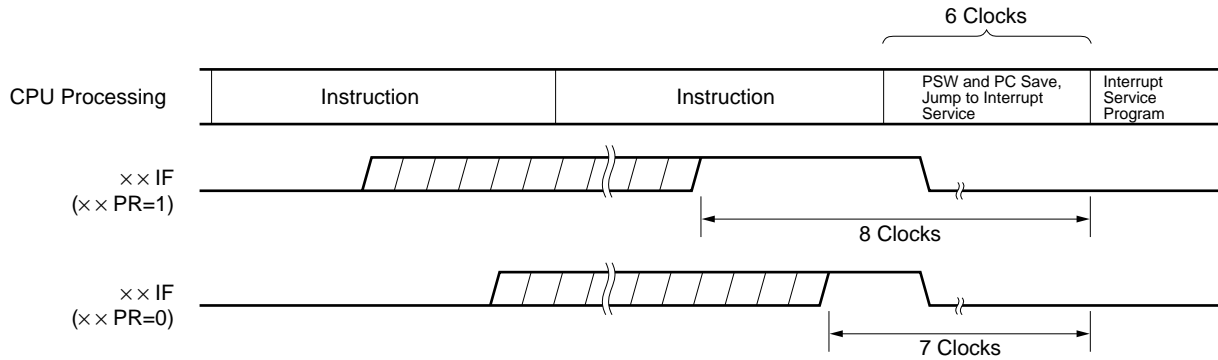
xxMK : Interrupt mask flag

xxPR : Priority specification flag

IE : Flag controlling acknowledging maskable interrupt request (1 = enable, 0 = disable)

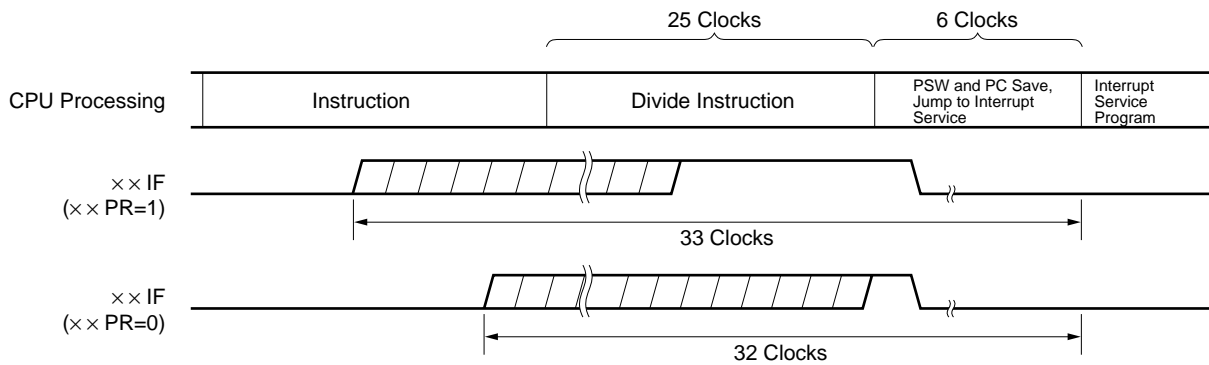
ISP : Flag indicating priority of interrupt currently serviced (0 = interrupt with high priority serviced, 1 = interrupt request is not acknowledged, or interrupt with low priority serviced)

Figure 11-10. Interrupt Request Acknowledge Timing (Minimum Time)



Remark 1 clock: $\frac{1}{f_{\text{CPU}}}$ (f_{CPU} : CPU clock)

Figure 11-11. Interrupt Request Acknowledge Timing (Maximum Time)



Remark 1 clock: $\frac{1}{f_{\text{CPU}}}$ (f_{CPU} : CPU clock)

11.4.2 Software interrupt request acknowledge operation

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

If a software interrupt request is acknowledged, it is saved in the stacks, PSW and PC, in that order, the IE flag is reset to 0 and the contents of the vector tables (003EH and 003FH) are loaded into PC and branched.

Return from the software interrupt is possible with the RETB instruction.

Caution Do not use the RETI instruction for returning from the software interrupt.

11.4.3 Nesting

Accepting another interrupt request while an interrupt is being serviced is called nesting.

Nesting does not take place unless the interrupts are enabled to be acknowledged (IE = 1). Accepting another interrupt request is disabled (IE = 0) when one interrupt has been acknowledged. Therefore, to enable nesting, the EI flag must be set to 1 during interrupt servicing, to enable the another interrupt.

Nesting may not occur even when the interrupts are enabled. This is controlled by the priorities of the interrupts. Although two types of priorities, default priority and programmable priority, may be assigned to an interrupt, nesting is controlled by using the programmable priority.

If an interrupt with the same level of priority as or the higher priority than the interrupt currently serviced occurs, that interrupt can be acknowledged and nested. If an interrupt with a priority lower than that of the currently serviced interrupt occurs, that interrupt cannot be acknowledged and nested.

An interrupt that is not acknowledged and nested because it is disabled or it has a low priority is kept pending. This interrupt is acknowledged after servicing of the current interrupt has been completed and one instruction of the main routine has been executed.

Table 11-4 shows the interrupts that can be nested, and Figure 11-12 shows an example of nesting.

Table 11-4. Interrupt Request Enabled for Nesting during Interrupt Service

Nesting Interrupt Request Interrupt being serviced		Maskable Interrupt Request			
		xxPR = 0		xxPR = 1	
		IE=1	IE=0	IE=1	IE=0
Maskable interrupt	ISP = 0	E	D	D	D
	ISP = 1	E	D	E	D
Software interrupt servicing		E	D	E	D

Remarks 1. E : Nesting enabled

2. D : Nesting disabled

3. ISP and IE are the flags contained in PSW

ISP = 0 : An interrupt with higher priority is being serviced

ISP = 1 : An interrupt request is not accepted or an interrupt with lower priority is being serviced

IE = 0 : Interrupt request acknowledge is disabled

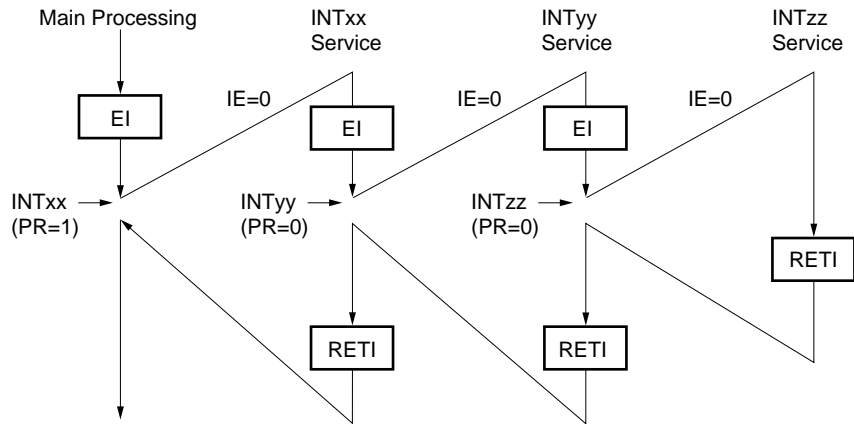
IE = 1 : Interrupt request acknowledge is enabled

4. xxPR is a flag contained in PR0L and PR0R.

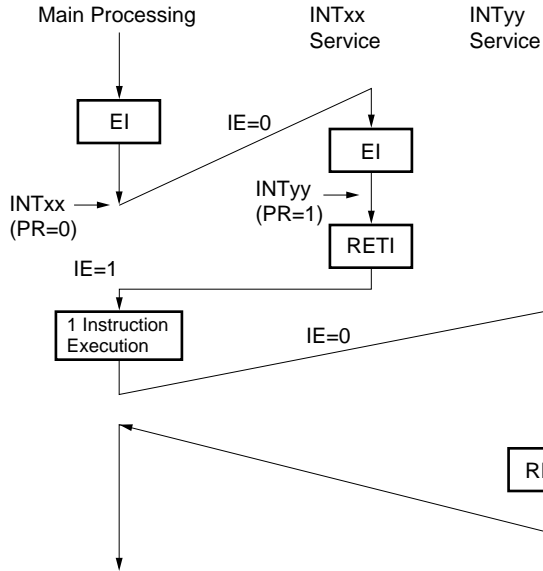
xxPR = 0 : Higher priority level

xxPR = 1 : Lower priority level

Figure 11-12. Nesting Example (1/2)

Example 1. Example where nesting takes place two times

Two interrupt requests, INTyy and INTzz, are acknowledged while interrupt INTxx is serviced, and nesting takes place. Before each interrupt request is acknowledged, the EI instruction is always executed, and the interrupt is enabled.

Example 2. Example where nesting does not take place because of priority control

Interrupt request INTyy that is generated while interrupt INTxx is being serviced is not acknowledged because its priority is lower than that of INTxx, and therefore, nesting does not take place. INTyy request is kept pending, and is acknowledged after one instruction of the main routine has been executed.

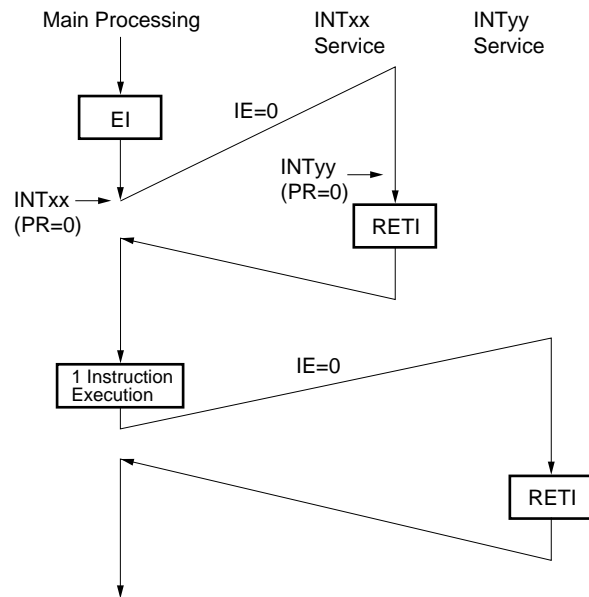
PR = 0 : High-priority level

PR = 1 : Low-priority level

IE = 0 : Acknowledging interrupt request is disabled.

Figure 11-12. Nesting Example (2/2)

Example 3. Example where nesting does not take place because interrupts are not enabled



Because interrupts are not enabled (EI instruction is not issued) in interrupt processing INTxx, interrupt request INTyy is not acknowledged, and nesting does not take place. INTyy request is kept pending, and is acknowledged after one instruction of the main routine has been executed.

PR = 0 : High priority level

IE = 0 : Acknowledging interrupts is disabled.

11.4.4 Pending interrupt requests

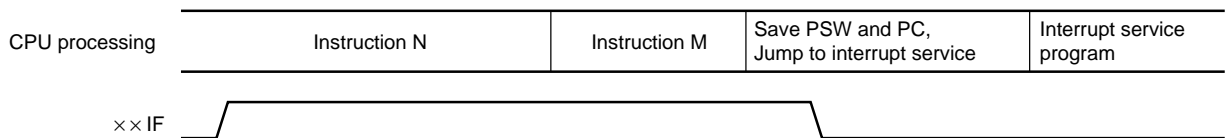
Even if an interrupt request is generated, the following instructions keep it pending.

- 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
- Instructions manipulating IF0L, IF0H, MK0L, MK0H, PR0L, PR0H, and INTM0 registers

Caution The BRK instruction is not one of the above instructions that keep an interrupt request pending. However, the software interrupt that is started by execution of the BRK instruction clears the IE flag to 0. Therefore, even if a maskable interrupt request is generated while the BRK instruction is being executed, it is not accepted.

Figure 11-13 shows the timing at which an interrupt request is accepted.

Figure 11-13. Pending Interrupt Request



- Remarks**
1. Instruction N: Instruction that keeps interrupt request pending
 2. Instruction M: Instruction that does not keep interrupt request pending
 3. Operation of $\times\times IF$ (interrupt request) is not affected by value of $\times\times PR$ (priority level).

11.5 Test Functions

The test function sets the corresponding test input flag and generates a standby release signal when a falling edge is detected at port 4.

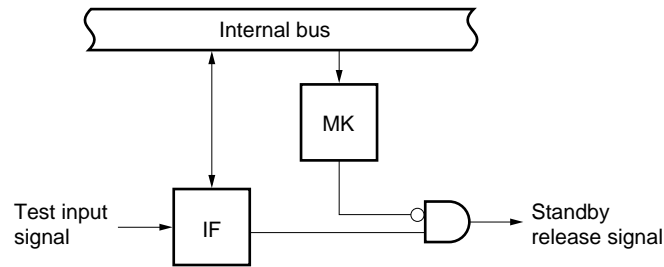
Unlike the interrupt function, this function does not perform vector processing.

There is one test input factor as shown in Table 11-5. The basic configuration is shown in Figure 11-14.

Table 11-5. Test Input Factor

Test Input Factor		Internal/ External
Name	Trigger	
INTPT4	Falling edge detection at port 4	External

Figure 11-14. Basic Configuration of Test Function



Remark IF: test input flag
MK: test mask flag

11.5.1 Registers controlling the test function

The test function is controlled by the following register.

- Key return mode register (KRM)

The name of the test input flag and test mask flag corresponding to the test input signal are listed in Table 11-6.

Table 11-6. Flag Corresponding to Test Input Signal

Test Input Signal Name	Test Input Flag	Test Mask Flag
INTPT4	KRIF	KRMK

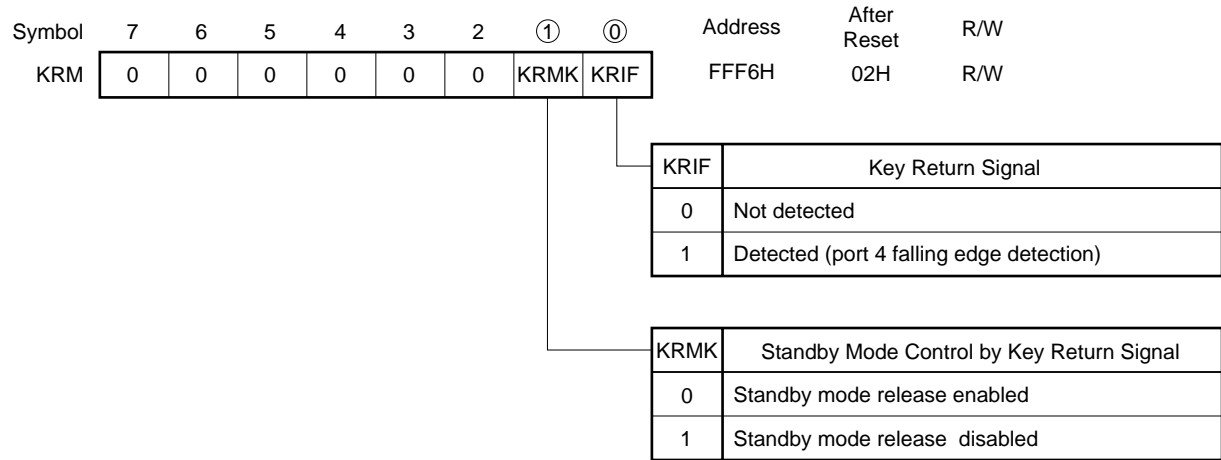
(1) Key return mode register (KRM)

This register is used to set enable/disable of standby function clear by key return signal (port 4 falling edge detection).

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

Reset input sets KRM to 02H.

Figure 11-15. Key Return Mode Register (KRM) Format



Caution When port 4 falling edge detection is used, clear KRIF to 0 in program (not cleared to 0 by hardware)

11.5.2 Test input signal acknowledge operation

- External test signal (INTPT4)**

When a falling edge is input to the port 4 (P40 to P47) pins, an external test input signal (INTPT4) is generated, setting the KRIF flag. At this time, the standby release signal is generated if it is not masked by the KRMK flag. By using port 4 as key matrix return signal input, whether or not a key input has been applied can be checked from the KRIF status.

[MEMO]

CHAPTER 12 PLL FREQUENCY SYNTHESIZER

12.1 Function of PLL Frequency Synthesizer

The PLL (Phase Locked Loop) frequency synthesizer is used to lock the frequency in the MF (Middle Frequency), HF (High Frequency), and VHF (Very High Frequency) ranges to a specific frequency by means of phase difference comparison.

The PLL frequency synthesizer divides the frequency of the signal input from the VCOL or VCOH pin by using a programmable divider, and outputs the phase difference between the frequency of this signal and reference frequency from the EO0 and EO1 pin.

The following two types of input pins and four frequency division modes are used.

(1) Direct division (MF) mode

The VCOL pin is used.

The VCOH pin goes into a high-impedance state.

(2) Pulse swallow (HF) mode

The VCOL pin is used.

The VCOH pin goes into a high-impedance state.

(3) Pulse swallow (VHF) mode

The VCOH pin is used.

The VCOL pin goes into a high-impedance state.

(4) VCOL, VCOH pin disable

The VCOL and VCOH pin go into a high-impedance state.

However, the phase comparator, reference frequency generator, and charge pump operate.

Therefore, the operation is different from the PLL frequency synthesizer disabled status which is described later.

These division modes are selected by using the PLL mode select register (PLLMD).

The division value (N value) is set to the programmable divider by using the PLL data register. Frequency division in each of the above modes is carried out according to the value (N value) set to the programmable divider.

Table 12-1 shows the division modes, input pins used (VCOL pin or VCOH pin), and the value that can be set to the programmable divider.

Table 12-1. Division Mode, Input Pin, and Division Value

Division Mode	Pin Used	Value That Can Be Set
Direct division (MF)	VCOL	32 to $2^{12}-1$
Pulse swallow (HF)	VCOL	1024 to $2^{17}-1$
Pulse swallow (VHF)	VCOH	1024 to $2^{17}-1$

Caution For the frequencies that can be actually input, and input amplitude, refer to Electrical Specifications in Data Sheet.

