To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

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

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

RENESAS

User's Manual

Phase-out/Discontinued

μPD789014 Subseries

8-bit Single-chip Microcontrollers

μPD789011
μPD789012
μPD78P9014

Document No. U11187EJ3V0UMJ1 (3rd edition)
Date Published September 2000 N CP(K)

[MEMO]

NOTES FOR CMOS DEVICES

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.

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 current as of September, 2000. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC's data sheets or data books, etc., for the most up-to-date specifications of NEC semiconductor products. Not all products and/or types are available in every country. Please check with an NEC sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without prior written consent of NEC. NEC assumes no responsibility for any errors that may appear in this document.
- NEC does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC semiconductor products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC 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 customer's equipment shall be done under the full responsibility of customer. NEC assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC endeavours to enhance the quality, reliability and safety of NEC semiconductor products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC semiconductor products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment, and anti-failure features.
- NEC semiconductor products are classified into the following three quality grades:
  "Standard", "Special" and "Specific". The "Specific" quality grade applies only to semiconductor products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of a semiconductor product depend on its quality grade, as indicated below. Customers must check the quality grade of each semiconductor product before using it in a particular application.

  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots

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

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

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

  (Note)

  (1) "NEC" as used in this statement means NEC Corporation and also includes its majority-owned subsidiaries.

  (2) "NEC semiconductor products" means any semiconductor product developed or manufactured by or for NEC (as defined above).

M8E 00.4

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.
Madrid 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
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
Guarulhos-SP Brasil
Tel: 55-11-6462-6810
Fax: 55-11-6462-6829

J00.7

Major Revisions in This Edition

Page	Description
p.32	Modification of recommended connection of unused pins in **Table 2-1. Types of Pin Input/ Output Circuits**
p.46	Modification of part of name and symbol in **Table 3-4. Special Function Register List**
p.80	Addition of **Caution** in **6.2 (1) 8-bit compare register 0n (CR0n)**
p.81	Modification of symbol and flag name in **Figure 6-2. 8-Bit Timer Mode Control Register 00 Format**
p.82	Modification of symbol and flag name in **Figure 6-3. 8-Bit Timer Mode Control Register 01 Format**
p.84	Modification of operation explanation in **6.4.1 Operation as interval timer**
p.86	Modification of operation explanation in **6.4.2 Operation as external event counter**
p.87	Modification of operation explanation in **6.4.3 Operation as square wave output**
p.101	Modification of symbol and flag name in **Figure 8-3. Serial Operating Mode Register 00 Format**, and addition of **Caution** regarding transmit/receive operation
p.102	Modification of symbol and flag name in **Figure 8-4. Asynchronous Serial Interface Mode Register 00 Format**, and addition of **Caution** regarding transmit operation
p.105	Modification of 1-bit memory manipulation instruction enabled in **8.3 (3) Asynchronous serial interface status register 00 (ASIS00)**
p.105	Modification of symbol and flag name in **Figure 8-5. Asynchronous Serial Interface Status Register 00 Format**
p.111	Addition of explanation regarding transmit operation and explanation regarding read operation of RXB00 register in **8.4.2 Asynchronous serial interface (UART) mode**
p.125	Addition of explanation regarding transmit/receive operation in **8.4.3 3-wire serial I/O mode**
p.135	Modification of flag name in **Figure 9-2. Interrupt Request Flag Register 0 Format**
p.136	Modification of flag name in **Figure 9-3. Interrupt Mask Flag Register 0 Format**
p.159	Addition of **Caution** in **Table 12-1. Differences between μPD78P9014 and Mask ROM Versions**
p.177	Addition of Solaris™ to operating system in **A.1 Language Processing Software**
p.178	Addition of PA-17K-DZ to PROM programmer adapter in **A.2.1 Hardware**
p.179	Modification of PC card interface name to IE-70000-CD-IF-A, and addition of IE-70000-PCI-IF to interface adapter in **A.3.1 Hardware**

The mark ★ shows major revised points.

INTRODUCTION

Readers

This manual is intended for user engineers who understand the functions of the μPD789014 Subseries to design and develop its application systems and programs.

- The target subseries is the μPD789014 Subseries, which consists of the μPD789011, 789012, and 78P9014.

Purpose

This manual is designed to deepen your understanding of the following functions using the following organization.

Organization

Two manuals are available for the μPD789014 Subseries: this manual and Instruction Manual (common to the 78K/0S Series).

μPD789014 Subseries User's Manual	78K/0S Series User's Manual — Instruction
• Pin functions • Internal block functions • Interrupt • Other internal peripheral functions	• CPU function • Instruction set • Instruction description

How to Read This Manual

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

□ To understand the overall functions of the μPD789014 Subseries
→ Read this manual in the order of the **CONTENTS**.

□ How to read register formats
→ The name of a bit whose number is in brackets is reserved for the assembler and is defined for the C compiler by the header file sfrbit.h.

□ To learn the detailed functions of a register whose register name is known
→ Refer to **APPENDIX C REGISTER INDEX**.

□ To learn the details of the instruction functions of the 78K/0S Series
→ Refer to **78K/0S Series User's Manual — Instruction (U11047E)** separately available.