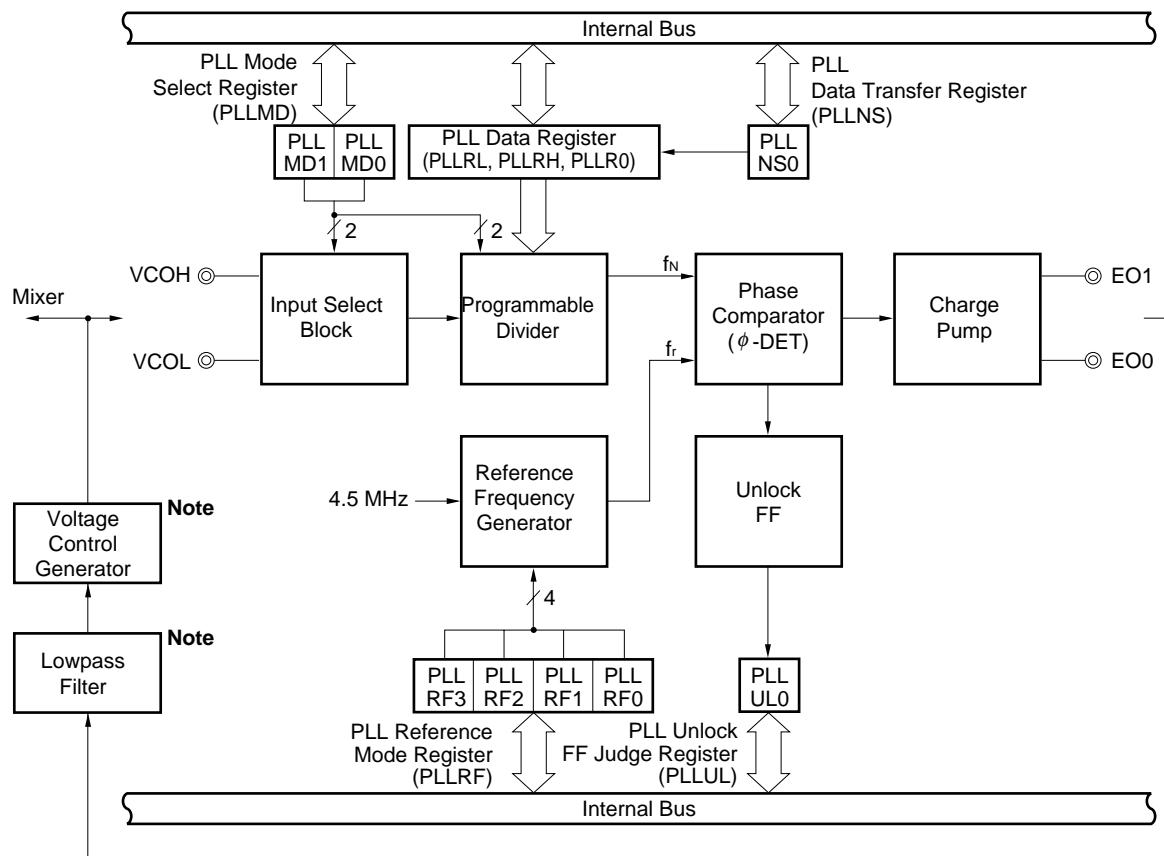
12.2 Configuration of PLL Frequency Synthesizer

The PLL frequency synthesizer consists of the following hardware units.

Table 12-2. PLL Frequency Synthesizer Configuration

Item	Configuration
Data register	PLL data register L (PLLRL) PLL data register H (PLLRH) PLL data register 0 (PLLR0)
Control register	PLL mode select register (PLLMD) PLL reference mode register (PLLRF) PLL unlock FF judge register (PLLUL) PLL data transfer register (PLLNS)

Figure 12-1. PLL Frequency Synthesizer Block Diagram



Note External circuit

(1) PLL data register L (PLLRL), PLL data register H (PLLRH), and PLL data register 0 (PLLRO)

These registers set the division value of the PLL frequency synthesizer. The division value of the PLL frequency synthesizer is made up of 17 bits. The high-order 16 bits of this value are set by the PLL data register L (PLLRL) and PLL data register H (PLLRH). The high-order 16 bits can also be set by the PLL data register (PLLR).

The least significant bit is set by bit 7 (PLLSCN) of the PLL data register 0 (PLLRO).

The contents of these registers are undefined at reset. These registers hold the current values in the STOP and HALT modes.

(2) Input select block

The input select block consists of the VCOL and VCOH pins, and input amplifiers of the respective pins.

(3) Programmable divider

The programmable divider consists of two modulus prescalers, a programmable counter (12 bits), a swallow counter (5 bits), and a division mode select switch.

(4) Reference frequency generator

The reference frequency generator consists of a divider that generates the reference frequency f_r of the PLL frequency synthesizer, and a multiplexer.

(5) Phase comparator

The phase comparator (ϕ -DET) compares the phase of the divided frequency output f_n of the programmable divider with that of the reference frequency output f_r of the reference frequency generator, and outputs an up request signal (\overline{UP}) and down request signal (\overline{DW}).

(6) Unlock FF

The unlock FF detects the unlock status of the PLL frequency synthesizer from the up request signal (\overline{UP}) and down request signal (\overline{DW}) of the phase comparator (ϕ -DET).

(7) Charge pump

The charge pump outputs the result of the output of the phase comparator from the error out pins (EO0 and EO1 pins).

12.3 Registers Controlling PLL Frequency Synthesizer

The PLL frequency synthesizer is controlled by the following four registers.

- PLL mode select register (PLLMD)
- PLL reference mode register (PLLRF)
- PLL unlock FF judge register (PLLUL)
- PLL data transfer register (PLLNS)

(1) PLL mode select register (PLLMD)

This register selects the input pin and division mode of the PLL frequency synthesizer.

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

The value of this register is set to 00H at reset and in the STOP mode.

In the HALT mode, it holds the value immediately before the HALT mode was set.

Figure 12-2. PLL Mode Select Register (PLLMD) Format

Symbol	7	6	5	4	3	2	①	②	Address	After Reset	R/W
PLLMD	0	0	0	0	0	0	PLLMD1	PLLMD0	FFA0H	00H	R/W

PLLMD1	PLLMD0	Selects Division Mode of PLL Frequency Synthesizer and VCO Input Pin
0	0	Disables VCOL and VCOH pins ^{Note}
0	1	Direct division (VCOL pin and MF mode)
1	0	Pulse swallow (VCOH pin and VHF mode)
1	1	Pulse swallow (VCOL pin and HF mode)

Note This does not mean that the PLL is disabled. The VCOL and VCOH pins go into a high-impedance state. The EO0 and EO1 pin go low.

Remark Bits 2 through 7 are fixed to 0 by hardware.

(2) PLL reference mode register (PLLRF)

This register selects the reference frequency f_r of the PLL frequency synthesizer and sets the disabled status of the PLL frequency synthesizer.

PLLRF is set by using 1-bit or 8-bit memory manipulation instruction.

The value of this register is set to 0FH at reset and in the STOP mode.

In the HALT mode, it holds the value immediately before the HALT mode was set.

Figure 12-3. PLL Reference Mode Register (PLLRF) Format

Symbol	7	6	5	4	③	②	①	①	Address	After Reset	R/W
PLLRF	0	0	0	0	PLLRF3	PLLRF2	PLLRF1	PLLRF0	FFA1H	0FH	R/W

PLLRF3	PLLRF2	PLLRF1	PLLRF0	Sets Reference Frequency f_r of PLL Frequency Synthesizer
0	0	0	0	Setting prohibited
0	0	0	1	Setting prohibited
0	0	1	0	5 kHz
0	0	1	1	10 kHz
0	1	0	0	Setting prohibited
0	1	0	1	Setting prohibited
0	1	1	0	25 kHz
0	1	1	1	50 kHz
1	0	0	0	3 kHz
1	0	0	1	9 kHz
1	0	1	0	Setting prohibited
1	0	1	1	PLL disable ^{Note}
1	1	0	0	1 kHz
1	1	0	1	Setting prohibited
1	1	1	0	Setting prohibited
1	1	1	1	PLL disable ^{Note}

Note When PLL disable is selected, the VCOL, VCOH, EO0, and EO1 pins go into a high-impedance state.

Remark Bits 4 through 7 are fixed to 0 by hardware.

(3) PLL unlock FF judge register (PLLUL)

This register detects whether the PLL frequency synthesizer is in the unlock status.

Because this register is an R&RESET register, it is reset to 0 after it has been read.

At reset, the value of this register is set to 0xH^{Note 1}.

In the STOP and HALT modes, this register holds the value immediately before the STOP or HALT mode was set.

Figure 12-4. PLL Unlock FF Judge Register (PLLUL) Format

Symbol	7	6	5	4	3	2	1	①	Address	After Reset	R/W
PLLUL	0	0	0	0	0	0	0	PLLUL0	FFA2H	0xH ^{Note 1}	R ^{Note 2}

PLLUL0	Detects Status of Unlock FF
0	Unlock FF = 0: PLL lock status
1	Unlock FF = 1: PLL unlock status

Notes 1. The value of bit 0 (PLLUL0) at reset differs depending on the type of reset that has been executed (refer to the table below).

2. Bit 0 (PLLUL0) is R&Reset.

		7	6	5	4	3	2	1	0
At reset	Power-ON clear	0	0	0	0	0	0	0	Undefined
	Watchdog timer								Retained
	RESET input								Retained
STOP mode									Retained
HALT mode		↓	↓	↓	↓	↓	↓	↓	Retained

Remark Bits 1 through 7 are fixed to 0 by hardware.

(4) PLL data transfer register (PLLNS)

This register transfers the values of the PLL data registers (PLLRL, PLLRH, and PLLR0) to the programmable counter and swallow counter.

The value of this register is 00H at reset and in the STOP mode.

In the HALT mode, this register holds the previous value immediately before the HALT mode is set.

Figure 12-5. PLL Data Transfer Register (PLLNS) Format

Symbol	7	6	5	4	3	2	1	①	Address	After Reset	R/W
PLLNS	0	0	0	0	0	0	0	PLLNS0	FFA3H	00H	W

PLLNS0	Transfers Value of PLL Data Register to Programmable Counter and Swallow Counter
0	Does not transfer
1	Transfers

Remark Bits 0 through 7 are fixed to 0 by hardware.

12.4 Operation of PLL Frequency Synthesizer

12.4.1 Operation of each block of PLL frequency synthesizer

(1) Operation of input select block and programmable divider

The input select block and programmable divider select the input pin and division mode of the PLL frequency synthesizer and divide the frequency in the selected division mode, according to the setting of the PLL mode select register (PLLMD).

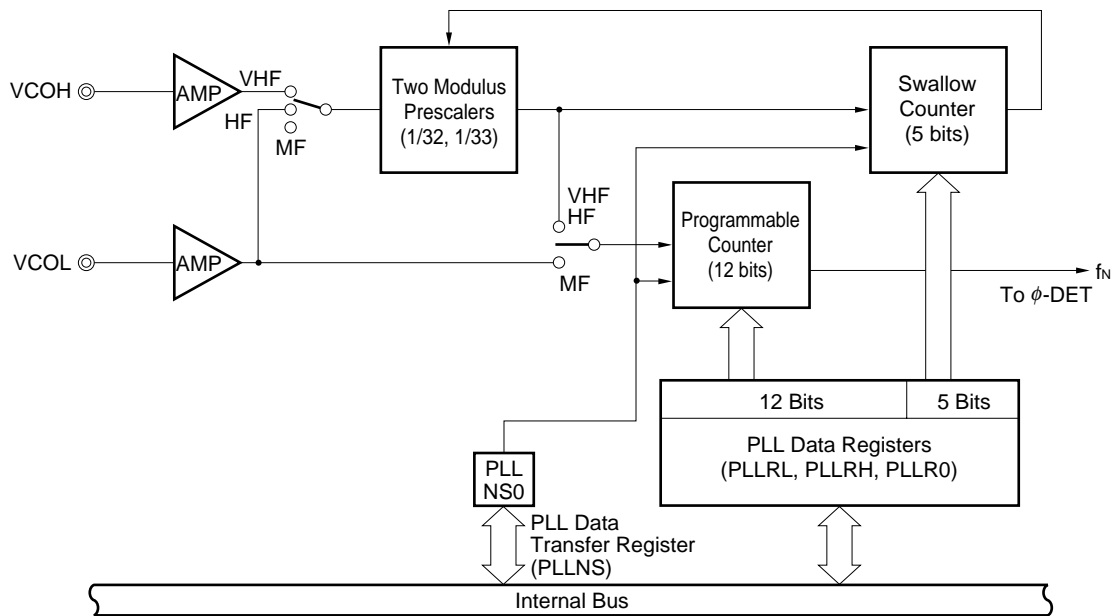
The programmable counter (12 bits) and pulse swallow counter (5 bits) are binary counters.

The division value (N value) is set to the programmable counter and swallow counter by the PLL data registers (PLLRL, PLLRH, and PLLR0).

When the N value has been transferred to the programmable counter and swallow counter, frequency division is performed in the selected division mode according to the status of bit 0 (PLLNS0) of the PLL data transfer register.

Figure 12-6 shows the configuration of the input select block and programmable divider.

Figure 12-6. Input Select Block and Programmable Divider Configuration



(2) Operation of reference frequency generator

The reference frequency generator divides the 4.5 MHz output of the crystal resonator and generates seven types of reference frequency f_r for the PLL frequency synthesizer.

Reference frequency f_r is selected by the PLL reference mode register (PLLRF).

Figure 12-7 shows the configuration of the reference frequency generator.

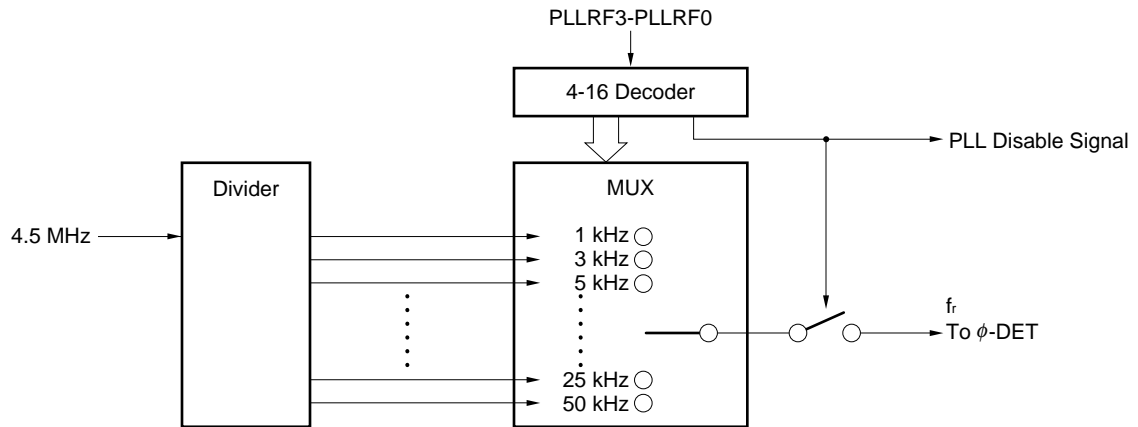
Figure 12-7. Reference Frequency Generator Configuration**(3) Operation of phase comparator (ϕ -DET)**

Figure 12-8 shows the configuration of the phase comparator (ϕ -DET), charge pump, and unlock FF.

The phase comparator (ϕ -DET) compares the phase of the divided frequency f_N of the programmable divider with that of the reference frequency f_r of the reference frequency generator, and outputs an up request signal, \overline{UP} , or a down request signal, \overline{DW} .

If the divided frequency f_N is lower than the reference frequency f_r , the up request signal is output. If f_N is higher than f_r , the down request signal is output.

Figure 12-9 shows the relation among reference frequency f_r , divided frequency f_N , up request signal \overline{UP} , and down request signal \overline{DW} .

When the PLL is disabled, neither the up nor the down request signal is output.

The up and down request signals are input to the charge pump and unlock FF.

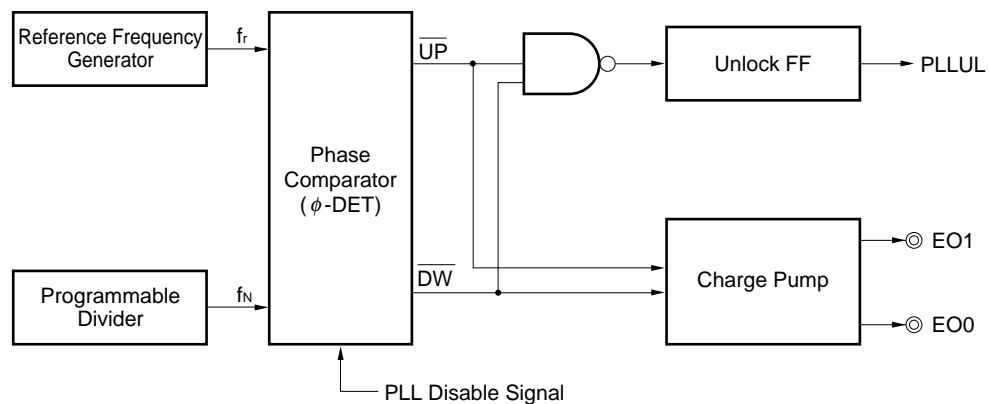
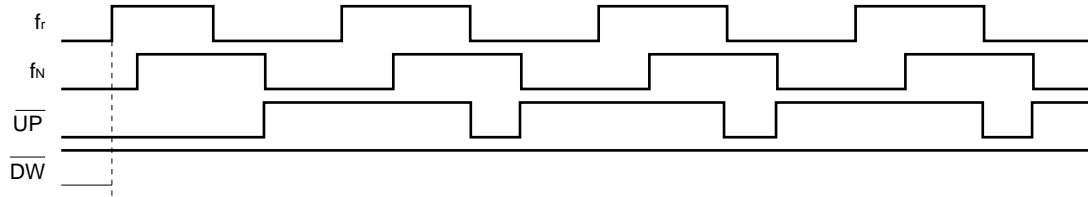
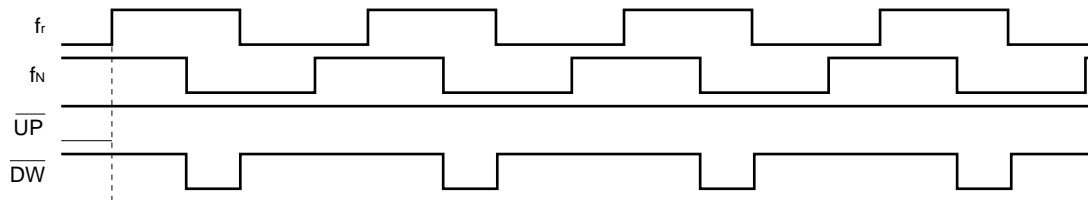
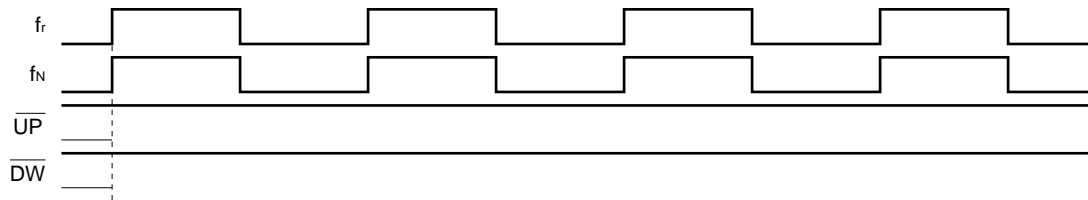
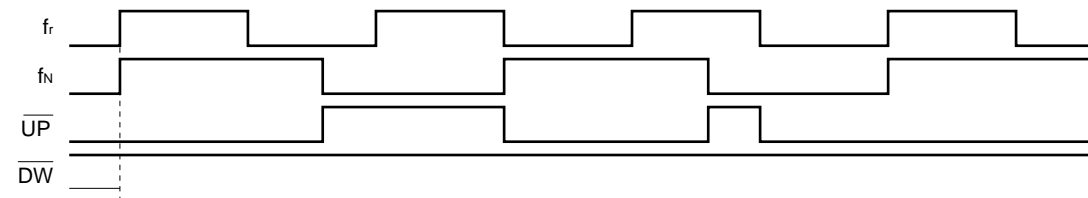
Figure 12-8. Phase Comparator, Charge Pump, and Unlock FF Configuration

Figure 12-9. Relationship between f_r , f_N , \overline{UP} , and \overline{DW} (a) If f_N advances f_r in phase(b) If f_N advances f_r in phase(c) If f_N and f_r are in phase(d) If f_N is lower than f_r **(4) Operation of charge pump**

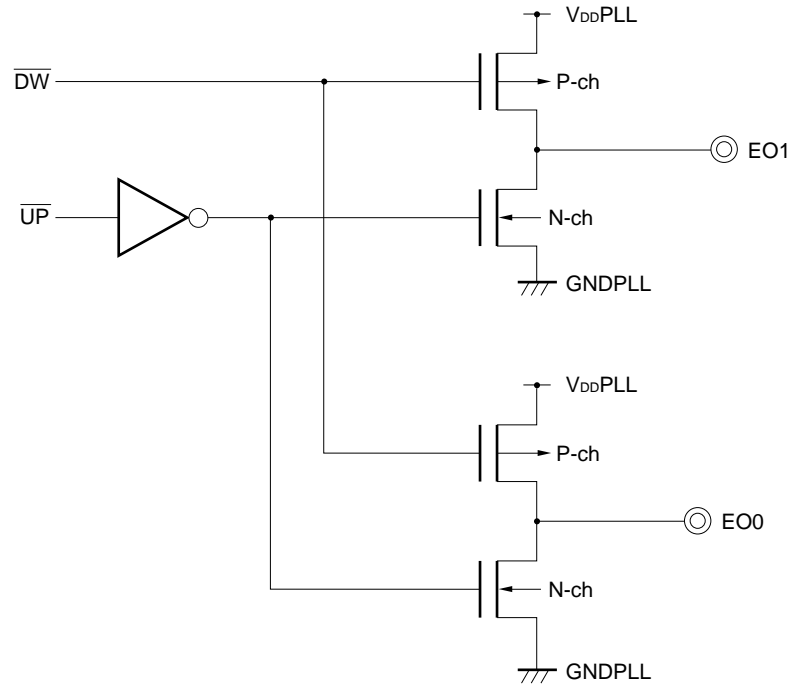
The charge pump outputs the result of the up request (\overline{UP}) or down request (\overline{DW}) signal from the phase comparator (ϕ -DET) from the error out pins (EO0 and EO1 pins). Table 12-3 shows the output signals.

The EO0 pin is of voltage-driven type, and EO1 pin is of current-driven type.

Figure 12-10 shows the configuration of the error out pins.

Table 12-3. Error Out Output Signal

Relationship between Divided Frequency f_N and Reference Frequency f_r	Error Out Output Signal
When $f_r > f_N$	Low level
When $f_r < f_N$	High level
When $f_r = f_N$	Floating (high impedance)

Figure 12-10. Error Out Pins Configuration

(5) Operation of unlock FF

The unlock FF detects the unlock status of the PLL frequency synthesizer.

It detects the unlock status of the PLL frequency synthesizer from the up request signal \overline{UP} and down request signal \overline{DW} of the phase comparator (ϕ -DET).

Because either of the up request or down request signal outputs a low level in the unlock status, the unlock status can be detected by using this low-level signal.

The status of the unlock FF is detected by bit 0 (PLLUL0) of the PLL unlock FF judge register (PLLUL).

The unlock FF is set at the cycle of reference frequency f_r selected at that time.

The PLL unlock FF judge register is reset when its contents have been read.

To read the PLL unlock FF judge register, therefore, it must be read at a cycle longer than the cycle ($1/f_r$) of the reference frequency.

12.4.2 Operation to set N value of PLL frequency synthesizer

The division value (N value) is set to the programmable counter (12 bits) and swallow counter (5 bits) by the PLL data registers (PLLRL, PLLRH, and PLLR0).

When the N value has been transferred to the programmable counter and swallow counter by bit 0 (PLLNS0) of the PLL data transfer register (PLLNS), frequency division is carried out in the selected division mode.

Examples of setting the N value in the respective division modes (MF, HF, and VHF) are shown below.

(1) Direct division mode (MF)

(a) Calculating division value N (value set to PLL data register)

$$N = \frac{f_{V_{COL}}}{f_r}$$

where, $f_{V_{COL}}$: input frequency of V_{COL} pin
 f_r : reference frequency

(b) Example of setting PLL data register

An example of setting the PLL data register to receive broadcasting stations in the following MW band is shown below.

Receive frequency : 1422 kHz (MW band)

Reference frequency : 9 kHz

Intermediate frequency : 450 kHz

Division value N is calculated as follows:

$$N = \frac{f_{V_{COL}}}{f_r} = \frac{1422 + 450}{9} = 208 \text{ (decimal)}$$

$$= 0D0H \text{ (hexadecimal)}$$

Data is set to the PLL data registers (PLLRL and PLLRH) as follows:

PLLRL																PLLR0																
PLLRH								PLLRL								← PLLSCN																
b7	b6	b5	b4	b3	b2	b1	b0	b7	b6	b5	b4	b3	b2	b1	b0	b7	b6	b5	b4	b3	b2	b1	b0									
b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																
Programmable counter value												Don't care				Fixed to 0																
0	0	0	0	1	1	0	1	0	0	0	0																					
0				D				0																								

After setting the above PLL data registers (PLLRL and PLLRH), data must be transferred to the programmable counter by setting bit 0 (PLLNS0) of the PLL data transfer register (PLLNS).

(2) Pulse swallow mode (HF)

(a) Calculating division value N (value set to PLL data register)

$$N = \frac{f_{\text{VCO}}}{f_r}$$

where, f_{VCO} : input frequency of V_{CO} pin
 f_r : reference frequency

(b) Example of setting PLL data register

An example of setting the PLL data register to receive broadcasting stations in the following SW band is shown below.

Receive frequency : 25.50 MHz (SW band)

Reference frequency : 5 kHz

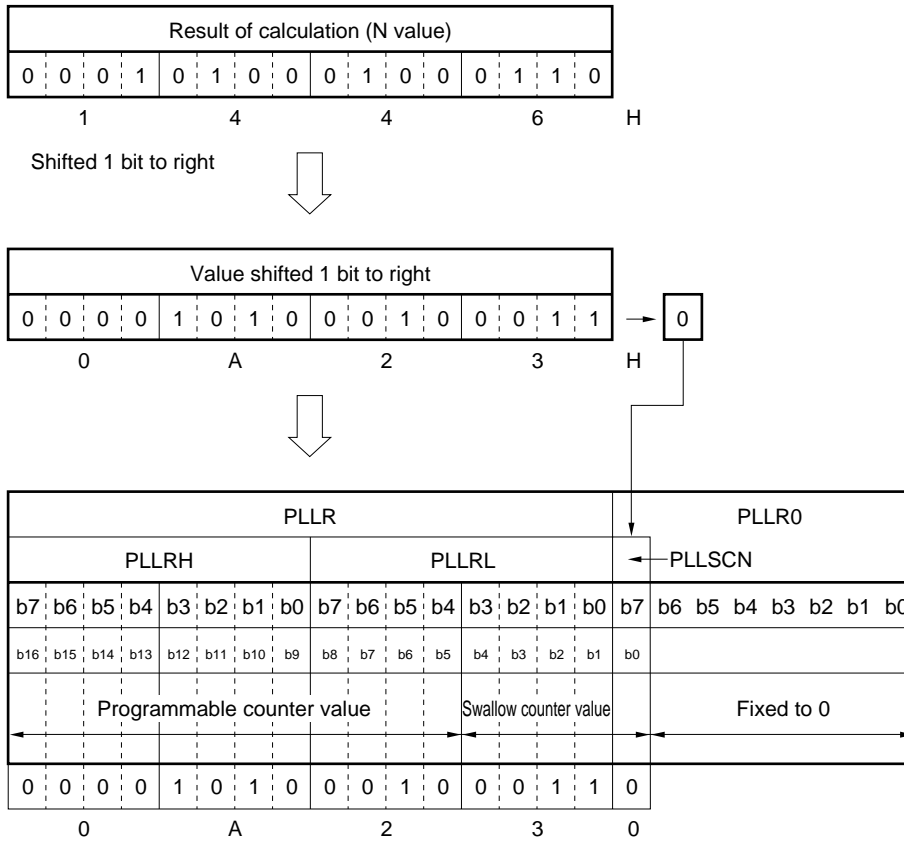
Intermediate frequency : 450 kHz

Division value N is calculated as follows:

$$N = \frac{f_{\text{VCO}}}{f_r} = \frac{25500 + 450}{5} = 5190 \text{ (decimal)}$$

$$= 01446\text{H (hexadecimal)}$$

Because the least significant bit of the division value N must be set to bit 7 (PLLSCN) of the PLL data register 0 (PLLRO), data must be set by shifting the result of the above calculation 1 bit to the right. Data is set to the PLL data registers (PLLRL and PLLRH) as follows:



After setting the above PLL data registers (PLLRL and PLLRH), data must be transferred to the programmable counter and swallow counter by setting bit 0 (PLLNS0) of the PLL data transfer register (PLLNS).

In this example, a value of half the N value is set to the high-order 16 bits of the PLL data register (PLLRL) by shifting the N value resulting from calculation 1 bit to the right.

If the N value is calculated as follows with the least significant bit of the N value in PLLSCN fixed to 0, the result of the calculation (N_{PLLRL}) can be set to the PLL data register (PLLRL) as is.

If the calculation result is set in this way, however, the input frequency (f_{VCO}) is $2 \times f_r$ (reference frequency) of the set value N_{PLLRL} .

$$N_{PLLRL} = \frac{f_{VCO}}{2f_r}$$

(3) Pulse swallow mode (VHF)

(a) Calculating division value N (value set to PLL data register)

$$N = \frac{f_{VCOH}}{f_r}$$

where, f_{VCOH} : input frequency of VCOH pin
 f_r : reference frequency

(b) Example of setting PLL data register

An example of setting the PLL data register to receive broadcasting stations in the following FM band is shown below.

Receive frequency : 100.0 MHz (FM band)

Reference frequency : 25 kHz

Intermediate frequency : +10.7 MHz

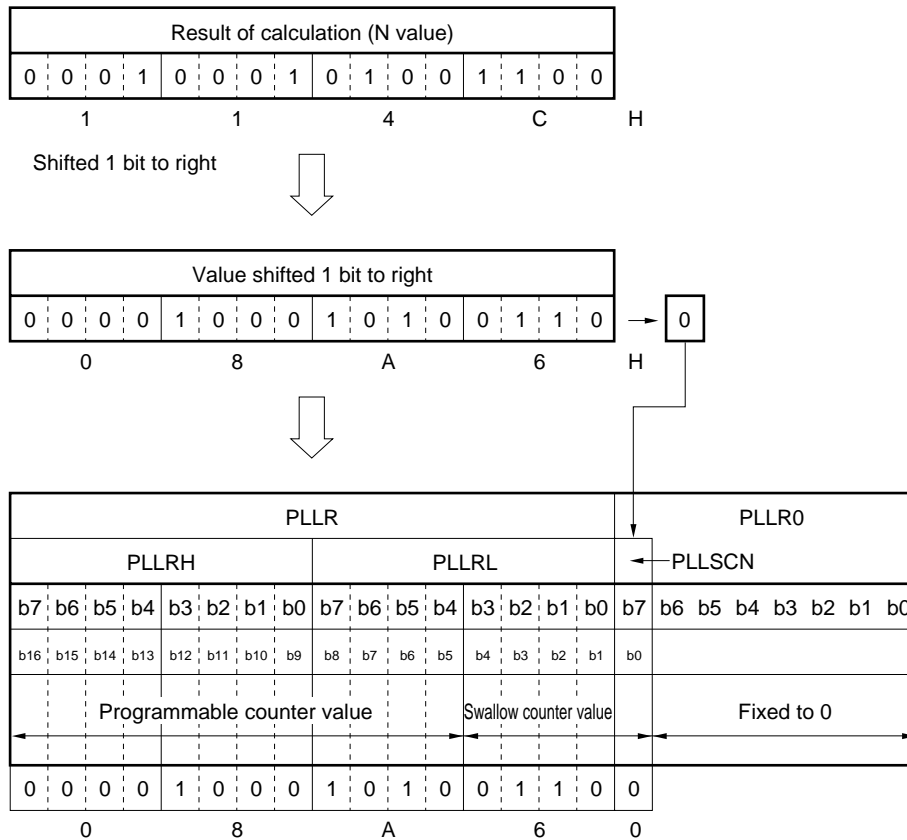
Division value N is calculated as follows:

$$N = \frac{f_{VCOH}}{f_r} = \frac{100.0 + 10.7}{0.025} = 4428 \text{ (decimal)}$$

$$= 0114CH \text{ (hexadecimal)}$$

Because the least significant bit of the division value N must be set to the PLL data register 0 (PLLRO), data must be set by shifting the value calculated by the above expression 1 bit to the right.

Data is set to the PLL data registers (PLLRL and PLLRH) as follows:



After setting the above PLL data registers (PLLR and PLLR0), data must be transferred to the programmable counter and swallow counter by setting bit 0 (PLLNS0) of the PLL data transfer register (PLLNS).

In this example, a value of half the N value is set to the high-order 16 bits of the PLL data register (PLLR) by shifting the N value resulting from calculation 1 bit to the right.

If the N value is calculated as follows with the least significant bit of the N value in PLLSCN fixed to 0, the result of the calculation (N_{PLLR}) can be set to the PLL data register (PLLR) as is.

If the calculation result is set in this way, however, the input frequency (f_{VCOH}) is $2 \times f_r$ (reference frequency) of the set value N_{PLLR} .

$$N_{\text{PLLR}} = \frac{f_{\text{VCOH}}}{2f_r}$$

12.5 PLL Disable Status

The PLL frequency synthesizer can be stopped (PLL disabled status) by performing any of the following settings while the PLL frequency synthesizer is operating.

- Setting value of PLL reference mode register to 0BH or 0FH to set PLL disabled status
- Setting STOP mode with the STOP instruction
- Setting reset status with the reset function

The following table shows the operation of each block and the status of each register in the PLL disabled status.

Table 12-4. Operation of Each Block and Register Status in PLL Disabled Status

Block/Register	Status in PLL Disabled Status
VCOL and VCOH pins	High impedance
Programmable divider	Division stops
Reference frequency generator	Output stops
Phase comparator	Output stops
EO0 and EO1 pin	High impedance
PLL mode select register	Retains value on execution of write instruction
PLL data register	
PLL unlock FF judge register	

12.6 Notes on PLL Frequency Synthesizer

• Notes on using PLL frequency synthesizer

Because the input pins (VCOL and VCOH pins) of the PLL frequency synthesizer are provided with an AC amplifier, cut the DC component of the input signal by connecting a capacitor to the input pins in series.

The potential of the selected input pin is intermediate (about $1/2V_{DD}$). The input pin not selected goes into a high-impedance state.

For the frequencies that can be actually input and input amplitude, refer to Electrical Specifications in Data Sheet.

[MEMO]

CHAPTER 13 FREQUENCY COUNTER

13.1 Function of Frequency Counter

The frequency counter counts the intermediate frequency (IF) of a tuner.

It counts the intermediate frequency input to the FMIFC or AMIFC pin for a specific time (1 ms, 4 ms, 8 ms, or open) with a 16-bit counter. The count value of the frequency counter is stored to the IF counter register.

For the range of the frequency that can be input to the FMIFC and AMIFC pins, refer to Electrical Characteristics in Data Sheet.

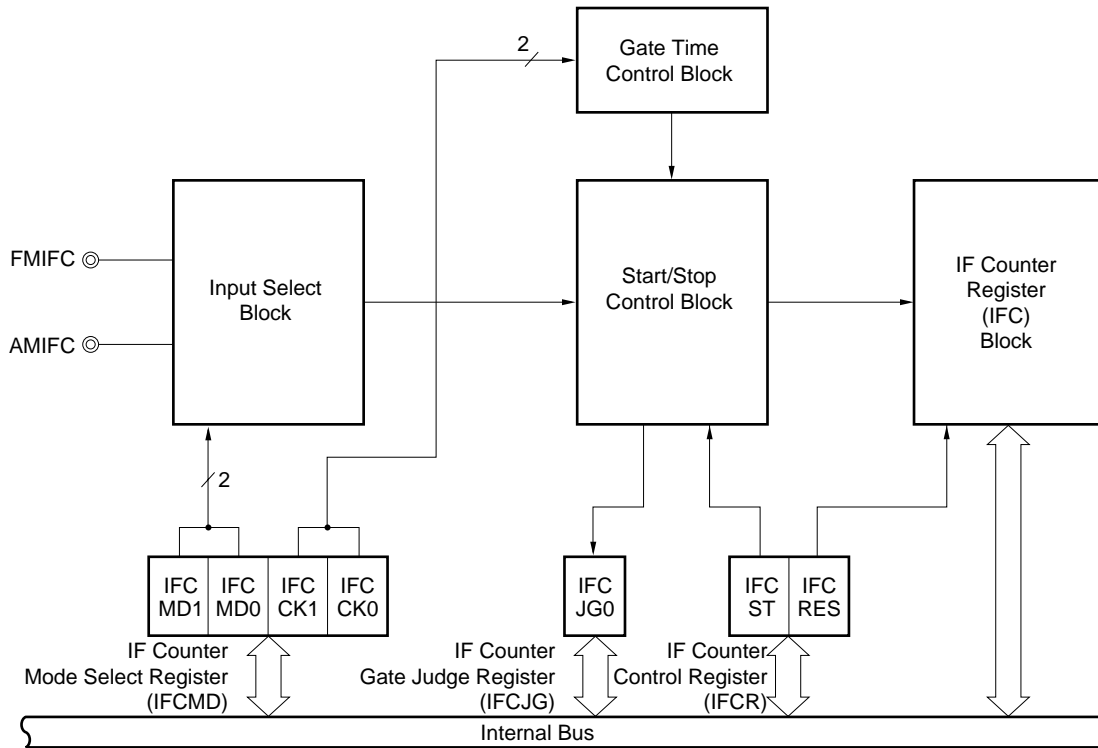
13.2 Configuration of Frequency Counter

The frequency counter consists of the following hardware units.

Table 13-1. Frequency Counter Configuration

Item	Configuration
Counter register	IF counter register (IFC)
Control register	IF counter mode select register (IFCMD) IF counter control register (IFCR) IF counter gate judge register (IFCJG)

Figure 13-1. Frequency Counter Block Diagram



(1) IF counter input select block

The IF counter input select block selects the pin to be used from the FMIFC and AMIFC pins, and a count mode.

(2) Gate time control block

The gate time control block sets a gate time (count time).

(3) Start/stop control block

The start/stop control block starts counting by the IF counter register and detects the end of counting.

(4) IF counter register block

The IF counter register block is a 16-bit register that counts up the frequency input in the set gate time. The count value is stored to the IF counter register (IFC). When the count value reaches FFFFH, the IF counter register holds FFFFH at the next input, and stops counting. The value of this register is reset to 0000H at reset or in the STOP mode. In the HALT mode, it holds the current count value.

13.3 Registers Controlling Frequency Counter

The frequency counter is controlled by the following three registers.

- IF counter mode select register (IFCMD)
- IF counter control register (IFCR)
- IF counter gate judge register (IFCJG)

(1) IF counter mode select register (IFCMD)

This register selects the input pin of the frequency counter, and selects a mode and gate time (count time).