Conventions

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

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

Device Related Documents

Document Name	Document No.	
	English	Japanese
μPD789011, 789012 Data Sheet	U11095E	U11095J
μPD78P9014 Data Sheet	U10912E	U10912J
μPD789014 Subseries User's Manual	This manual	U11187J
78K/0S Series User's Manual — Instruction	U11047E	U11047J

Documents for Development Tool (User's Manual)

Document Name		Document No.	
		English	Japanese
RA78K0S Assembler Package	Operation	U11622E	U11622J
	Assembly Language	U11599E	U11599J
	Structured Assembly Language	U11623E	U11623J
CC78K0S C Compiler	Operation	U11816E	U11816J
	Language	U11817E	U11817J
SM78K0S System Simulator Windows^{TM} Based	Reference	U11489E	U11489J
SM78K Series System Simulator	External Part User Open Interface Specifications	U10092E	U10092J
ID78K0S Integrated Debugger Windows Based	Reference	U12901E	U12901J

Document for Embedded Software (User's Manual)

Document Name	Document No.	
	English	Japanese
78K/0S Series OS MX78K0S	U12938E	U12938J

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

Other Related Documents

Document Name	Document No.	
	English	Japanese
NEC IC Package Manual (CD-ROM)	C13388E	—
Semiconductor Device Mounting Technology Manual	C10535E	C10535J
Quality Grades on NEC Semiconductor Devices	C11531E	C11531J
NEC Semiconductor Device Reliability/Quality Control System	C10983E	C10983J
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E	C11892J
Guide to Quality Assurance for Semiconductor Devices	MEI-1202	—
Microcomputer Product Series Guide	—	U11416J

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

Phase-out/Discontinued

[MEMO]

CONTENTS

LIST OF FIGURES (1/2)

Figure No. Title Page

LIST OF FIGURES (2/2)

Figure No. Title Page

LIST OF TABLES (1/2)

LIST OF TABLES (2/2)

Table No. Title Page

CHAPTER 1 GENERAL

1.1 Features

○ ROM and RAM capacity

Item Part Number	Program Memory		Data Memory (Internal High-Speed RAM)
μPD789011	ROM	2 Kbytes	128 bytes
μPD789012		4 Kbytes	
μPD78P9014	PROM	8 Kbytes	256 bytes

○ Minimum instruction execution time can be changed from high-speed (0.4 μs) to low-speed (1.6 μs) (system clock at 5.0-MHz operation)

○ I/O port: 22 lines

○ Serial interface: 1 channel
3-wire serial I/O mode/UART mode selection

○ Timer: 3 channels
- 8-bit timer/event counter : 2 channels
- Watchdog timer : 1 channel

★ ○ Vectored interrupt source: 9

○ Supply voltage: V_{DD} = 1.8 to 5.5 V

○ Operating ambient temperature: T_A = –40 to +85°C

1.2 Applications

Small home appliances, remote controls, video games, etc.

1.3 Ordering Information

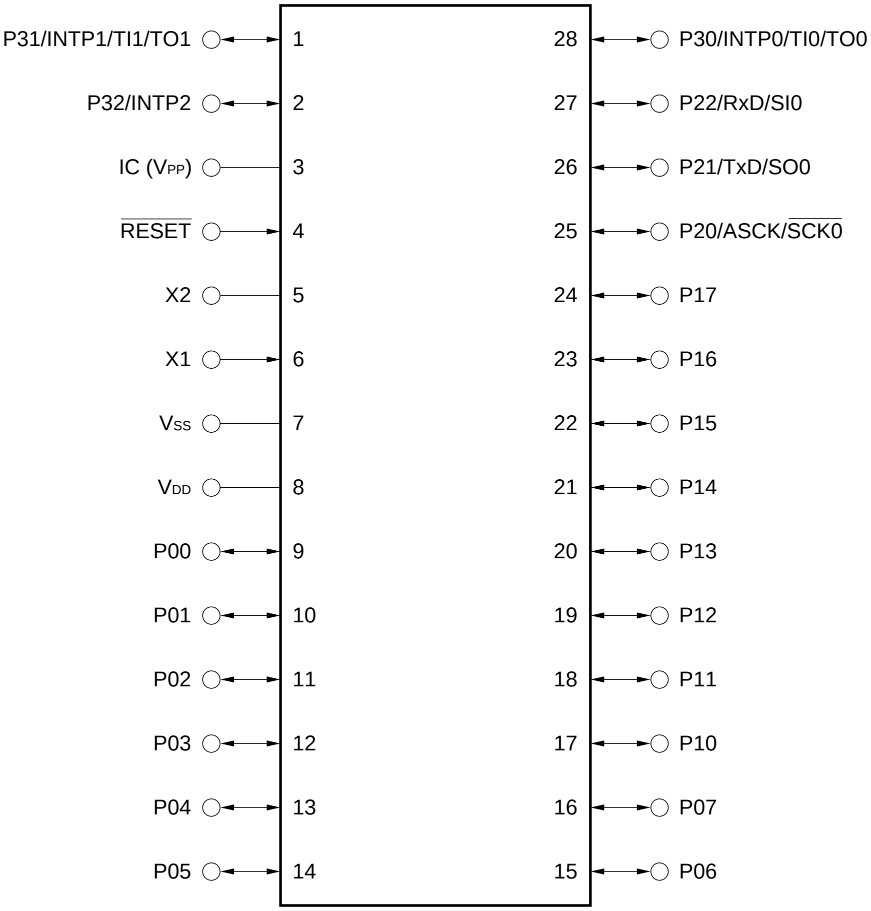
Part number	Package	Internal ROM
μPD789011CT-×××	28-pin plastic shrink DIP (400 mil)	Mask ROM
μPD789011GT-×××	28-pin plastic SOP (375 mil)	Mask ROM
μPD789012CT-×××	28-pin plastic shrink DIP (400 mil)	Mask ROM
μPD789012GT-×××	28-pin plastic SOP (375 mil)	Mask ROM
μPD78P9014CT	28-pin plastic shrink DIP (400 mil)	One-time PROM
μPD78P9014GT	28-pin plastic SOP (375 mil)	One-time PROM

Remark ××× indicates ROM code suffix.