This register is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is reset to 00H at reset or in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode is set.

Figure 13-2. IF Counter Mode Select Register (IFCMD) Format

Symbol	7	6	5	4	③	②	①	①	Address	After Reset	R/W
IFCMD	0	0	0	0	IFCMD1	IFCMD0	IFCCK1	IFCCK0	FFA9H	00H	R/W

IFCMD1	IFCMD0	Selects Frequency Counter Pin and Mode
0	0	Disables FMIFC AMIFC pins ^{Note}
0	1	AMIFC pin, AMIF count mode
1	0	FMIFC pin, FMIF count mode
1	1	FMIFC pin, AMIF count mode

IFCCK1	IFCCK0	Selects Gate Time
0	0	1 ms
0	1	4 ms
1	0	8 ms
1	1	Open

Note The FMIFC and AMIFC pins go into a high-impedance state.

Remark Bits 4 through 7 are fixed to 0 by hardware.

(2) IF counter control register (IFCR)

This register starts counting by the IF counter register and clears the IF counter register.

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

The value of this register is reset to 00H at reset and in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode is set.

Figure 13-3. IF Counter Control Register (IFCR) Format

Symbol	7	6	5	4	3	2	①	②	Address	After Reset	R/W
IFCR	0	0	0	0	0	0	IFCST	IFCRES	FFACH	00H	W

IFCST	Starts IF Counter Register
0	Nothing is affected
1	Starts counting

IFCRES	Clears Data of IF Counter Register
0	Nothing is affected
1	Clears data of IF counter register

Remark Bits 2 through 7 are fixed to 0 by hardware.

(3) IF counter gate judge register (IFCJG)

This register detects opening/closing of the gate of the frequency counter.

The value of this register is reset to 00H at reset and in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode is set.

Figure 13-4. IF Counter Gate Judge Register (IFCJG) Format

Symbol	7	6	5	4	3	2	1	②	Address	After Reset	R/W
IFCJG	0	0	0	0	0	0	0	IFCJG0	FFABH	00H	R

IFCJG0	Detects Opening/Closing of Gate of Frequency Counter
0	Gate is closed
1	<ul style="list-style-type: none"> If gate time is set to other than open Status until gate is closed after IFCST has been set to 1 If gate time is set to open Status where gate is open as soon as it has been set to be opened

Remark Bits 1 through 7 are fixed to 0 by hardware.

Caution IFCJG0 remains set even if the IF counter register overflows and stops counting, until the set gate time expires.

13.4 Operation of Frequency Counter

- (1) Select an input pin, mode, and gate time by using the IF counter mode select register (IFCMD).
Figure 13-5 shows a block that selects an input pin and mode.
- (2) Set bit 0 (IFCRES) of the IF counter control register (IFCR) to 1, and clears the data of the IF counter register.
- (3) Set bit 1 (IFCST) of the IF counter control register (IFCR) to 1.
- (4) The gate is opened only for the set gate time since a 1-kHz internal signal has risen after IFCST was set. If the gate time is set to be opened, the gate is opened as soon as it has been specified to be opened.
Bit 0 (IFCJG0) of the IF counter gate judge register (IFCJG) is automatically set to 1 as soon as IFCST has been set to 1.
When the gate time has expired, bit 0 (IFCJG0) of the IF counter gate judge register (IFCJG) is automatically cleared to 0. If it is specified that the gate be open, however, IFCJG0 is not automatically cleared. In this case, set a gate time. Figure 13-6 shows the gate timing of the frequency counter.
- (5) While the gate opens the frequency input to the selected FMIFC or AMIFC pin, the IF counter register counts the frequency.
If the FMIFC pin is used in the FMIF count mode, however, the input frequency is divided by half before it is counted.

The relationship between count value x (decimal), input frequencies (f_{FMIFC} and f_{AMIFC}), and gate time (T_{GATE}) is shown below.

- FMIF count mode (FMIFC pin)

$$f_{\text{FMIFC}} = \frac{x}{T_{\text{GATE}}} \times 2 \text{ (kHz)}$$

- AMIF count mode (FMIFC or AMIFC pin)

$$f_{\text{AMIFC}} = \frac{x}{T_{\text{GATE}}} \text{ (kHz)}$$

Figure 13-5. Input Pin and Mode Selection Block Diagram

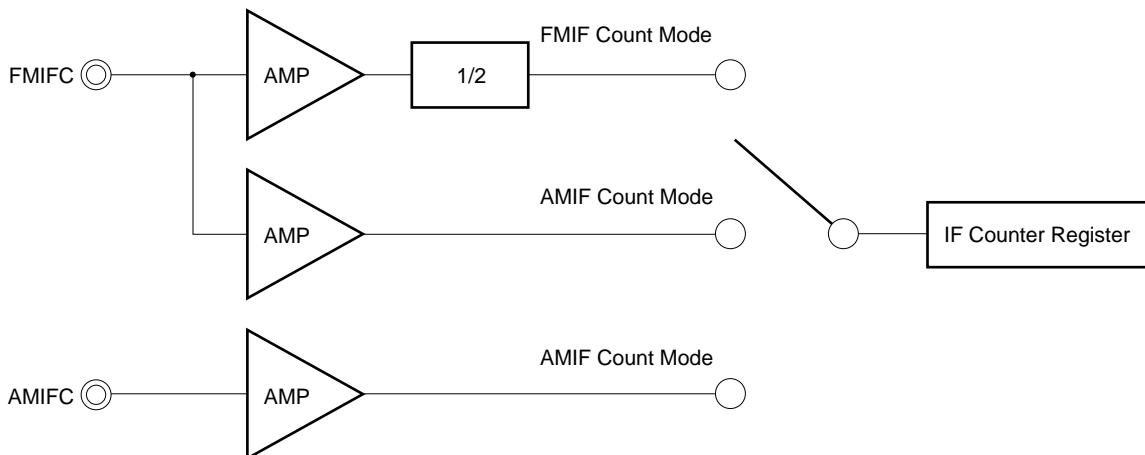
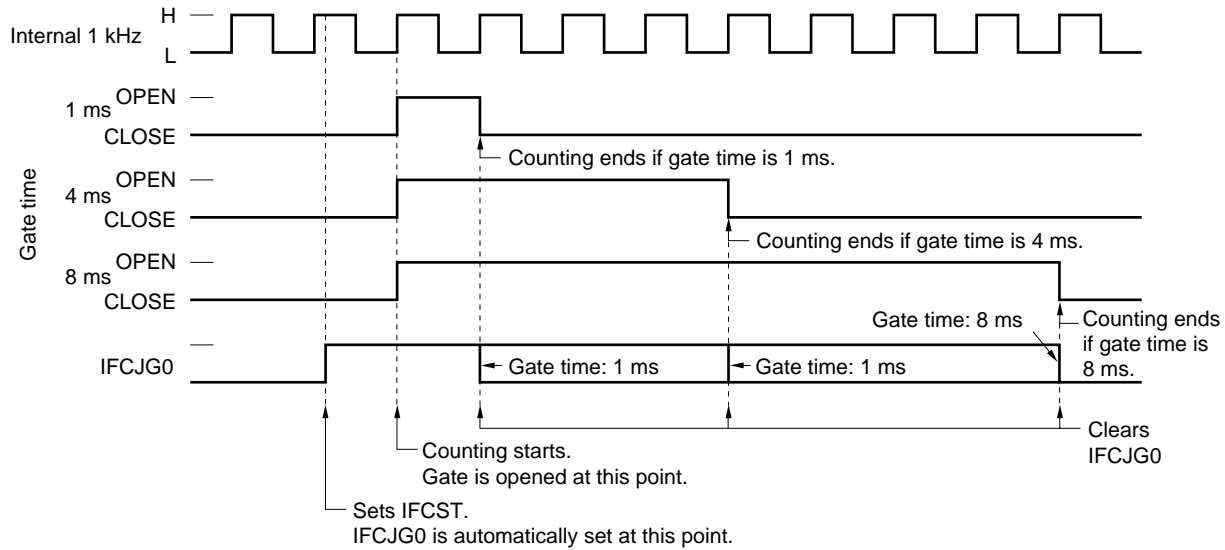
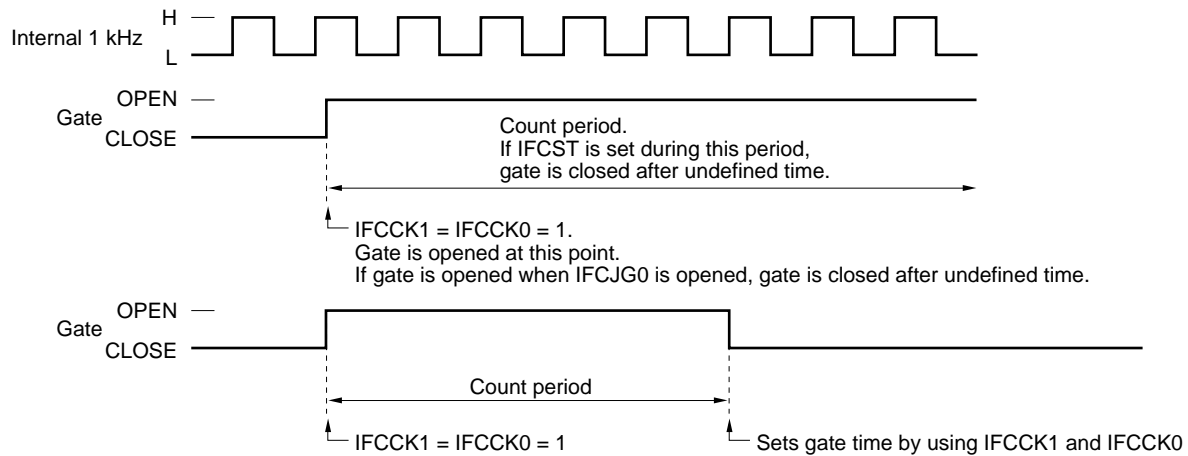


Figure 13-6. Gate Timing of Frequency Counter

(a) If gate time is set to 1, 4, or 8 ms



(b) If gate is set to be open



Caution If counting is started by using IFCST while this gate is open, the gate is closed after undefined time. To open the gate, therefore, do not set IFCST to 1.

Remark IFCST : Bit 1 of IF counter control register (IFCR)
 IFCJG0 : Bit 0 of IF counter gate judge register (IFJG)
 IFCCK1, 0 : Bit 1 and 0 of IF counter mode select register (IFCMD)

13.5 Notes on Frequency Counter

(1) Notes on using frequency counter

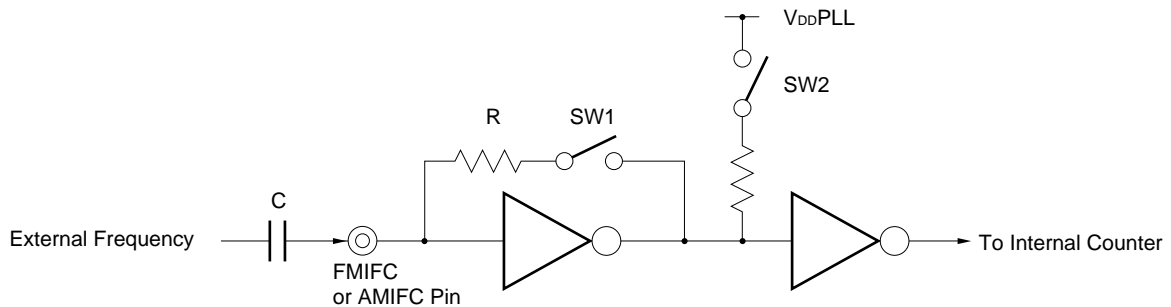
Because signals are input to the frequency counter from an input pin (FMIFC or AMIFC pin) with an AC amplifier as shown in Figure 13-7, cut the DC component of the input signals by using capacitor C.

If the FMIFC or AMIFC pin is selected by the IF counter mode select register, switch SW1 turns ON, and switch SW2 turns OFF. As a result, the voltage on the pin is about $1/2V_{DD}$.

Unless the voltage has risen to a sufficient intermediate level at this time, counting may not be performed normally because the AC amplifier is not in the normal operating range.

Therefore, make sure that sufficient wait time elapses after a pin has been selected and before counting is started (IFCST = 1).

Figure 13-7. Frequency Counter Input Pin Circuit



(2) Notes in HALT mode

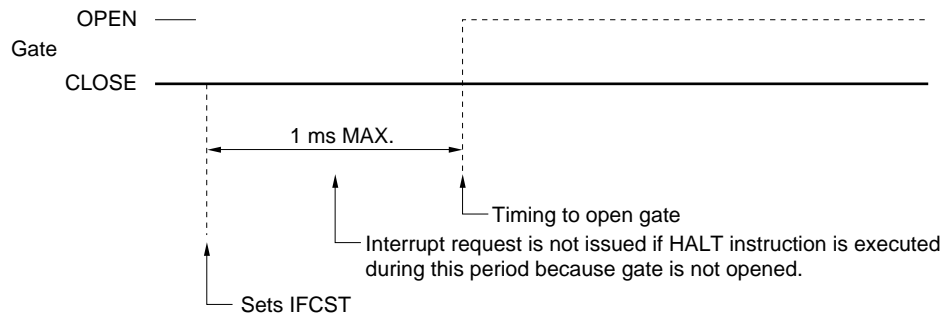
The FMIFC and AMIFC pins hold the status immediately before the HALT status was set.

To release the HALT mode by using the interrupt of the frequency counter at this time, the following point must be noted.

The gate will not be opened if the HALT instruction is executed after counting has been started by IFCST before the gate is actually opened.

Therefore, wait for at least 1 ms before executing the HALT instruction.

Figure 13-8. Gate Status When HALT Instruction Is Executed



(3) Error of frequency counter

The error of the frequency counter includes an error of gate time and a count error.

(1) Error of gate time

The gate time of the frequency counter is created by dividing 4.5 MHz.

Therefore, if 4.5 MHz is shifted “+x” ppm, the gate time is also shifted “-x” ppm.

(2) Count error

The frequency counter counts the frequency at the rising edge of the input signal.

If a high level is input to the pin when the gate is opened, therefore, one excess pulse is counted. When the gate is closed, however, counting is not affected by the status of the pin.

Therefore, the count error is “maximum + 1”.

CHAPTER 14 STANDBY FUNCTION

14.1 Standby Function and Configuration

14.1.1 Standby function

The standby function is designed to decrease 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. System clock oscillator continues oscillation. In this mode, current consumption cannot be decreased as in the STOP mode. The HALT mode is valid to restart immediately upon interrupt request and to carry out intermittent operations such as in watch applications.

(2) STOP mode

STOP instruction execution sets the STOP mode. In the STOP mode, the system clock oscillator stops and the whole system stops. CPU current consumption can be considerably decreased.

Data memory low-voltage hold (down to $V_{DD} = 1.8\text{ V}$) is possible. Thus, the STOP mode is effective to hold 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 necessary 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 any mode, all the contents of the register, flag and data memory just before standby mode setting are held. The input/output port output latch and output buffer statuses are also held.

However, in STOP mode, the contents of registers and flags of the PLL frequency synthesizer and frequency counter are not held. Refer to **CHAPTER 12 PLL FREQUENCY SYNTHESIZER** and **CHAPTER 13 FREQUENCY COUNTER**.

- Cautions**
1. When proceeding to the STOP mode, be sure to stop the peripheral hardware operation and execute the STOP instruction.
 2. The following sequence is recommended for power consumption reduction of the A/D converter: first clear bit 7 (CS) of the A/D converter mode register (ADM) to 0 to stop the A/D conversion operation. Clear bit 3 (ADON) of the A/D converter input select register (ADIS) to 0 and turn the resistor string power supply off. Then execute the HALT or STOP instruction.

14.1.2 Standby function control register

A wait time after the STOP mode is cleared upon interrupt request till the oscillation stabilizes is controlled with the oscillation stabilization time select register (OSTS).

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

Reset input sets OSTS to 04H. However, it takes $2^{17}/f_x$, not $2^{18}/f_x$, until the STOP mode is cleared by $\overline{\text{RESET}}$ input.

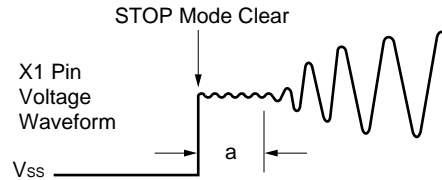
Figure 14-1. Oscillation Stabilization Time Select Register (OSTS) Format

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		
				MCS = 1	MCS = 0
0	0	0	$2^{12}/f_{xx}$	$2^{12}/f_x(90 \mu s)$	$2^{13}/f_x(1.82 \text{ ms})$
0	0	1	$2^{14}/f_{xx}$	$2^{14}/f_x(3.64 \text{ ms})$	$2^{15}/f_x(7.28 \text{ ms})$
0	1	0	$2^{15}/f_{xx}$	$2^{15}/f_x(7.28 \text{ ms})$	$2^{16}/f_x(14.6 \text{ ms})$
0	1	1	$2^{16}/f_{xx}$	$2^{16}/f_x(14.6 \text{ ms})$	$2^{17}/f_x(29.1 \text{ ms})$
1	0	0	$2^{17}/f_{xx}$	$2^{17}/f_x(29.1 \text{ ms})$	$2^{18}/f_x(58.3 \text{ ms})$
Other than above			Setting prohibited		

- Remarks**
1. f_{xx} : System clock frequency (f_x or $f_x/2$)
 2. f_x : System clock oscillation frequency
 3. MCS : Oscillation mode selection register (OSMS) bit 0
 4. Values in parentheses apply to operating at $f_x = 4.5 \text{ MHz}$

Caution The wait time when the STOP mode is released does not include the time required for the clock oscillation to start after the STOP mode has been released (see “a” in the figure below), regardless of whether the mode has been released by the $\overline{\text{RESET}}$ signal or an interrupt request.



14.2 Standby Function Operations

14.2.1 HALT mode

(1) HALT mode set and operating status

The HALT mode is set by executing the HALT instruction.

The operating status in the HALT mode is described below.

Table 14-1. HALT Mode Operating Status

Item	Status
Clock generator	Can oscillate system clock. Stops clock supply to CPU.
CPU	Stops operating.
Port	Holds status before HALT mode is set.
8-bit timer/event counter	Holds operation before HALT mode is set and can operate.
Basic timer	
Buzzer output control circuit	
A/D converter	
Serial interface	Holds operation before HALT mode is set and can operate.
External interrupt	INTP0 stops only if sampling clock is $f_{xx}/2^N$. Otherwise, holds operation before HALT mode is set and can operate. INTP1 holds operation before HALT mode is set and can operate.
PLL frequency synthesizer	Hold operation before HALT mode is set and can operate.
Frequency counter	
Power-ON clear circuit	Can operate

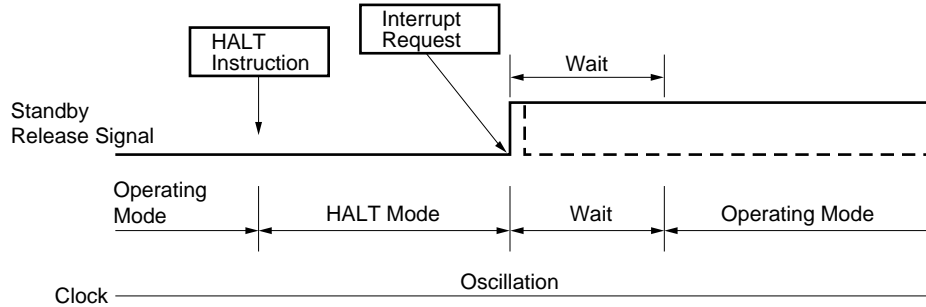
(2) HALT mode release

The HALT mode can be released with the following three types of sources.

(a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the HALT mode is released. If interrupt request acknowledge is enabled, vectored interrupt service is carried out. If disabled, the next address instruction is executed.

Figure 14-2. HALT Mode Release by Interrupt Generation



Remarks 1. The broken line indicates the case when the interrupt request which has released the standby status is acknowledged.

2. Wait time will be as follows:

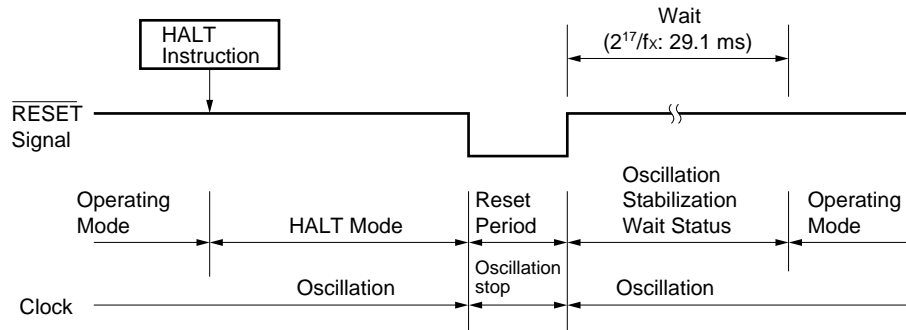
- When vectored interrupt service is carried out: 8 to 9 clocks
- When vectored interrupt service is not carried out: 2 to 3 clocks

(b) Release by unmasked test input

If an unmasked test is input, the HALT mode is released and the next address instruction of the HALT instruction is executed.

(c) Release by $\overline{\text{RESET}}$ input

If a $\overline{\text{RESET}}$ signal is input, the HALT mode is released. As is the case with normal reset operation, a program is executed after branch to the reset vector address.

Figure 14-3. HALT Mode Release by $\overline{\text{RESET}}$ Input

Remarks 1. f_x : System clock oscillation frequency

2. (): At $f_x = 4.5 \text{ MHz}$

Table 14-2. Operation after HALT Mode Release

Release Source	MK $\times\times$	PR $\times\times$	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt service execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt service execution
	1	×	×	×	HALT mode hold
Test input	0	—	×	×	Next address instruction execution
	1	—	×	×	HALT mode hold
$\overline{\text{RESET}}$ input	—	—	×	×	Reset processing

Remark ×: Don't care

14.2.2 STOP mode

(1) STOP mode set and operating status

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_{DD} via a pull-up resistor to minimize the leakage current at the crystal oscillator.
 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 released if set. Thus, the STOP mode is reset to the HALT mode immediately after execution of the STOP instruction. After the wait set using the oscillation stabilization time select register (OSTS), the operating mode is set.

The operating status in the STOP mode is described below.

Table 14-3. STOP Mode Operating Status

Item	Status
Clock generator	Can oscillate system clock. Stops clock supply to CPU.
CPU	Stops operating.
Port	Holds status before HALT mode is set.
8-bit timer/event counter	Operable when TI1 and TI2 are set to count clock. Holds status before STOP mode is set. Other functions stop operation and cannot operate.
Basic timer	Operation stops and cannot operate.
Buzzer output control circuit	
A/D converter	
Serial interface	Holds operation before STOP mode is set and can operate.
External interrupt	INTP0 stops and cannot operate. INTP1 holds operation before STOP mode is set and can operate.
PLL frequency synthesizer	Operation stops and cannot operate.
Frequency counter	
Power-ON clear circuit	Can operate

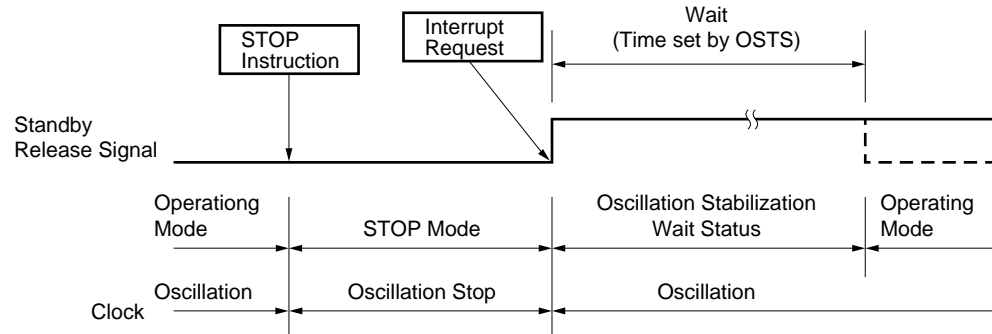
(2) STOP mode release

The STOP mode can be released with the following three types of sources.

(a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the STOP mode is released. If interrupt request acknowledge is enabled after the lapse of oscillation stabilization time, vectored interrupt service is carried out. If interrupt request acknowledge is disabled, the next address instruction is executed.

Figure 14-4. STOP Mode Release by Interrupt Request Generation



Remark The broken line indicates the case when the interrupt request which has released the standby status is acknowledged.

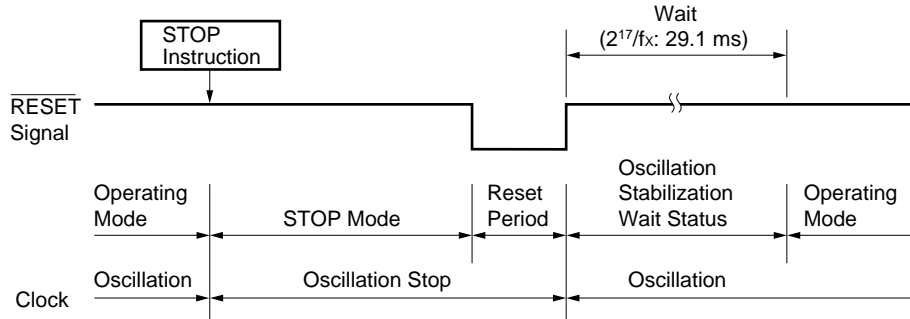
(b) Release by unmasked test input

If an unmasked test is input, the STOP mode is released. After the lapse of oscillation stabilization time, the instruction at the next address of the STOP instruction is executed.

(c) Release by $\overline{\text{RESET}}$ input

If a $\overline{\text{RESET}}$ signal is input, the STOP mode is released, and after the lapse of oscillation stabilization time, reset operation is carried out.

Figure 14-5. Release by STOP Mode $\overline{\text{RESET}}$ Input



Remarks 1. f_x : System clock oscillation frequency

2. (): At $f_x = 4.5 \text{ MHz}$

Table 14-4. Operation after STOP Mode Release

Release Source	MK $\times\times$	PR $\times\times$	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt service execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt service execution
	1	×	×	×	STOP mode hold
Test input	0	–	×	×	Next address instruction execution
	1	–	×	×	STOP mode hold
$\overline{\text{RESET}}$ input	–	–	×	×	Reset processing

Remark ×: Don't care

CHAPTER 15 RESET FUNCTION

15.1 Reset Function

The following two operations are available to generate the reset signal.

- (1) External reset input with $\overline{\text{RESET}}$ pin
- (2) Internal reset by power-ON clear (POC)

There is no functional difference between external reset and internal reset. Program execution is started from the address written in 0000H and 0001H in both methods.

(1) External reset input by $\overline{\text{RESET}}$ pin

When a low level is input to the $\overline{\text{RESET}}$ pin, the device is reset, and each hardware unit enters the status shown in Table 15-1. While the reset signal is input and during the oscillation stabilization time immediately after the $\overline{\text{RESET}}$ signal has been deasserted, each pin goes into a high-impedance state (the P132 through P134 pins go low, however).

The $\overline{\text{RESET}}$ signal is deasserted when a high level is input to the $\overline{\text{RESET}}$ pin, and the program execution is started after the oscillation stabilization time ($2^{17}/f_x$) has elapsed.

(2) Internal reset by power-ON clear (POC)

Reset is effected by means of power-ON clear under the following conditions:

- If supply voltage is less than $3.5 V^{\text{Note}}$ on power application
- If supply voltage drops to less than $2.5 V^{\text{Note}}$ in STOP mode
- If supply voltage drops to less than $3.5 V^{\text{Note}}$ while operation is performed with CPU clock of $f_x/2$ or lower (including HALT mode). Also if supply voltage drops to less than $4.5 V^{\text{Note}}$ while operation is performed with CPU clock of f_x (including HALT mode)

When these reset conditions of power-ON clear are satisfied, reset is effected, and each hardware unit enters the status shown in Table 15-1. While the reset signal is input and during the oscillation stabilization time immediately after the reset signal has been deasserted, each pin goes into a high-impedance state (the P132 through P134 pins go low, however).

Reset by power-ON clear is cleared if the supply voltage rises beyond a specific level, and the program execution is started after the oscillation stabilization time ($2^{17}/f_x$) has elapsed.

Note These voltage values are maximum values. Actually, reset is effected at a voltage lower than these.

- Cautions**
1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.
 2. During reset input, system clock oscillation remains stopped.
 3. When the STOP mode is cleared by $\overline{\text{RESET}}$ input, the STOP mode register contents are held during reset input. However, the I/O port pin becomes high-impedance. Output dedicated port pin (P132 to P134) becomes low level regardless of the previous status.

Figure 15-1. Reset Function Block Diagram

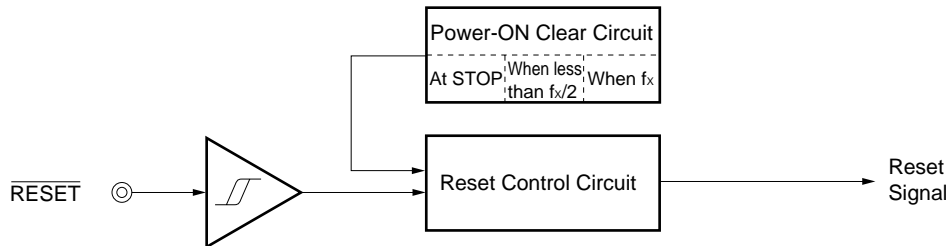
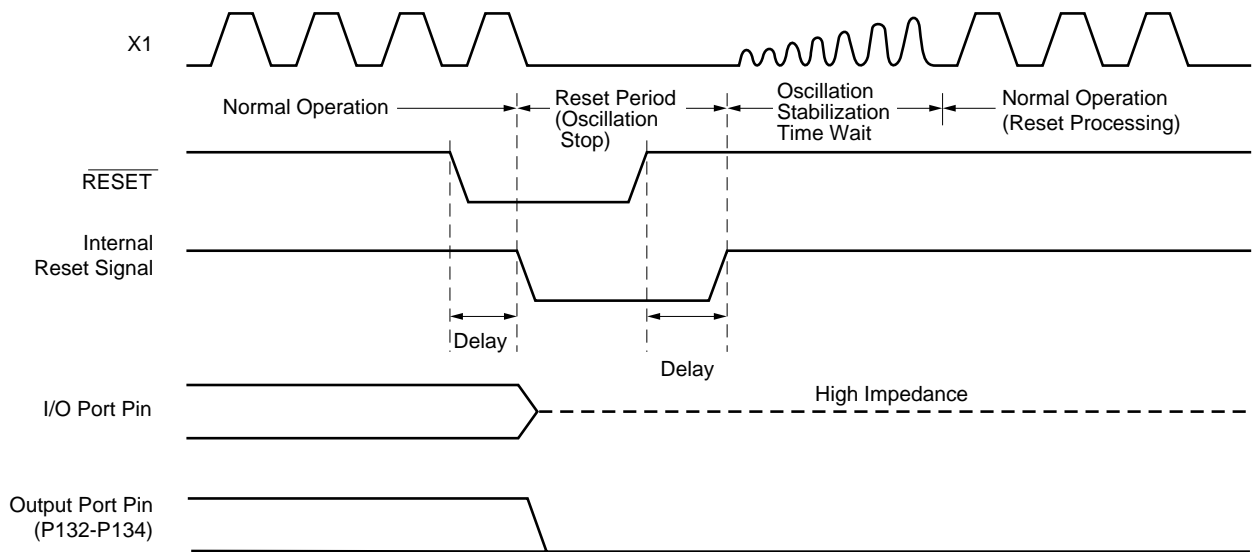


Figure 15-2. Timing of Reset Input by $\overline{\text{RESET}}$ Input

(a) In normal operating mode



(b) In STOP mode

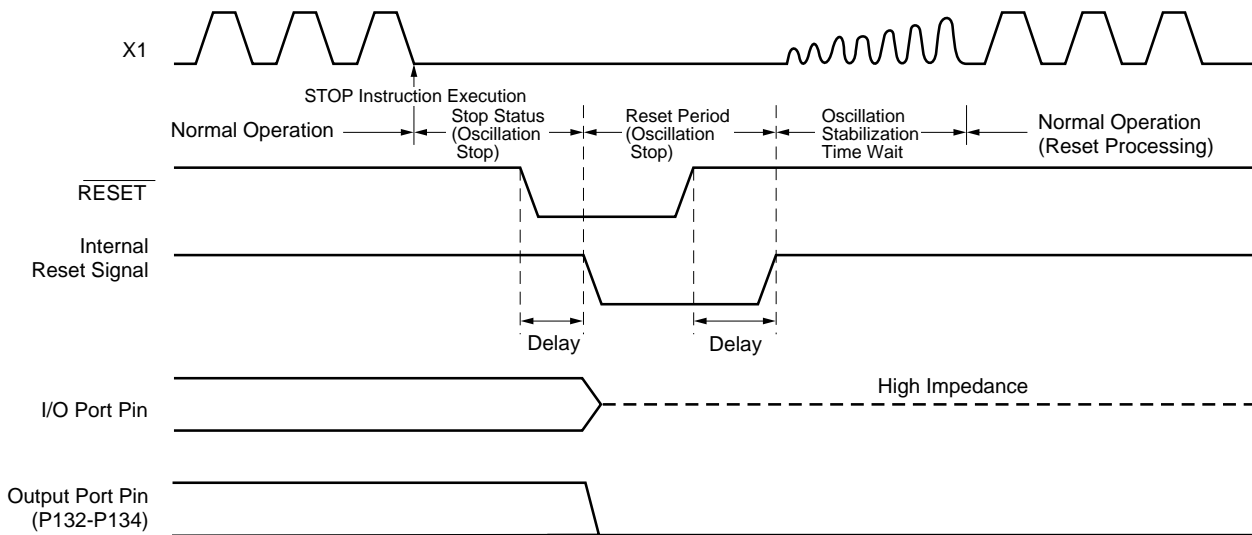
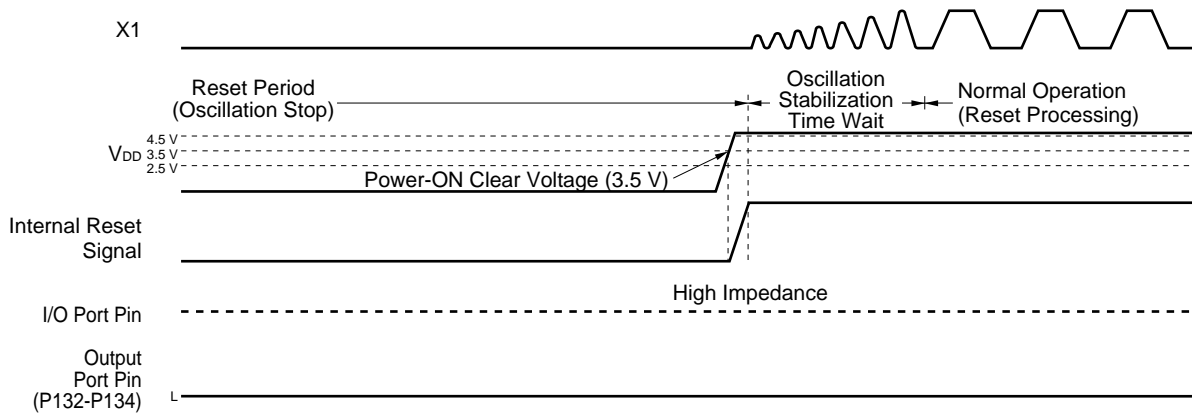


Figure 15-3. Timing of Reset by Power-ON Clear (1/2)

(a) At Power-ON



(b) In STOP mode

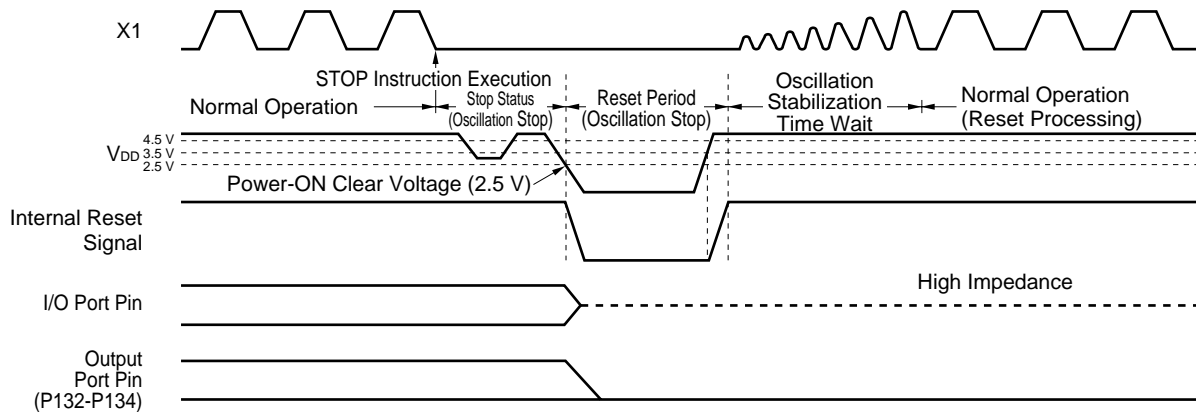


Figure 15-3. Timing of Reset by Power-ON Clear (2/2)

(c) In normal operating mode (including HALT mode)