1.4 Pin Configuration (Top View)

(1) Normal operating mode

- 28-pin plastic shrink DIP (400 mil)
  μPD789011CT-×××, 789012CT-×××, and 78P9014CT
- 28-pin plastic SOP (375 mil)
  μPD789011GT-×××, 789012GT-×××, and 78P9014GT



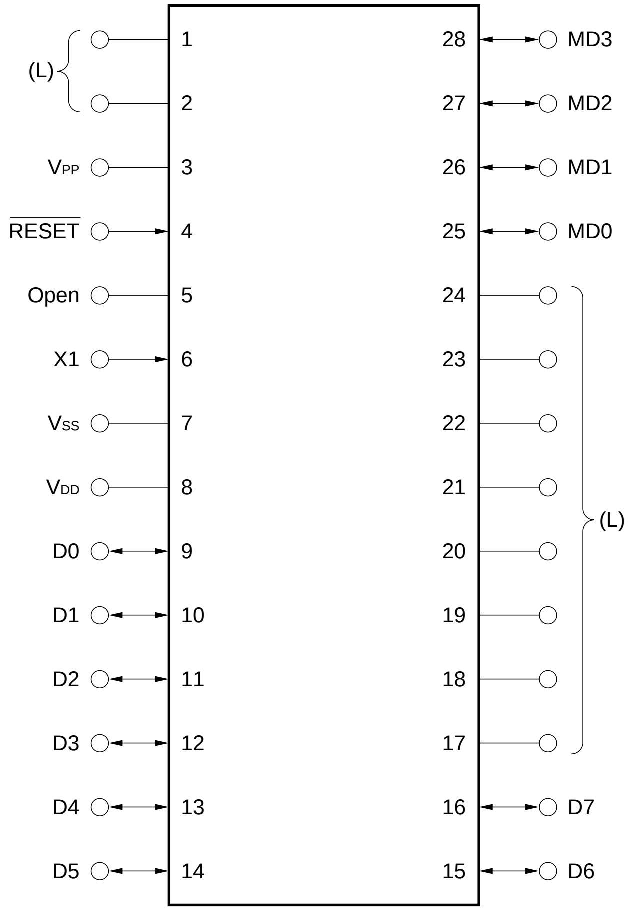
Caution Connect IC directly to V_{SS}.

Remark ( ): μPD78P9014

ASCK	:	Asynchronous Serial Clock	$\overline{\text{SCK0}}$	:	Serial Clock
IC	:	Internally Connected	SI0	:	Serial Input
P00 to P07	:	Port0	SO0	:	Serial Output
P10 to P17	:	Port1	TI0	:	Timer Input
P20 to P22	:	Port2	TO0	:	Timer Output
P30 to P32	:	Port3	TxD	:	Transmit Data
$\overline{\text{RESET}}$	:	Reset	V_{DD}	:	Power Supply
RxD	:	Receive Data	V_{PP}	:	Programming Power Supply
			V_{SS}	:	Ground

(2) **PROM programming mode**

- 28-pin plastic shrink DIP (400 mil)
  μPD78P9014CT
- 28-pin plastic SOP (375 mil)
  μPD78P9014GT



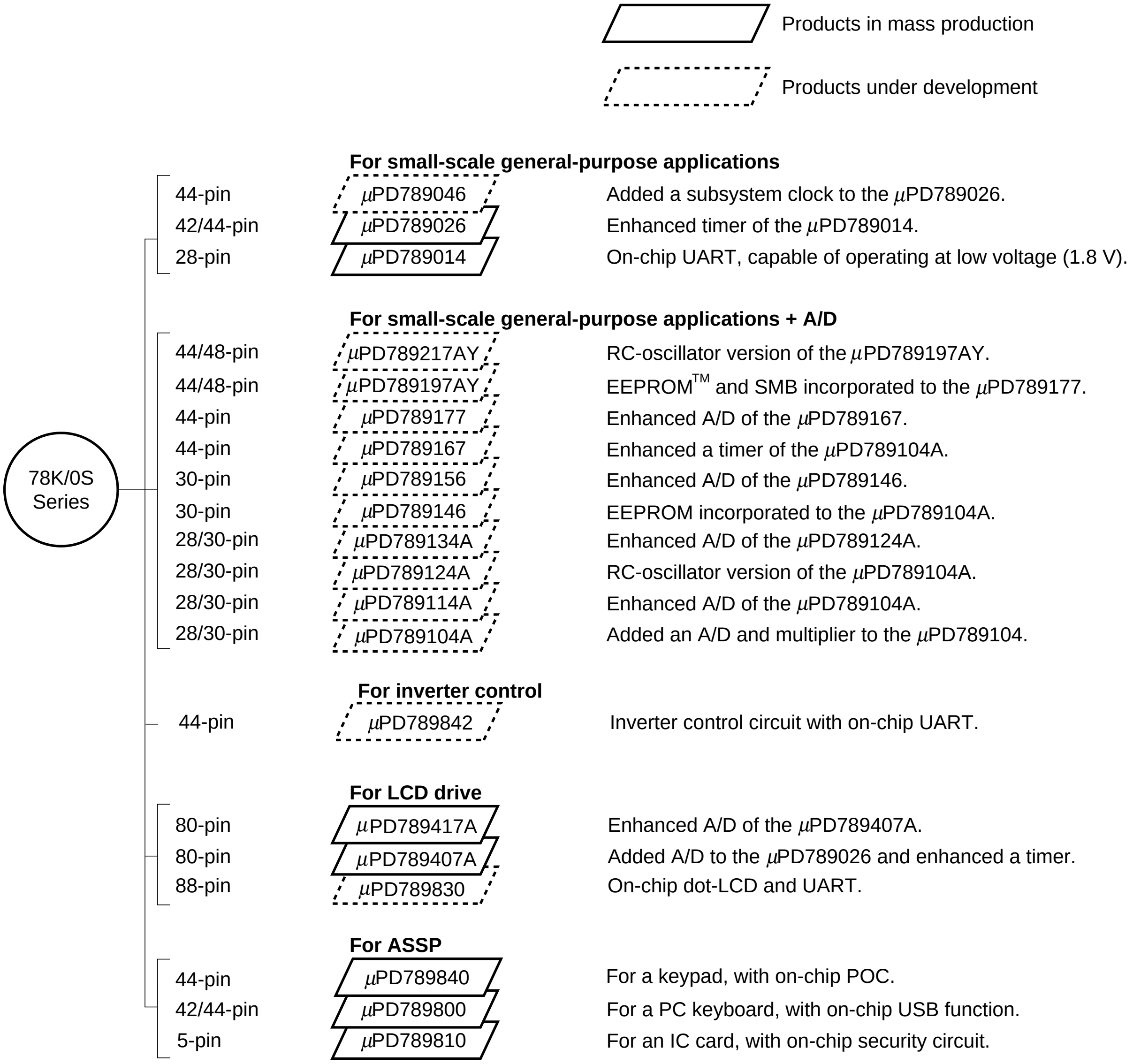
Cautions 1. (L) : Independently connect to V_{SS} via a pull-down resistor.
2. V_{SS} : Connect to the ground.
3. $\overline{\text{RESET}}$: Set to low level.
4. Open : Leave open.