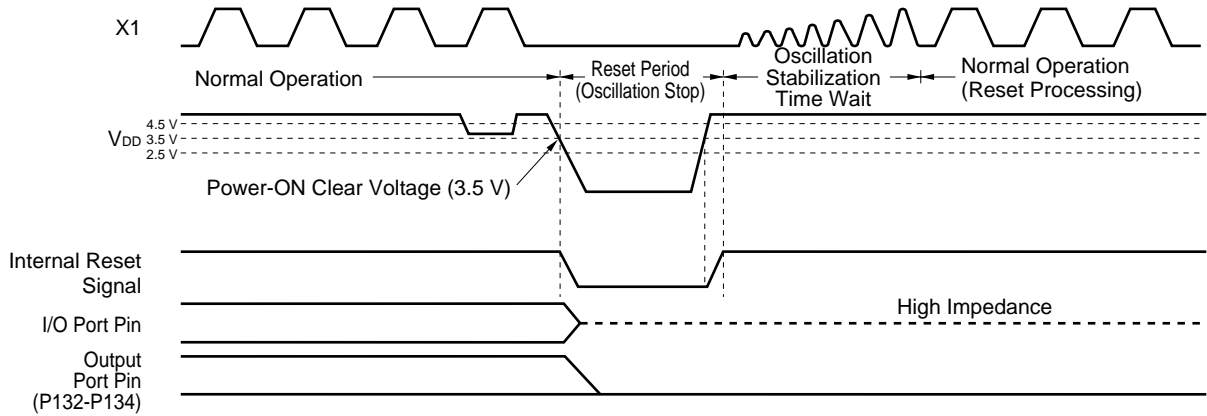
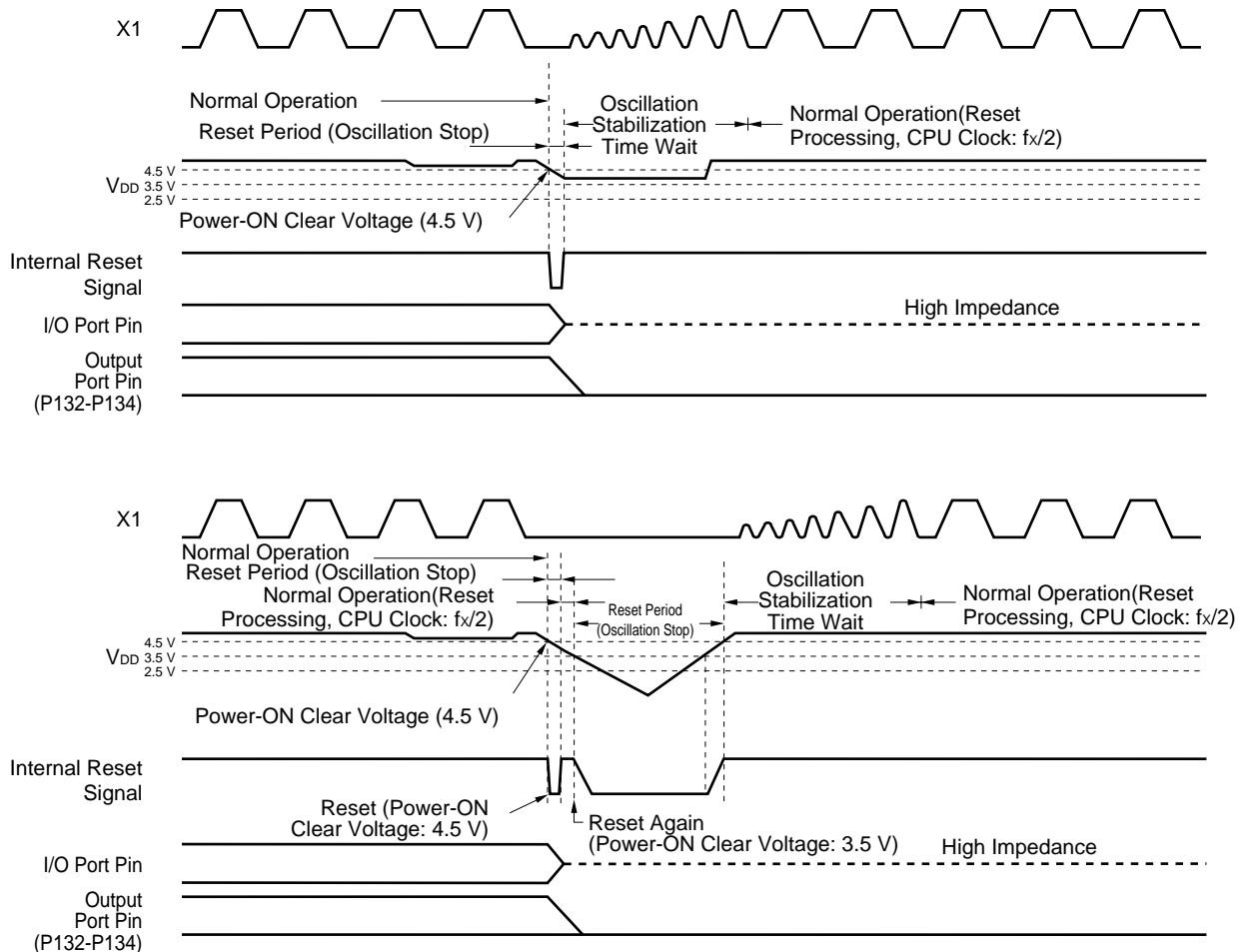
<1> When CPU clock is $f_x/2$ or less<2> When CPU clock is $f_x/2$ 

Table 15-1. Hardware Status after Reset (1/2)

Hardware		Status after Reset
Program counter (PC) ^{Note 1}		The contents of reset vector tables (0000H and 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 to 3, Port 12, 13 (P0 to P3, P12, P13)	00H
	Port 4 to Port 6 (P4 to P6)	Undefined
Port mode register (PM0 to PM3, PM5, PM6, PM12)		FFH
Port mode register 4 (MM)		10H
Processor clock control register (PCC)		04H
Oscillation mode selection register (OSMS)		00H
Oscillation stabilization time select register (OSTS)		04H
8-bit timer/event counter	Timer register (TM1, TM2)	00H
	Compare registers (CR10, CR20)	00H
	Clock select register (TCL1)	00H
	Mode control registers (TMC1)	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 remains unchanged after reset.
 2. The status before reset is retained even after reset in the standby mode.

Table 15-1. Hardware Status after Reset (2/2)

	Hardware	Status after Reset
Serial interface	Clock select register (TCL3)	88H
	Shift registers (SIO1)	Undefined
	Mode registers (CSIM1)	01H
A/D converter	Mode register (ADM)	01H
	Conversion result register (ADCR)	Undefined
	Input select register (ADIS)	00H
Interrupt	Request flag register (IF0L, IF0H)	00H
	Mask flag register (MK0L, MK0H)	FFH
	Priority specification flag register (PR0L, PR0H)	FFH
	External interrupt mode register (INTM0)	00H
	Key return mode register (KRM)	02H
	Sampling clock select register (SCS)	00H
PLL frequency synthesizer	PLL most select register (PLLMD)	00H
	PLL reference mode register (PLLRF)	0FH
	PLL unlock FF judge register (PLLUL)	Retained ^{Note 1}
	PLL data transfer register (PLLNS)	00H
	PLL data register (PLLRL, PLLRH, PLLR0)	Undefined
Frequency counter	IF counter mode select register (IFCMD)	00H
	IF counter gate judge register (IFCJG)	00H
	IF counter control register (IFCR)	00H
	IF counter register (IFC)	0000H
Power-ON clear	POC status register (POCS)	Retained ^{Note 2}

Notes 1. Only reset by power-ON clear becomes undefined.

2. Only reset by power-ON clear becomes 01H.

15.2 Power Failure Detection Function

If reset is effected by means of power-ON clear, bit 0 (POC45) of the POC status register (POCS) is set to 1. If reset is effected by the $\overline{\text{RESET}}$ pin, however, POC45 holds the previous status.

A power failure status can be detected by detecting this POC45 after reset by power-ON clear has been cleared (after program execution has been started from address 0000H).

Figure 15-4. POC Status Register (POCS) Format

Symbol	7	6	5	4	3	2	1	①	Address	After Reset	R/W
POCS	0	0	0	0	0	0	0	POC45	FFBFH	Retained ^{Note}	R&Reset

POC45	Detects Power-ON Clear Occurrence Status
0	Power-ON clear does not occur
1 ^{Note}	Reset is effected by power-ON clear

Note The value of this register is set to 01H only when reset is effected through power-ON clearing.

Remark The values of the special function registers, other than POCS, are the same as the values at reset that is effected by means of power-ON clear.

CHAPTER 16 INSTRUCTION SET

This chapter describes each instruction set of the μ PD178003 subseries as list table. For details of its operation and operation code, refer to the separate document **78K/0 series User's Manual—Instruction (U12326E)**.

16.1 Legends

16.1.1 Operand symbols and description

Operands are written in “Operand” column of each instruction in accordance with the description of the instruction operand symbols (refer to the assembler specifications for detail). When there are two or more descriptions, select one of them. Alphabetic letters in capitals and symbols, #, !, \$ and [] are key words and must be written as they are. Each symbol has the following meaning.

- # : Immediate data specification
- ! : Absolute address specification
- \$: Relative address specification
- [] : Indirect address specification

In the case of immediate data, write an appropriate numeric value or a label. When using a label, be sure to write the #, !, \$, and [] symbols.

For operand register symbols, 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.

Table 16-1. Operand Symbols and Descriptions

Symbol	Description
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-FF1FH Immediate data or labels
saddrp	FE20H-FF1FH Immediate data or labels (even address only)
addr16	0000H-FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions)
addr11	0800H-0FFFH Immediate data or labels
addr5	0040H-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 3-2 Special Function Register List**.

16.1.2 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	: High-order 8 bits and low-order 8 bits of 16-bit register
∧	: Logical product (AND)
∨	: Logical sum (OR)
⊕	: Exclusive logical sum (exclusive OR)
—	: Inverted data
addr16	: 16-bit immediate data or label
jdisp8	: Signed 8-bit data (displacement value)

16.1.3 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

16.2 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	—	$r \leftarrow \text{byte}$			
		saddr, #byte	3	6	7	$(\text{saddr}) \leftarrow \text{byte}$			
		sfr, #byte	3	—	7	$\text{sfr} \leftarrow \text{byte}$			
		A, r Note 3	1	2	—	$A \leftarrow r$			
		r, A Note 3	1	2	—	$r \leftarrow A$			
		A, saddr	2	4	5	$A \leftarrow (\text{saddr})$			
		saddr, A	2	4	5	$(\text{saddr}) \leftarrow A$			
		A, sfr	2	—	5	$A \leftarrow \text{sfr}$			
		sfr, A	2	—	5	$\text{sfr} \leftarrow A$			
		A, !addr16	3	8	9	$A \leftarrow (\text{addr16})$			
		!addr16, A	3	8	9	$(\text{addr16}) \leftarrow A$			
		PSW, #byte	3	—	7	$\text{PSW} \leftarrow \text{byte}$	×	×	×
		A, PSW	2	—	5	$A \leftarrow \text{PSW}$			
		PSW, A	2	—	5	$\text{PSW} \leftarrow A$	×	×	×
		A, [DE]	1	4	5	$A \leftarrow (\text{DE})$			
		[DE], A	1	4	5	$(\text{DE}) \leftarrow A$			
		A, [HL]	1	4	5	$A \leftarrow (\text{HL})$			
		[HL], A	1	4	5	$(\text{HL}) \leftarrow A$			
		A, [HL + byte]	2	8	9	$A \leftarrow (\text{HL} + \text{byte})$			
		[HL + byte], A	2	8	9	$(\text{HL} + \text{byte}) \leftarrow A$			
		A, [HL + B]	1	6	7	$A \leftarrow (\text{HL} + \text{B})$			
		[HL + B], A	1	6	7	$(\text{HL} + \text{B}) \leftarrow A$			
		A, [HL + C]	1	6	7	$A \leftarrow (\text{HL} + \text{C})$			
		[HL + C], A	1	6	7	$(\text{HL} + \text{C}) \leftarrow A$			
	XCH	A, r Note 3	1	2	—	$A \leftrightarrow r$			
		A, saddr	2	4	6	$A \leftrightarrow (\text{saddr})$			
		A, sfr	2	—	6	$A \leftrightarrow \text{sfr}$			
		A, !addr16	3	8	10	$A \leftrightarrow (\text{addr16})$			
		A, [DE]	1	4	6	$A \leftrightarrow (\text{DE})$			
		A, [HL]	1	4	6	$A \leftrightarrow (\text{HL})$			
		A, [HL + byte]	2	8	10	$A \leftrightarrow (\text{HL} + \text{byte})$			
		A, [HL + B]	2	8	10	$A \leftrightarrow (\text{HL} + \text{B})$			
		A, [HL + C]	2	8	1	$A \leftrightarrow (\text{HL} + \text{C})$			

- 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"

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.
 2. This clock cycle applies to internal ROM program.

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"

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{cpu}) selected by the PCC register.
 2. This clock cycle applies to internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	SUB	A, #byte	2	4	—	$A, CY \leftarrow A - \text{byte}$	×	×	×
		saddr, #byte	3	6	8	$(saddr), CY \leftarrow (saddr) - \text{byte}$	×	×	×
		A, r Note 3	2	4	—	$A, CY \leftarrow A - r$	×	×	×
		r, A	2	4	—	$r, CY \leftarrow r - A$	×	×	×
		A, saddr	2	4	5	$A, CY \leftarrow A - (saddr)$	×	×	×
		A, !addr16	3	8	9	$A, CY \leftarrow A - (\text{addr16})$	×	×	×
		A, [HL]	1	4	5	$A, CY \leftarrow A - (HL)$	×	×	×
		A, [HL + byte]	2	8	9	$A, CY \leftarrow A - (HL + \text{byte})$	×	×	×
		A, [HL + B]	2	8	9	$A, CY \leftarrow A - (HL + B)$	×	×	×
		A, [HL + C]	2	8	9	$A, CY \leftarrow A - (HL + C)$	×	×	×
	SUBC	A, #byte	2	4	—	$A, CY \leftarrow A - \text{byte} - CY$	×	×	×
		saddr, #byte	3	6	8	$(saddr), CY \leftarrow (saddr) - \text{byte} - CY$	×	×	×
		A, r Note 3	2	4	—	$A, CY \leftarrow A - r - CY$	×	×	×
		r, A	2	4	—	$r, CY \leftarrow r - A - CY$	×	×	×
		A, saddr	2	4	5	$A, CY \leftarrow A - (saddr) - CY$	×	×	×
		A, !addr16	3	8	9	$A, CY \leftarrow A - (\text{addr16}) - CY$	×	×	×
		A, [HL]	1	4	5	$A, CY \leftarrow A - (HL) - CY$	×	×	×
		A, [HL + byte]	2	8	9	$A, CY \leftarrow A - (HL + \text{byte}) - CY$	×	×	×
		A, [HL + B]	2	8	9	$A, CY \leftarrow A - (HL + B) - CY$	×	×	×
		A, [HL + C]	2	8	9	$A, CY \leftarrow A - (HL + C) - CY$	×	×	×
	AND	A, #byte	2	4	—	$A \leftarrow A \wedge \text{byte}$	×		
		saddr, #byte	3	6	8	$(saddr) \leftarrow (saddr) \wedge \text{byte}$	×		
		A, r Note 3	2	4	—	$A \leftarrow A \wedge r$	×		
		r, A	2	4	—	$r \leftarrow r \wedge A$	×		
		A, saddr	2	4	5	$A \leftarrow A \wedge (saddr)$	×		
		A, !addr16	3	8	9	$A \leftarrow A \wedge (\text{addr16})$	×		
		A, [HL]	1	4	5	$A \leftarrow A \wedge [HL]$	×		
		A, [HL + byte]	2	8	9	$A \leftarrow A \wedge [HL + \text{byte}]$	×		
		A, [HL + B]	2	8	9	$A \leftarrow A \wedge [HL + B]$	×		
		A, [HL + C]	2	8	9	$A \leftarrow A \wedge [HL + C]$	×		

- 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"

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.
 2. This clock cycle applies to internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	OR	A, #byte	2	4	—	$A \leftarrow A \vee \text{byte}$	×		
		saddr, #byte	3	6	8	$(\text{saddr}) \leftarrow (\text{saddr}) \vee \text{byte}$	×		
		A, r Note 3	2	4	—	$A \leftarrow A \vee r$	×		
		r, A	2	4	—	$r \leftarrow r \vee A$	×		
		A, saddr	2	4	5	$A \leftarrow A \vee (\text{saddr})$	×		
		A, !addr16	3	8	9	$A \leftarrow A \vee (\text{addr16})$	×		
		A, [HL]	1	4	5	$A \leftarrow A \vee (\text{HL})$	×		
		A, [HL + byte]	2	8	9	$A \leftarrow A \vee (\text{HL} + \text{byte})$	×		
		A, [HL + B]	2	8	9	$A \leftarrow A \vee (\text{HL} + B)$	×		
		A, [HL + C]	2	8	9	$A \leftarrow A \vee (\text{HL} + C)$	×		
	XOR	A, #byte	2	4	—	$A \leftarrow A \nabla \text{byte}$	×		
		saddr, #byte	3	6	8	$(\text{saddr}) \leftarrow (\text{saddr}) \nabla \text{byte}$	×		
		A, r Note 3	2	4	—	$A \leftarrow A \nabla r$	×		
		r, A	2	4	—	$r \leftarrow r \nabla A$	×		
		A, saddr	2	4	5	$A \leftarrow A \nabla (\text{saddr})$	×		
		A, !addr16	3	8	9	$A \leftarrow A \nabla (\text{addr16})$	×		
		A, [HL]	1	4	5	$A \leftarrow A \nabla (\text{HL})$	×		
		A, [HL + byte]	2	8	9	$A \leftarrow A \nabla (\text{HL} + \text{byte})$	×		
		A, [HL + B]	2	8	9	$A \leftarrow A \nabla (\text{HL} + B)$	×		
		A, [HL + C]	2	8	9	$A \leftarrow A \nabla (\text{HL} + C)$	×		
	CMP	A, #byte	2	4	—	$A - \text{byte}$	×	×	×
		saddr, #byte	3	6	8	$(\text{saddr}) - \text{byte}$	×	×	×
		A, r Note 3	2	4	—	$A - r$	×	×	×
		r, A	2	4	—	$r - A$	×	×	×
		A, saddr	2	4	5	$A - (\text{saddr})$	×	×	×
		A, !addr16	3	8	9	$A - (\text{addr16})$	×	×	×
		A, [HL]	1	4	5	$A - (\text{HL})$	×	×	×
		A, [HL + byte]	2	8	9	$A - (\text{HL} + \text{byte})$	×	×	×
		A, [HL + B]	2	8	9	$A - (\text{HL} + B)$	×	×	×
		A, [HL + C]	2	8	9	$A - (\text{HL} + C)$	×	×	×

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"