D0 to D7	: Data Bus
MD0 to MD3	: Programming Power Supply
$\overline{\text{RESET}}$	: Reset
V_{DD}	: Power Supply
V_{PP}	: Programming Power Supply
V_{SS}	: Ground
X1	: Programming Clock Input

Phase-out/Discontinued

★ 1.5 78K/0S Series Lineup

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



Products in mass production

Products under development

78K/0S Series

For small-scale general-purpose applications

44-pin	μPD789046	Added a subsystem clock to the μPD789026.
42/44-pin	μPD789026	Enhanced timer of the μPD789014.
28-pin	μPD789014	On-chip UART, capable of operating at low voltage (1.8 V).

For small-scale general-purpose applications + A/D

44/48-pin	μPD789217AY	RC-oscillator version of the μPD789197AY.
44/48-pin	μPD789197AY	EEPROM^{TM} and SMB incorporated to the μPD789177.
44-pin	μPD789177	Enhanced A/D of the μPD789167.
44-pin	μPD789167	Enhanced a timer of the μPD789104A.
30-pin	μPD789156	Enhanced A/D of the μPD789146.
30-pin	μPD789146	EEPROM incorporated to the μPD789104A.
28/30-pin	μPD789134A	Enhanced A/D of the μPD789124A.
28/30-pin	μPD789124A	RC-oscillator version of the μPD789104A.
28/30-pin	μPD789114A	Enhanced A/D of the μPD789104A.
28/30-pin	μPD789104A	Added an A/D and multiplier to the μPD789104.

For inverter control

44-pin	μPD789842	Inverter control circuit with on-chip UART.

For LCD drive

80-pin	μPD789417A	Enhanced A/D of the μPD789407A.
80-pin	μPD789407A	Added A/D to the μPD789026 and enhanced a timer.
88-pin	μPD789830	On-chip dot-LCD and UART.

For ASSP

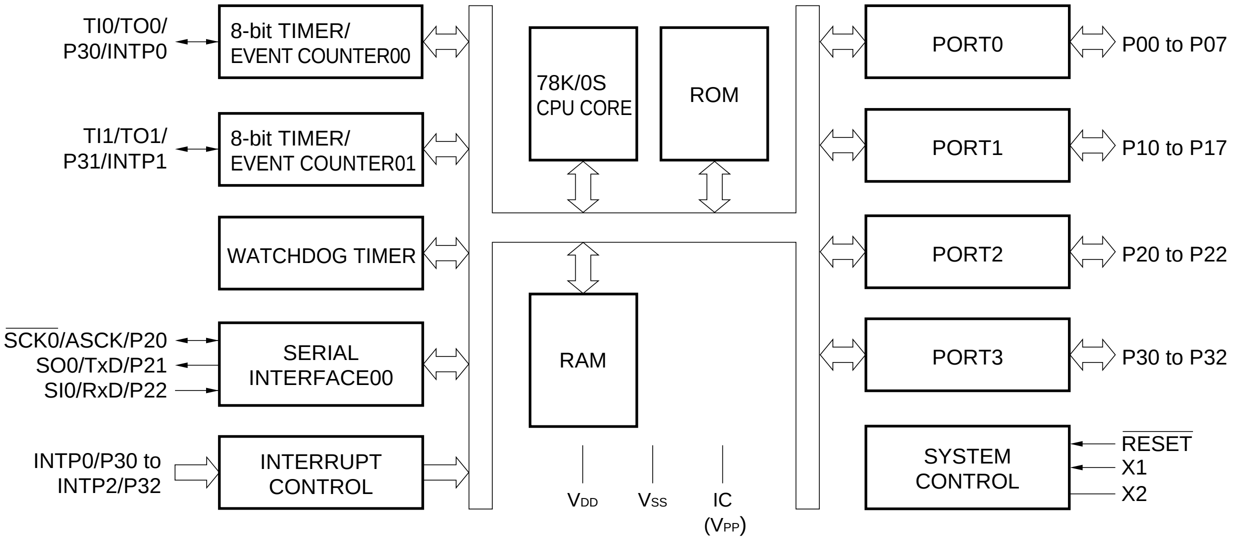
44-pin	μPD789840	For a keypad, with on-chip POC.
42/44-pin	μPD789800	For a PC keyboard, with on-chip USB function.
5-pin	μPD789810	For an IC card, with on-chip security circuit.

The major functional differences among the subseries are listed below.

Subseries Name	Function	ROM Capacity	Timer				8-Bit A/D	10-Bit A/D	Serial Interface	I/O	V_{DD} MIN. Value	Remark
			8-Bit	16-Bit	Watch	WDT						
Small-scale general-purpose	μPD789046	16 K	1 ch	1 ch	1 ch	1 ch	–	–	1 ch (UART: 1 ch)	34	1.8 V	–
	μPD789026	4 K to 16 K			–							
	μPD789014	2 K to 4 K	2 ch	–						22		
Small-scale general-purpose + A/D	μPD789217AY	16 K to 24 K	3 ch	1 ch	1 ch	1 ch	–	8 ch	2 ch (UART: 1 ch SMB: 1 ch)	31	1.8 V	RC-oscillator-version, On-chip EEPROM
	μPD789197AY											On-chip EEPROM
	μPD789177								1 ch (UART: 1 ch)			–
	μPD789167						8 ch	–				
	μPD789156	8 K to 16 K	1 ch		–		–	4 ch		20		On-chip EEPROM
	μPD789146						4 ch	–				
	μPD789134A	2 K to 8 K					–	4 ch				RC-oscillator-version
	μPD789124A						4 ch	–				
	μPD789114A						–	4 ch				–
	μPD789104A						4 ch	–				
Inverter control	μPD789842	8 K to 16 K	3 ch	**Note**	1 ch	1 ch	8 ch	–	1 ch (UART: 1 ch)	30	4.0 V	–
LCD drive	μPD789417A	12 K to 24 K	3 ch	1 ch	1 ch	1 ch	–	7 ch	1 ch (UART: 1 ch)	43	1.8 V	–
	μPD789407A						7 ch	–				
	μPD789830	24 K	1 ch				–			30	2.7 V	
ASSP	μPD789840	8 K	2 ch	–	–	1 ch	4 ch	–	1 ch	29	2.8 V	–
	μPD789800						–		2ch (USB: 1 ch)	31	4.0 V	
	μPD789810	6 K	–						–	1	1.8 V	On-chip EEPROM

Note 10-bit timer: 1 channel

1.6 Block Diagram



Remarks 1. The internal ROM and RAM capacities vary depending on the product.

2. An item in parentheses applies to the μPD78P9014 only.

1.7 Overview of Functions

Part Number Item		μPD789011	μPD789012	μPD78P9014
Internal memory	ROM	Mask ROM		One-time PROM
		2 Kbytes	4 Kbytes	8 Kbytes
	High-speed RAM	128 bytes		256 bytes
Minimum instruction execution time		0.4 μs/1.6 μs (@ 5.0-MHz operation with system clock)		
Instruction set		• 16-bit operation • Bit manipulation (set, reset, test), etc.		
I/O ports		CMOS input/output : 22		
Serial interface		3-wire serial I/O mode/UART mode selectable: 1 channel		
Timer		• 8-bit timer/event counter : 2 channels • Watchdog timer : 1 channel		
Timer outputs		2		
★ Vectored interrupt sources	Maskable	Internal: 5, External: 3		
	Non-maskable	Internal: 1		
Power supply voltage		V_{DD} = 1.8 to 5.5 V		
Operating ambient temperature		T_A = –40 to +85°C		
Packages		• 28-pin plastic shrink DIP (400 mil) • 28-pin plastic SOP (375 mil)		

[MEMO]

CHAPTER 2   PIN FUNCTIONS

2.1 List of Pin Functions

2.1.1 Pins in normal operating mode

(1) Port pins

Pin Name	Input/Output	Function	After Reset	Alternate Function
P00 to P07	Input/output	Port 0 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register (PUO). LEDs can be driven directly.	Input	—
P10 to P17	Input/output	Port 1 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register (PUO). LEDs can be driven directly.	Input	—
P20	Input/output	Port 2 3-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register (PUO). LEDs can be driven directly.	Input	ASCK/$\overline{\text{SCK0}}$
P21				TxD/SO0
P22				RxD/SI0
P30	Input/output	Port 3 3-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register (PUO). LEDs can be driven directly.	Input	INTP0/TI0/TO0
P31				INTP1/TI1/TO1
P32				INTP2

(2) Non-port pins

Pin Name	Input/Output	Function	After Reset	Alternate Function
INTP0Note	Input	External interrupt input for which the valid edge (rising edge, falling edge, or both rising edge and falling edge) can be specified	Input	P30/TI0/TO0
INTP1Note				P31/TI1/TO1
INTP2Note				P32
SI0Note	Input	Serial interface serial data input	Input	P22/RxD
SO0	Output	Serial interface serial data output	Input	P21/TxD
$\overline{\text{SCK0}}$Note	Input/output	Serial interface serial clock input/output	Input	P20/ASCK
RxDNote	Input	Serial data input for asynchronous serial interface	Input	P22/SI0
TxD	Output	Serial data output for asynchronous serial interface	Input	P21/SO0
ASCKNote	Input	Serial clock input for asynchronous serial interface	Input	P20/$\overline{\text{SCK0}}$
TI0Note	Input	External count clock input to 8-bit timer (TM0)	Input	P30/INTP0/TO0
TI1Note		External count clock input to 8-bit timer (TM1)		P31/INTP1/TO1
TO0	Output	8-bit timer output	Input	P30/INTP0/TI0
TO1				P31/INTP1/TI1
$\overline{\text{RESET}}$	Input	System reset input	Input	—
X1	Input	Connecting crystal resonator for system clock oscillation	—	—
X2	—		—	—
V_{DD}	—	Positive power supply	—	—
V_{PP}	—	High voltage applied during program write/verify. Connect directly to V_{SS} in normal operating mode.	—	—
V_{SS}	—	Ground potential	—	—
IC	—	Internally connected. Connect directly to V_{SS}.	—	—

Note Pin input from Schmitt-trigger (refer to Type 5-D of **Figure 2-1 Pin Input/Output Circuits**).