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

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 \leftarrow AX + word	×	×	×
	SUBW	AX, #word	3	6	—	AX, CY \leftarrow AX – word	×	×	×
	CMPW	AX, #word	3	6	—	AX – word	×	×	×
Multiply/divide	MULU	X	2	16	—	AX \leftarrow A \times X			
	DIVUW	C	2	25	—	AX (Quotient), C (Remainder) \leftarrow AX \div C			
Increment/decrement	INC	r	1	2	—	$r \leftarrow r + 1$	×	×	
		saddr	2	4	6	$(saddr) \leftarrow (saddr) + 1$	×	×	
	DEC	r	1	2	—	$r \leftarrow r - 1$	×	×	
		saddr	2	4	6	$(saddr) \leftarrow (saddr) - 1$	×	×	
	INCW	rp	1	4	—	$rp \leftarrow rp + 1$			
	DECW	rp	1	4	—	$rp \leftarrow rp - 1$			
Rotate	ROR	A, 1	1	2	—	$(CY, A_7 \leftarrow A_0, A_{m-1} \leftarrow A_m) \times 1 \text{ time}$			×
	ROL	A, 1	1	2	—	$(CY, A_0 \leftarrow A_7, A_{m+1} \leftarrow A_m) \times 1 \text{ time}$			×
	RORC	A, 1	1	2	—	$(CY \leftarrow A_0, A_7 \leftarrow CY, A_{m-1} \leftarrow A_m) \times 1 \text{ time}$			×
	ROLC	A, 1	1	2	—	$(CY \leftarrow A_7, A_0 \leftarrow CY, A_{m+1} \leftarrow A_m) \times 1 \text{ time}$			×
	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 manipulate	MOV1	CY, saddr.bit	3	6	7	$CY \leftarrow (saddr.bit)$			×
		CY, sfr.bit	3	—	7	$CY \leftarrow sfr.bit$			×
		CY, A.bit	2	4	—	$CY \leftarrow A.bit$			×
		CY, PSW.bit	3	—	7	$CY \leftarrow PSW.bit$			×
		CY, [HL].bit	2	6	7	$CY \leftarrow (HL).bit$			×
		saddr.bit, CY	3	6	8	$(saddr.bit) \leftarrow CY$			
		sfr.bit, CY	3	—	8	$sfr.bit \leftarrow CY$			
		A.bit, CY	2	4	—	$A.bit \leftarrow CY$			
		PSW.bit, CY	3	—	8	$PSW.bit \leftarrow CY$	×	×	
		[HL].bit, CY	2	6	8	$(HL).bit \leftarrow CY$			

- 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

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.
 2. This clock cycle applies to internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Bit manipulate	AND1	CY, saddr.bit	3	6	7	$CY \leftarrow CY \wedge (saddr.bit)$			×
		CY, sfr.bit	3	—	7	$CY \leftarrow CY \wedge sfr.bit$			×
		CY, A.bit	2	4	—	$CY \leftarrow CY \wedge A.bit$			×
		CY, PSW.bit	3	—	7	$CY \leftarrow CY \wedge PSW.bit$			×
		CY, [HL].bit	2	6	7	$CY \leftarrow CY \wedge (HL).bit$			×
	OR1	CY, saddr.bit	3	6	7	$CY \leftarrow CY \vee (saddr.bit)$			×
		CY, sfr.bit	3	—	7	$CY \leftarrow CY \vee sfr.bit$			×
		CY, A.bit	2	4	—	$CY \leftarrow CY \vee A.bit$			×
		CY, PSW.bit	3	—	7	$CY \leftarrow CY \vee PSW.bit$			×
		CY, [HL].bit	2	6	7	$CY \leftarrow CY \vee (HL).bit$			×
	XOR1	CY, saddr.bit	3	6	7	$CY \leftarrow CY \oplus (saddr.bit)$			×
		CY, sfr.bit	3	—	7	$CY \leftarrow CY \oplus sfr.bit$			×
		CY, A.bit	2	4	—	$CY \leftarrow CY \oplus A.bit$			×
		CY, PSW.bit	3	—	7	$CY \leftarrow CY \oplus PSW.bit$			×
		CY, [HL].bit	2	6	7	$CY \leftarrow CY \oplus (HL).bit$			×
	SET1	saddr.bit	2	4	6	$(saddr.bit) \leftarrow 1$			
		sfr.bit	3	—	8	$sfr.bit \leftarrow 1$			
		A.bit	2	4	—	$A.bit \leftarrow 1$			
		PSW.bit	2	—	6	$PSW.bit \leftarrow 1$	×	×	×
		[HL].bit	2	6	8	$(HL).bit \leftarrow 1$			
	CLR1	saddr.bit	2	4	6	$(saddr.bit) \leftarrow 0$			
		sfr.bit	3	—	8	$sfr.bit \leftarrow 0$			
		A.bit	2	4	—	$A.bit \leftarrow 0$			
		PSW.bit	2	—	6	$PSW.bit \leftarrow 0$	×	×	×
		[HL].bit	2	6	8	$(HL).bit \leftarrow 0$			
	SET1	CY	1	2	—	$CY \leftarrow 1$			1
	CLR1	CY	1	2	—	$CY \leftarrow 0$			0
	NOT1	CY	1	2	—	$CY \leftarrow \overline{CY}$			×

- 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

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.
 2. This clock cycle applies to internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Call/return	CALL	!addr16	3	7	–	$(SP - 1) \leftarrow (PC + 3)_H$, $(SP - 2) \leftarrow (PC + 3)_L$, $PC \leftarrow \text{addr16}$, $SP \leftarrow SP - 2$			
	CALLF	!addr11	2	5	–	$(SP - 1) \leftarrow (PC + 2)_H$, $(SP - 2) \leftarrow (PC + 2)_L$, $PC_{15:11} \leftarrow 00001$, $PC_{10:0} \leftarrow \text{addr11}$, $SP \leftarrow SP - 2$			
	CALLT	[addr5]	1	6	–	$(SP - 1) \leftarrow (PC + 1)_H$, $(SP - 2) \leftarrow (PC + 1)_L$, $PC_H \leftarrow (00000000, \text{addr5} + 1)$, $PC_L \leftarrow (00000000, \text{addr5})$, $SP \leftarrow SP - 2$			
	BRK		1	6	–	$(SP - 1) \leftarrow PSW$, $(SP - 2) \leftarrow (PC + 1)_H$, $(SP - 3) \leftarrow (PC + 1)_L$, $PC_H \leftarrow (003FH)$, $PC_L \leftarrow (003EH)$, $SP \leftarrow SP - 3$, $IE \leftarrow 0$			
	RET		1	6	–	$PC_H \leftarrow (SP + 1)$, $PC_L \leftarrow (SP)$, $SP \leftarrow SP + 2$			
	RETI		1	6	–	$PC_H \leftarrow (SP + 1)$, $PC_L \leftarrow (SP)$, $PSW \leftarrow (SP + 2)$, $SP \leftarrow SP + 3$, $NMIS \leftarrow 0$	R	R	R
	RETB		1	6	–	$PC_H \leftarrow (SP + 1)$, $PC_L \leftarrow (SP)$, $PSW \leftarrow (SP + 2)$, $SP \leftarrow SP + 3$	R	R	R
Stack manipulate	PUSH	PSW	1	2	–	$(SP - 1) \leftarrow PSW$, $SP \leftarrow SP - 1$			
		rp	1	4	–	$(SP - 1) \leftarrow rp_H$, $(SP - 2) \leftarrow rp_L$, $SP \leftarrow SP - 2$			
	POP	PSW	1	2	–	$PSW \leftarrow (SP)$, $SP \leftarrow SP + 1$	R	R	R
		rp	1	4	–	$rp_H \leftarrow (SP + 1)$, $rp_L \leftarrow (SP)$, $SP \leftarrow SP + 2$			
	MOVW	SP, #word	4	–	10	$SP \leftarrow \text{word}$			
		SP, AX	2	–	8	$SP \leftarrow AX$			
		AX, SP	2	–	8	$AX \leftarrow SP$			
Unconditional branch	BR	!addr16	3	6	–	$PC \leftarrow \text{addr16}$			
		\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$			
		AX	2	8	–	$PC_H \leftarrow A$, $PC_L \leftarrow X$			
Conditional branch	BC	\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $CY = 1$			
	BNC	\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $CY = 0$			
	BZ	\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $Z = 1$			
	BNZ	\$addr16	2	6	–	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $Z = 0$			

- Notes**
1. When the internal high-speed RAM area is accessed or instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.
 2. This clock cycle applies to internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Conditional branch	BT	saddr.bit, \$addr16	3	8	9	$PC \leftarrow PC + 3 + jdisp8$ if(saddr.bit) = 1			
		sfr.bit, \$addr16	4	–	11	$PC \leftarrow PC + 4 + jdisp8$ if sfr.bit = 1			
		A.bit, \$addr16	3	8	–	$PC \leftarrow PC + 3 + jdisp8$ if A.bit = 1			
		PSW.bit, \$addr16	3	–	9	$PC \leftarrow PC + 3 + jdisp8$ if PSW.bit = 1			
		[HL].bit, \$addr16	3	10	11	$PC \leftarrow PC + 3 + jdisp8$ if (HL).bit = 1			
	BF	saddr.bit, \$addr16	4	10	11	$PC \leftarrow PC + 4 + jdisp8$ if(saddr.bit) = 0			
		sfr.bit, \$addr16	4	–	11	$PC \leftarrow PC + 4 + jdisp8$ if sfr.bit = 0			
		A.bit, \$addr16	3	8	–	$PC \leftarrow PC + 3 + jdisp8$ if A.bit = 0			
		PSW.bit, \$addr16	4	–	11	$PC \leftarrow PC + 4 + jdisp8$ if PSW. bit = 0			
		[HL].bit, \$addr16	3	10	11	$PC \leftarrow PC + 3 + jdisp8$ if (HL).bit = 0			
	BTCLR	saddr.bit, \$addr16	4	10	12	$PC \leftarrow PC + 4 + jdisp8$ if(saddr.bit) = 1 then reset(saddr.bit)			
		sfr.bit, \$addr16	4	–	12	$PC \leftarrow PC + 4 + jdisp8$ if sfr.bit = 1 then reset sfr.bit			
		A.bit, \$addr16	3	8	–	$PC \leftarrow PC + 3 + jdisp8$ if A.bit = 1 then reset A.bit			
		PSW.bit, \$addr16	4	–	12	$PC \leftarrow PC + 4 + jdisp8$ if PSW.bit = 1 then reset PSW.bit	×	×	×
		[HL].bit, \$addr16	3	10	12	$PC \leftarrow PC + 3 + jdisp8$ if (HL).bit = 1 then reset (HL).bit			
	DBNZ	B, \$addr16	2	6	–	$B \leftarrow B - 1$, then $PC \leftarrow PC + 2 + jdisp8$ if $B \neq 0$			
		C, \$addr16	2	6	–	$C \leftarrow C - 1$, then $PC \leftarrow PC + 2 + jdisp8$ if $C \neq 0$			
		saddr, \$addr16	3	8	10	(saddr) \leftarrow (saddr) – 1, then $PC \leftarrow PC + 3 + jdisp8$ if (saddr) $\neq 0$			
CPU control	SEL	RBn	2	4	–	RBS1, $0 \leftarrow n$			
	NOP		1	2	–	No Operation			
	EI		2	–	6	$IE \leftarrow 1$ (Enable Interrupt)			
	DI		2	–	6	$IE \leftarrow 0$ (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

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.
 2. This clock cycle applies to internal ROM program.

16.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

Second Operand First Operand	#byte	A	rNote	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 XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH XCH	MOV XCH XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH XCH	MOV XCH XCH	MOV XCH XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH 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 1st Operand	#word	AX	rp ^{Note}	sfrp	saddrp	!addr16	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW ^{Note}						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
SP	MOVW	MOVW						

Note Only when rp = BC, DE, HL**(3) Bit manipulation instructions**

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

Second Operand First Operand	A.bit	sfr.bit	saddr.bit	PSW.bit	[HL].bit	CY	\$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call/instructions/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

Second Operand First Operand	AX	!addr16	!addr11	[addr5]	\$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruction					BT BF BTCLR DBNZ

(5) Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

[MEMO]

APPENDIX A DIFFERENCES BETWEEN μ PD178002, 178003 AND μ PD178P018A

The differences between μ PD178002, 178003 and μ PD178P018A are shown in Table A-1.

For the μ PD178P018A, refer to **μ PD178018A Subseries User's Manual**.

Table A-1. Differences between μ PD178002, 178003 and μ PD178P018A (1/2)

Part Number		μ PD178002	μ PD178003	μ PD178P018A
Item				
Internal memory	ROM (ROM structure)	16 Kbytes (mask ROM)	24 Kbytes (mask ROM)	60 Kbytes (one-time PROM)
	High-speed ROM	512 bytes		1024 bytes
	Buffer RAM	None		32 bytes
	Extended RAM	None		2048 bytes
A/D converter	Number of channels	3 channels		6 channels
	Starting A/D conversion	<ul style="list-style-type: none"> • Software start 		<ul style="list-style-type: none"> • Software start • Hardware start
D/A converter (PWM output)		None		8-/9-bit resolution \times 3 channels (shared by 8-bit timer)
Serial interface		1 channel <ul style="list-style-type: none"> • 3-wire serial I/O mode: 1 channel 		2 channels <ul style="list-style-type: none"> • 3-wire serial I/O /SBI/ 2-wire serial I/O/I²C bus^{Note} mode selectable: 1 channel • 3-wire serial I/O mode (with automatic transmit/receive function up to 32 bytes): 1 channel
Timer		3 channels <ul style="list-style-type: none"> • Basic timer (timer carry FF (10 Hz)): 1 channel • 8-bit timer/event counter: 2 channels 		5 channels <ul style="list-style-type: none"> • Basic timer (timer carry FF (10 Hz)): 1 channel • 8-bit timer/event counter: 2 channels • 8-bit timer (D/A converter: PWM output) : 1 channel • Watchdog timer:
Vectored interrupt sources	Non-maskable	None		Internal: 1
	Maskable	Internal: 5, external: 2		Internal: 8, external: 7
	Software	1		
PLL frequency synthesizer	EO1 pin output circuit	Buffer type (high-impedance not supported)		Buffer type (high-impedance supported)

Note When using the I²C bus mode (including when this mode is implemented by software without using the internal hardware), consult NEC when you place an order for mask.

[MEMO]

APPENDIX B DEVELOPMENT TOOLS

The following development tools are available for the development of systems that employ the μ PD178003 Subseries.

Figure B-1 shows the development tool configuration.

- ★
 - **Support for PC98-NX series**

Unless otherwise specified, products compatible with IBM PC/AT™ computers are compatible with PC98-NX series computers. When using PC98-NX series, refer to the description for IBM PC/AT computers.
- ★
 - **Windows**

Unless otherwise specified, "Windows" means the following OSs.

 - Windows 3.1
 - Windows95
 - WindowsNT™ Ver. 4.0

★

Figure B-1. Development Tool Configuration (1/2)

(1) When using the in-circuit emulator IE-78K0-NS

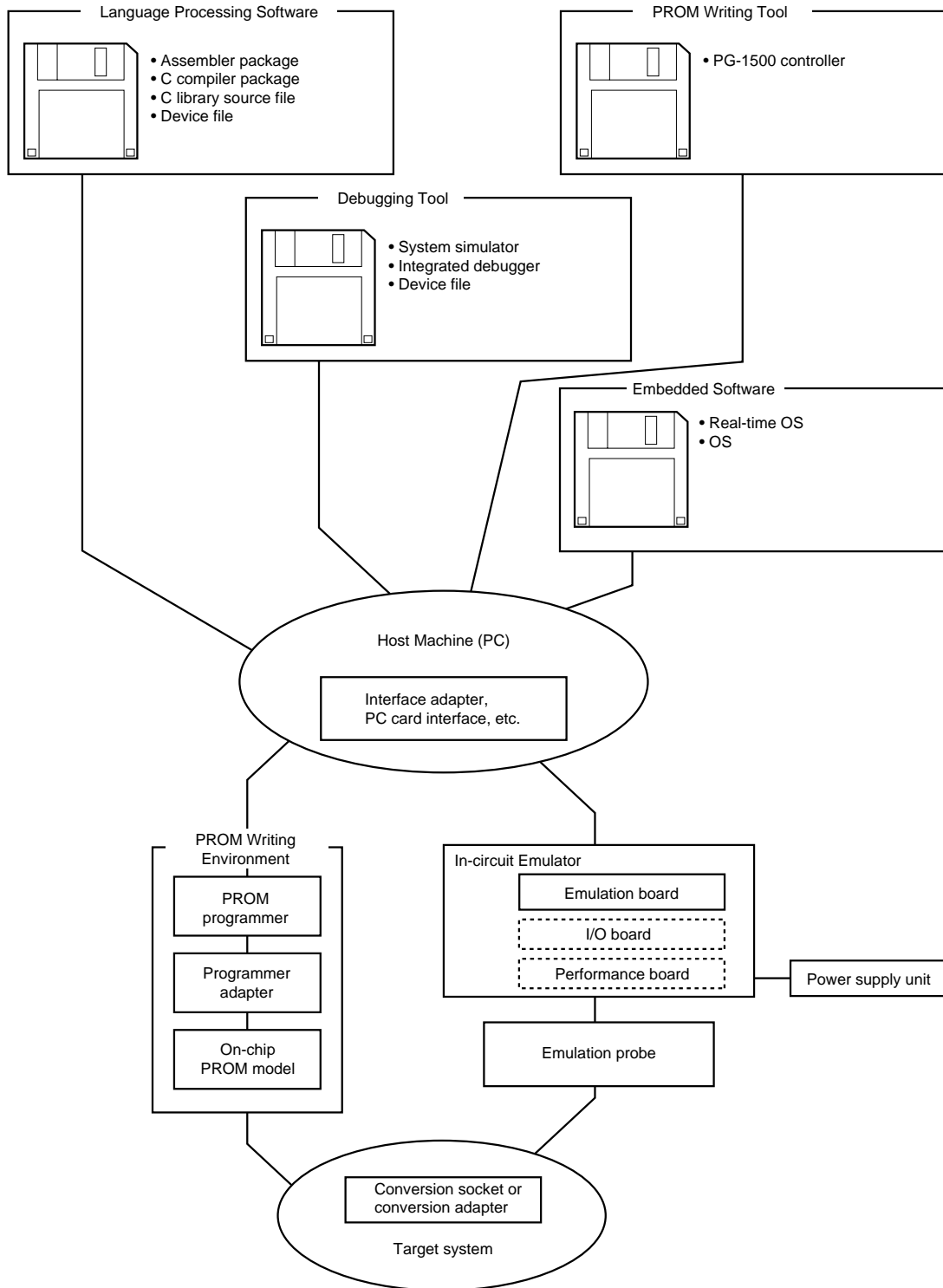
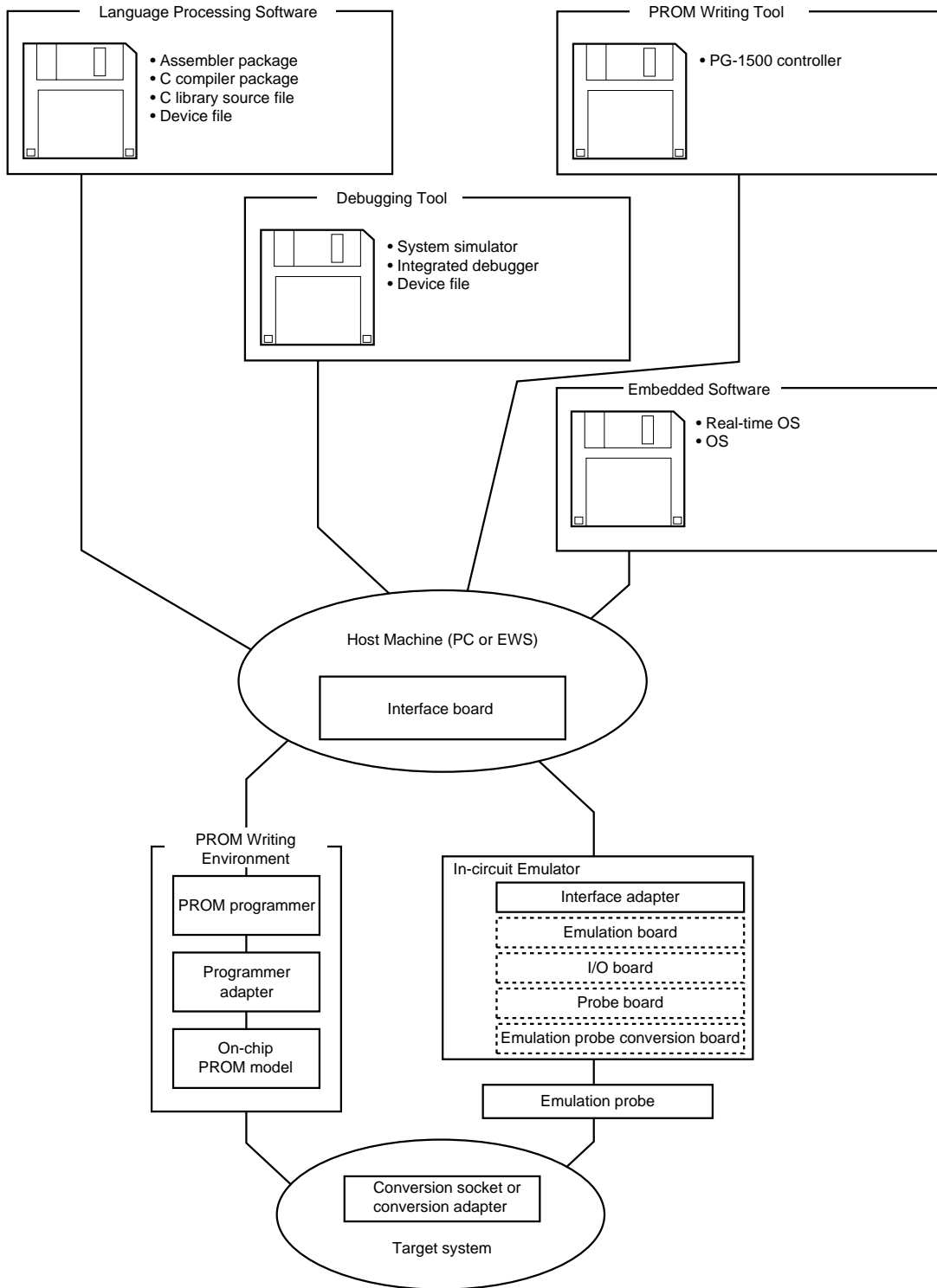


Figure B-1. Development Tool Configuration (2/2)

(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 **B.3.1. Hardware**.