2.1.2 Pins in PROM programming mode

Pin Name	Input/Output	Function
$\overline{\text{RESET}}$	Input	Connect to V_{SS}.
V_{PP}	Input	PROM programming mode setting, and high voltage applied during program write/verify. When +5.5 V and +12.5 V are applied to V_{DD} and V_{PP} pins, respectively, PROM programming mode is set.
MD0 to MD3	Input/output	Operating mode selected in PROM programming mode
D0 to D7	Input/output	Data bus
X1	Input	Address update clock input in PROM programming mode
V_{DD}	—	PROM programming mode setting and positive power supply
V_{SS}	—	Ground potential

2.2 Description of Pin Functions

2.2.1 P00 to P07 (Port 0)

These pins constitute an 8-bit I/O port and can be set in the input or output port mode in 1-bit units by using port mode register 0 (PM0). When these pins are used as an input port, an on-chip pull-up resistor can be used in the pull-up resistor option register (PUO).

LEDs can be driven directly.

2.2.2 P10 to P17 (Port 1)

These pins constitute an 8-bit I/O port and can be set in the input or output port mode in 1-bit units by using port mode register 1 (PM1). When these pins are used as an input port, an on-chip pull-up resistor can be used in the pull-up resistor option register (PUO).

LEDs can be driven directly.

2.2.3 P20 to P22 (Port 2)

These pins constitute a 3-bit I/O port. In addition, these pins provide the function to input/output the data and clock of the serial interface.

LEDs can be driven directly.

Port 2 can be specified in the following operation modes in 1-bit units.

(1) Port mode

In this mode, port 2 functions as a 3-bit I/O port which can be set in the input or output port mode in 1-bit units by using the port mode register 2 (PM2). When the port is used as an input port, an on-chip pull-up resistor can be used in the pull-up resistor option register (PUO).

(2) Control mode

In this mode, the pins of port 2 function as the data input/output and the clock input/output of the serial interface.

(a) SI0, SO0

These are the serial data I/O pins of the serial interface.

(b) $\overline{\text{SCK0}}$

This is the serial clock I/O pin of the serial interface.

(c) RxD, TxD

This is the serial data I/O pin of asynchronous serial interface.

(d) ASCK

This is the serial clock input pin of asynchronous serial interface.

Caution When using P20 through P22 as serial interface pins, the I/O or output latch must be set according to the functions to be used. For the details of the setting, refer to Table 8-2 Serial Interface 00 Operating Mode Settings.

2.2.4 P30 to P32 (Port 3)

These pins constitute a 3-bit I/O port. In addition, they also function as external interrupt input and timer I/O.
LEDs can be driven directly.
Port 3 can be specified in the following operation modes in 1-bit units.

(1) Port mode

In this mode, port 3 functions as a 3-bit I/O port which can be set in the input or output port mode in 1-bit units by using the port mode register 3 (PM3). When the port is used as an input port, an on-chip pull-up resistor can be used in the pull-up resistor option register (PUO).

(2) Control mode

In this mode, the pins of port 3 function as external interrupt input and timer I/O.

(a) INTP0 to INTP2

These pins input external interrupt signals whose valid edge can be specified to either rising, falling, or both edges.

(b) TI0, TI1

These pins input an external clock to the 8-bit timer/event counter.

(c) TO0, TO1

These are 8-bit timer output pins.

2.2.5 $\overline{\text{RESET}}$

This pin inputs an active-low system reset signal.

2.2.6 X1, X2

These pins are used to connect a crystal resonator for system clock oscillation.

To supply an external clock, input the clock to X1 and input the inverted signal to X2.

2.2.7 V_{DD}

Positive power supply pin

2.2.8 V_{SS}

Ground pin

2.2.9 V_{PP} (μPD78P9014 only)

A high voltage should be applied to this pin when the PROM programming mode is set and when the program is written or verified.

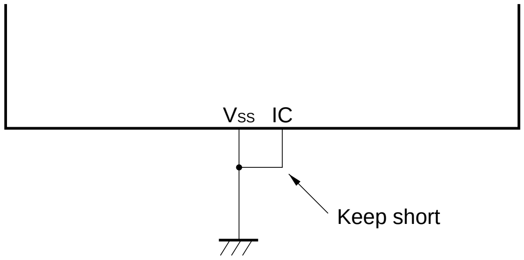
Connect this pin directly to V_{SS} in the normal operation mode.

2.2.10 IC (mask ROM version only)

The IC (Internally Connected) pin is used to set the μPD789011 and 789012 in the test mode before shipment. In the normal operation mode, connect this pin directly to the V_{SS} pin with as short a wiring length as possible.

If a potential difference is generated between the IC pin and V_{SS} pin due to a long wiring length between the IC pin and V_{SS} pin or an external noise superimposed on the IC pin, a user program may not run correctly.

○ Connect the IC pin directly to the V_{SS} pin.



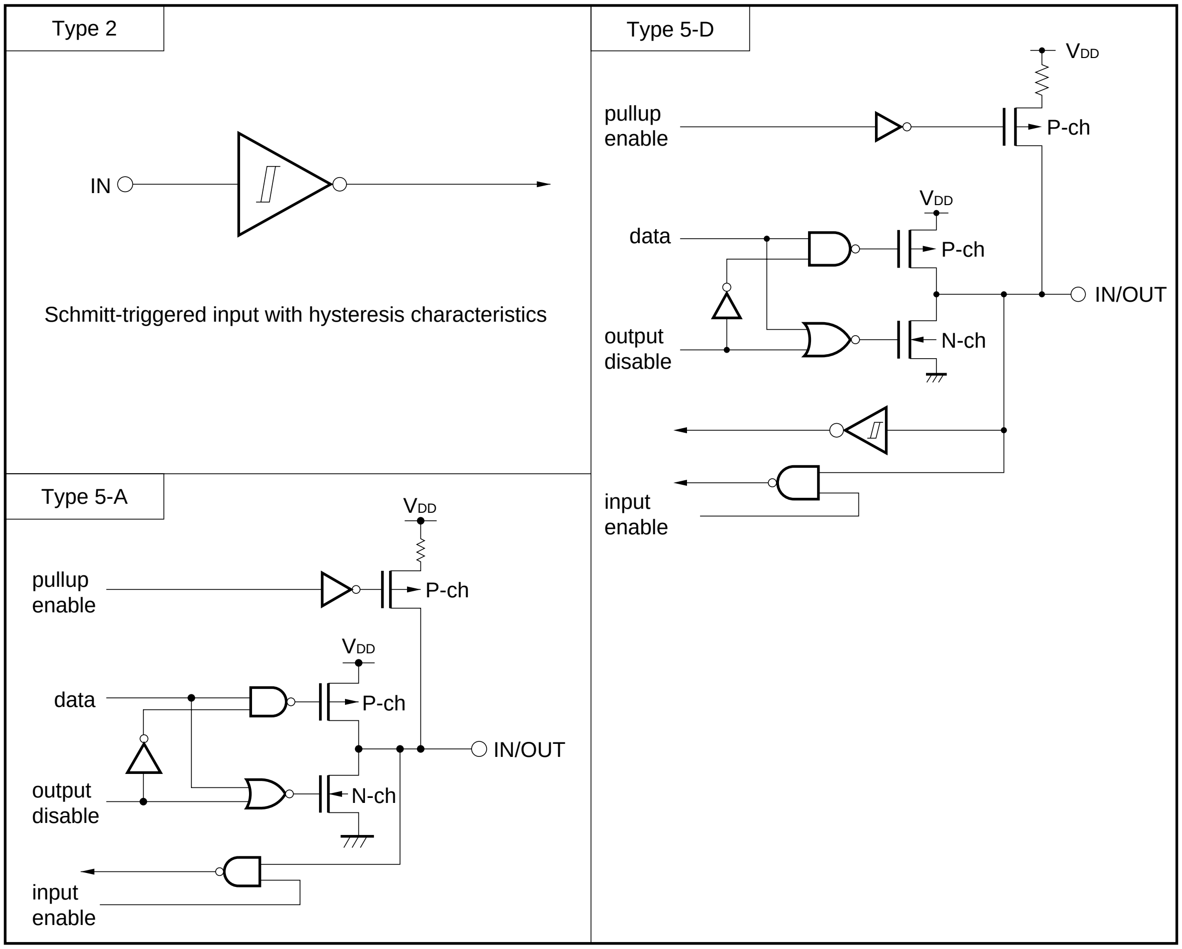
2.3 Pin Input/Output Circuits and Recommended Connection of Unused Pins

The input/output circuit type of each pin and recommended connection of unused pins are shown in Table 2-1.
For the input/output circuit configuration of each type, refer to Figure 2-1.

★ **Table 2-1. Types of Pin Input/Output Circuits**

Pin Name	Input/Output Circuit Type	Input/Output	Recommended Connection of Unused Pins
P00 to P07	5-A	Input/output	Input: Independently connect to V_{DD} or V_{SS} via a resistor. Output: Leave open.
P10 to P17			
P20/ASCK/$\overline{\text{SCK0}}$	5-D		
P21/TxD/SO0	5-A		
P22/RxD/SI0	5-D		
P30/INTP0/TI0/TO0			
P31/INTP1/TI1/TO1			
P32/INTP2			
$\overline{\text{RESET}}$	2	—	—
IC (mask ROM version)	—	—	Connect directly to V_{SS}.
V_{PP} (μPD78P9014)			

Figure 2-1. Pin Input/Output Circuits



3.1 Memory Space

The μPD789014 Subseries can access 64 Kbytes of memory space. Figures 3-1 through 3-3 show the memory maps.

Figure 3-1. Memory Map (μPD789011)