★ B.1 Language Processing Software

RA78K0 Assembler Package	<p>This assembler converts programs written in mnemonics into an object codes executable with a microcomputer.</p> <p>Further, this assembler is provided with functions capable of automatically creating symbol tables and branch instruction optimization.</p> <p>This assembler should be used in combination with an optical device file (DF178018).</p> <p><Precaution when using RA78K0 in PC environment></p> <p>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.</p>
	Part Number: μ SxxxxRA78K0
CC78K0 C Compiler Package	<p>This compiler converts programs written in C language into object codes executable with a microcomputer.</p> <p>This compiler should be used in combination with an optical assembler package and device file.</p> <p><Precaution when using CC78K0 in PC environment></p> <p>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.</p>
	Part Number: μ SxxxxCC78K0
DF178018 ^{Note} Device File	<p>This file contains information peculiar to the device.</p> <p>This device file should be used in combination with an optical tool (RA78K0, CC78K0, SM78K0, ID78K0-NS, and ID78K0).</p> <p>Corresponding OS and host machine differ depending on the tool to be used with.</p>
	Part Number: μ SxxxxDF178018
CC78K0-L C Library Source File	<p>This is a source file of functions configuring the object library included in the C compiler package.</p> <p>This file is required to match the object library included in C compiler package to the customer's specifications.</p> <p>It does not depend on the operating environment because it is a source file.</p>
	Part Number: μ SxxxxCC78K0-L

Note The DF178018 can be used in common with the RA78K0, CC78K0, SM78K0, ID78K0-NS, and ID78K0.


Remark xxxx in the part number differs depending on the host machine and OS used.

μSxxxxRA78K0

μSxxxxCC78K0

μSxxxxDF178018

μSxxxxCC78K0-L



xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version) Note	3.5-inch 2HD FD
AB13	IBM PC/AT 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 be operated in DOS environment.

★ B.2 PROM Programming Tools

B.2.1 Hardware

PG-1500 PROM Programmer	This is a PROM programmer capable of programming the single-chip microcontroller incorporating PROM by manipulating from the stand-alone or host machine through connection of an optional PROM programmer adapter and attached board. It can also program representative PROMs ranging from 256K bits to 4M bits.
PA-178P018GC PA-178P018KK-T PROM Programmer Adapter	PROM programmer adapter for the μ PD178P018. Used connected to the PG-1500. <ul style="list-style-type: none"> PA-178P018GC : 80-pin plastic QFP (GC-3B9 type) PA-178P018KK-T : 80-pin ceramic WQFN (KK-T type)

B.2.2 Software

PG-1500 Controller	The PG-1500 is controlled in the host machine through connection with the host machine and PG-1500 via serial and parallel interfaces. The PG-1500 controller is a DOS-based application. It should be used in the DOS Prompt when using in Windows.
	Part Number : μ SxxxxPG1500

Remark xxxx of the part number differs depending on the host machine and OS used. Refer to the table below.

μ Sxxxx PG1500

xxxx	Host Machine	OS	Supply Medium
5A13	PC-9800 series	MS-DOS (ver. 3.30 to 6.2) ^{Note}	3.5-inch 2HD
7B13	IBM PC/AT compatible machine	Refer to B.4.	3.5-inch 2HD

Note The task swap function is not available with this software though the function is provided in MS-DOS version 5.0 or later.



B.3 Debugging Tools

B.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-78K0-NS-PA Performance Board	This board is used for extending the IE-78K0-NS functions. With the addition of this board, the addition of a coverage function, enhancement of tracer and timer functions, and other such debugging function enhancements are possible.
IE-70000-98-IF-C Interface Adapter	This adapter is required when using the PC-9800 Series computer (except notebook PC) as the IE-78K0-NS host machine (C bus compatible).
IE-70000-CD-IF-A PC Card Interface	This is PC card and interface cable required when using the notebook-type computer as the IE-78K0-NS host machine (PCMCIA socket compatible).
IE-70000-PC-IF-C Interface Adapter	This adapter is required when using the IBM PC compatible computers as the IE-78K0-NS host machine (ISA bus compatible).
IE-70000-PCI-IF Interface Adapter	This adapter is required when using a PC with a PCI bus as the IE-78K0-NS host machine.
IE-178018-NS-EM1 Emulation Board	This board emulates the operations of the peripheral hardware peculiar to a device. It should be used in combination with an in-circuit emulator.
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-3B9 type).
EV-9200GC-80 Conversion Socket (Refer to Figures B-2 and B-3)	This conversion socket connects the NP-64GC to the target system board designed to mount a 80-pin plastic QFP (GC-3B9 type).

- Remarks**
1. NP-80GC is a product of Naito Densai Machida Mfg. Co., Ltd. (Phone: (044) 822-3813)
Consult NEC distributor on purchasing this product.
 2. EV-9200GC-80 is sold in five units.

B.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 computer (except notebook PC) as the IE-78001-R-A host machine (C bus compatible).
IE-70000-PC-IF-C Interface Adapter	This adapter is required when using the IBM PC/AT compatible computers as the IE-78001-R-A host machine (ISA bus compatible).
★ IE-70000-PCI-IF Interface Adapter	This adapter is required when using a PC with a PCI bus as the IE-78001-R-A host machine.
IE-78000-R-SV3 Interface Adapter	This is adapter and cable required when using an EWS computer as the IE-78001-R-A host machine, and is used connected to the board in the IE-78000-R-A.
IE-178018-NS-EM1 Emulation Board	This board emulates the operations of the peripheral hardware peculiar to a device. It should be used in combination with an in-circuit emulator and emulation conversion board.
IE-78K0-R-EX1 Emulation Probe Conversion Board	This board is required when using the IE-178018-NS-EM1 on the IE-78001-R-A.
IE-178018-R-EM Emulation Board	This board emulates the operations of the peripheral hardware peculiar to a device. It should be used in combination with an in-circuit emulator and emulation probe conversion board.
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-3B9 type).
EV-9200GC-80 Conversion Socket (Refer to Figures B-2 and B-3)	This conversion socket connects the EP-78230GC-R to the target system board designed to mount a 80-pin plastic QFP (GC-3B9 type).
EV-9900	This is a jig used to for removing the μ PD178P018KK-T.

Remark EV-9200GC-80 is sold in five units.

B.3.2 Software (1/2)

SM78K0 System Simulator	<p>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.</p> <p>This simulator runs on Windows.</p> <p>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.</p> <p>The SM78K0 should be used in combination with the optical device file (DF178018).</p>
	Part Number: μ SxxxxSM78K0

Remark xxxx in the part number differs depending on the host machine and OS used.

μ SxxxxSM78K0

xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB13		Windows (English version)	

B.3.2 Software (2/2)

★ ID78K0-NS Integrated Debugger (supporting in-circuit emulator IE-78K0-NS)	<p>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 programs, and thus trace results can be displayed on screen in C 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.</p> <p>It should be used in combination with the optical device file.</p>
ID78K0 Integrated Debugger (supporting in-circuit emulator IE-78001-R-A)	
	Part Number: μ SxxxxID78K0-NS, μ SxxxxID78K0

Remark xxxx in the part number differs depending on the host machine and OS used.

μ SxxxxID78K0-NS

xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB13		Windows (English version)	

μ SxxxxID78K0

xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT 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

B.4 System Upgrade 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 (under development).

Table B-1. System Upgrade Method from Former In-circuit Emulator for 78K/0 Series to the IE-78001-R-A

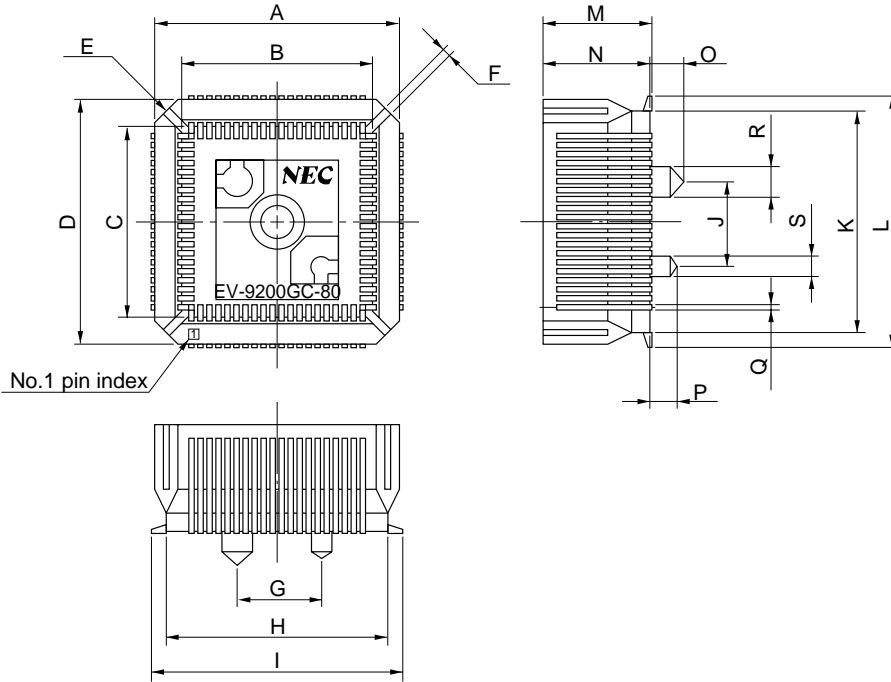
In-circuit Emulator Owned	In-circuit Emulator Cabinet System Upgrade ^{Note}	Board to be Purchased
IE-78000-R	Required	IE-78001-R-BK
IE-78000-R-A	Not required	

Note For system upgrade of a cabinet, send your in-circuit emulator to NEC.

Drawing for Conversion Socket (EV-9200GC-80) Package and Recommended Board Mounting Pattern

Figure B-2. EV-9200GC-80 Package Drawing (For Reference Only)

Based on EV-9200GC-80
(1) Package drawing (in mm)

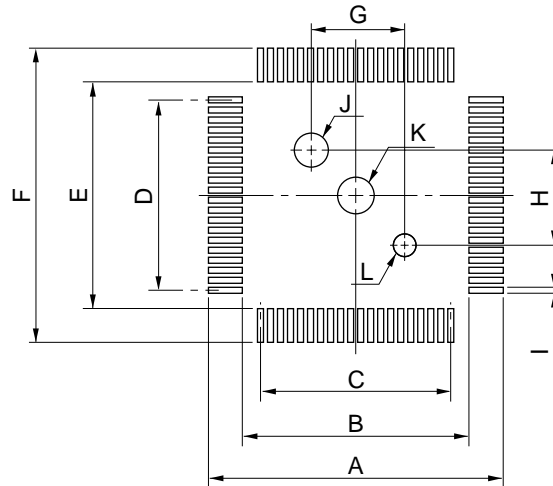


EV-9200GC-80-G1E

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
N	8.0	0.315
O	2.5	0.098
P	2.0	0.079
Q	0.35	0.014
R	φ2.3	φ0.091
S	φ1.5	φ0.059

Figure B-3. Recommended Board Mounting Pattern EV-9200GC-80 (For Reference Only)

Based on EV-9200GC-80
(2) Pad drawing (in mm)



EV-9200GC-80-P1E

ITEM	MILLIMETERS	INCHES
A	19.7	0.776
B	15.0	0.591
C	$0.65 \pm 0.02 \times 19 = 12.35 \pm 0.05$	$0.026^{+0.001}_{-0.002} \times 0.748 = 0.486^{+0.003}_{-0.002}$
D	$0.65 \pm 0.02 \times 19 = 12.35 \pm 0.05$	$0.026^{+0.001}_{-0.002} \times 0.748 = 0.486^{+0.003}_{-0.002}$
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	$\phi 2.36 \pm 0.03$	$\phi 0.093^{+0.001}_{-0.002}$
K	$\phi 2.3$	$\phi 0.091$
L	$\phi 1.57 \pm 0.03$	$\phi 0.062^{+0.001}_{-0.002}$

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).

[MEMO]

APPENDIX C EMBEDDED SOFTWARE

For efficient development and maintenance of the μ PD178003 subseries, the following embedded products are available.

Real-Time OS (1/2)

RX78K/0 Real-time OS	<p>RX78K/0 is a real-time OS conforming to the μITRON specifications.</p> <p>Tool (configurator) for generating nucleus of RX78K/0 and plural information tables is supplied.</p> <p>Used in combination with an optical assembler package (RA78K0) and device file (DF178018).</p> <p><Precaution when using RX78K/0 in PC environment></p> <p>The real-time OS is a DOS-based application. It should be used in the DOS Prompt when using in Windows.</p>
	Part Number: μ SxxxxRX78013- $\Delta\Delta\Delta\Delta$

Caution When purchasing the RX78K/0, fill in the purchase application form in advance and sign the User Agreement.

Remark xxxx and $\Delta\Delta\Delta$ in the part number differ depending on the host machine and OS used.

μ SxxxxRX78013- $\Delta\Delta\Delta\Delta$

$\Delta\Delta\Delta\Delta$	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

xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB10		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

Note Can also be operated in DOS environment.

Real-Time OS (2/2)

MX78K0 OS	<p>MX78K0 is an OS for μTRON specification subsets. A nucleus for the MX78K0 is also included as a companion product.</p> <p>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.</p> <p><Precaution when using MX78K0 in PC environment></p> <p>The MX78K0 is a DOS-based application. It should be used in the DOS Prompt when using in Windows.</p>
	Part Number: μ SxxxxMX78K0- $\Delta\Delta\Delta$

Remark xxxx and $\Delta\Delta\Delta$ in the part number differ depending on the host machine and OS used.

μ SxxxxMX78K0- $\Delta\Delta\Delta$

$\Delta\Delta\Delta$	Product Outline	Maximum Number for Use in Mass Production
001	Evaluation object	Use in preproduction stages.
xx	Mass-production object	Use in mass production stages.
S01	Source program	Only the users who purchased mass-production objects are allowed to purchase this program.

xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	Windows (Japanese version) ^{Note}	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version) ^{Note}	3.5-inch 2HC FD
BB10		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 DOS environment.

APPENDIX D REVISION HISTORY

The revision history is described below. The “Applied to” column indicates the chapter in each edition.

Edition	Major Revisions from Previous Edition	Applied to:
2nd edition	Number of A/D converter channels changed from six to three.	Throughout
	2.1.2 Pins other than port pins Table 2-1. I/O Circuit Type of Each Circuit I/O circuit type of P60-P63 changed from 13 to 13-G. Figure 2-1. Pin Input/Output Circuit List (2/2) Type 13 changed to 13-G.	CHAPTER 2 PIN FUNCTION
	A/D Converter Configuration Figure 9-1. A/D Converter Block Diagram ADIS2 and ADM3 changed to 0. Figure 9-2. A/D Converter Mode Register (ADM) Format Bit 3 modified. Figure 9-3. A/D Converter Input Select Register (ADIS) Format Bit 2 modified.	CHAPTER 9 A/D CONVERTER
	Supported OS <ul style="list-style-type: none"> • WindowsNT Ver.4.0 added • Solaris added • Version of HP-UX changed Figure B-1. Development Tool Configuration (1/2) (1) When using the in-circuit emulator IE-78K0-NS In-circuit emulator modified B.3.1 Hardware <ul style="list-style-type: none"> • The following products added Interface adapter IE-70000-PCI-IF Performance board IE-78K0-NS-PA <ul style="list-style-type: none"> • The following products developed In-circuit emulator IE-78K0-NS Interface adapter IE-70000-98-IF-C, IE-70000-PC-IF-C PC card interface IE-70000-CD-IF-A Emulation board IE-178018-NS-EM1 Emulation probe conversion board IE-78K0-R-EX B.3.2 Software <ul style="list-style-type: none"> • The following products developed Integrated debugger ID78K0-NS	APPENDIX B DEVELOPMENT TOOLS
	Supported OS <ul style="list-style-type: none"> • WindowsNT Ver.4.0 added • Solaris added • Version of HP-UX changed 	APPENDIX C EMBEDDED SOFTWARE

[MEMO]

Facsimile Message

From:

Name

Company

Tel.

FAX

Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

Thank you for your kind support.

North America

NEC Electronics Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Hong Kong, Philippines, Oceania

NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Asian Nations except Philippines

NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Europe

NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

Korea

NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: 02-528-4411

Japan

NEC Semiconductor Technical Hotline
Fax: 044-548-7900

South America

NEC do Brasil S.A.
Fax: +55-11-6465-6829

Taiwan

NEC Electronics Taiwan Ltd.
Fax: 02-2719-5951

I would like to report the following error/make the following suggestion:

Document title: _____

Document number: _____ Page number: _____

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Technical Accuracy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>