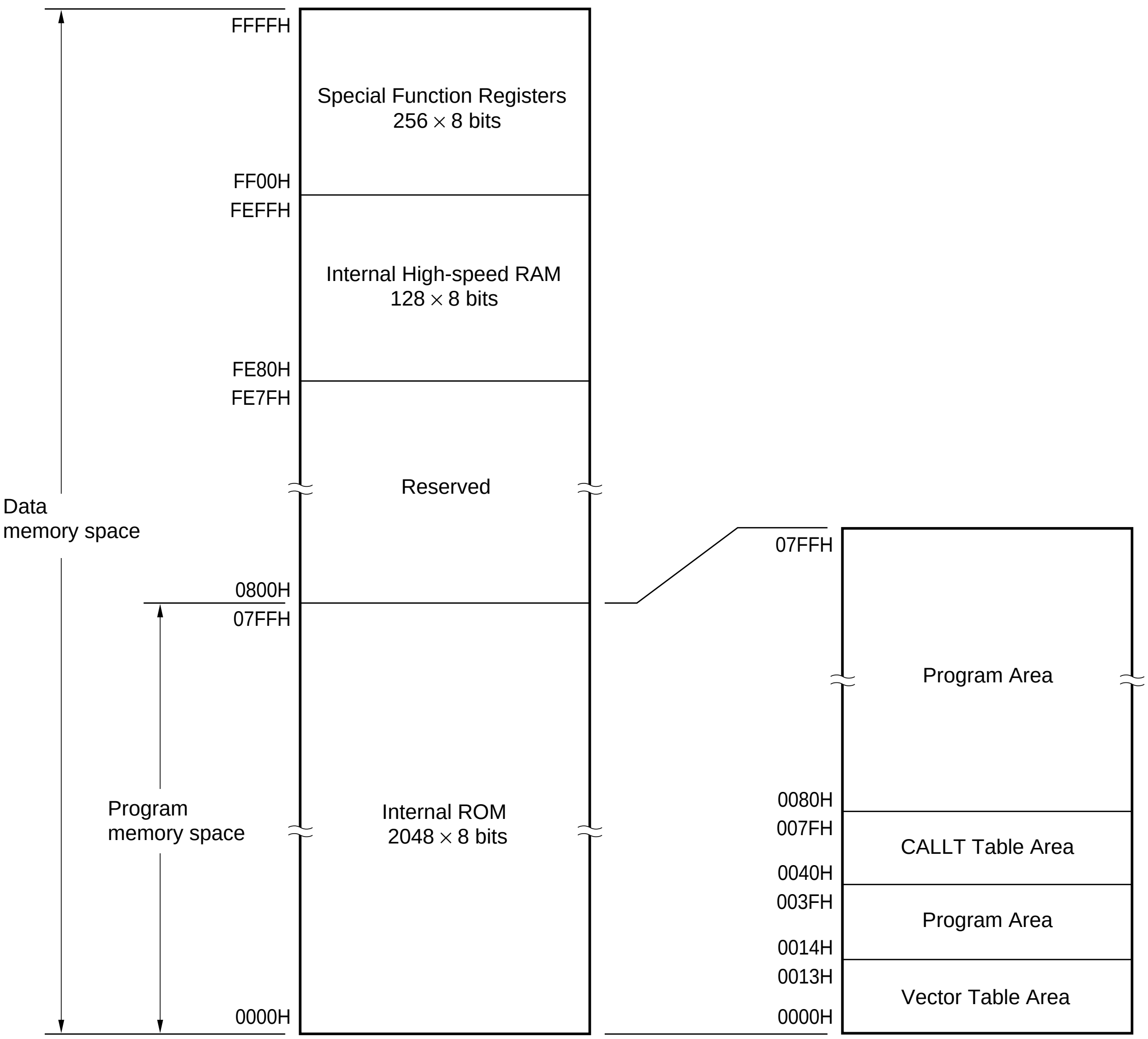


Figure 3-2. Memory Map (μPD789012)

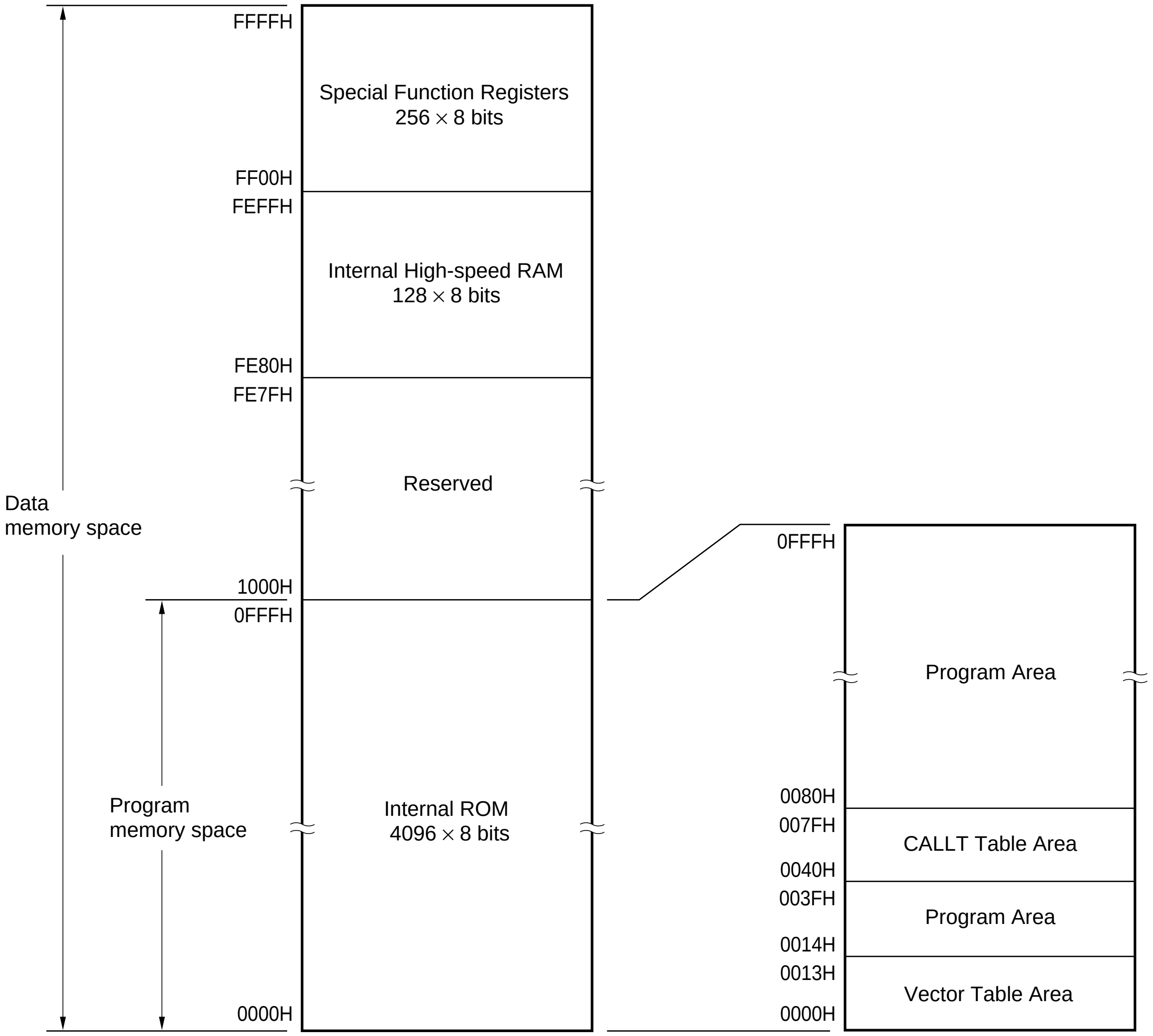
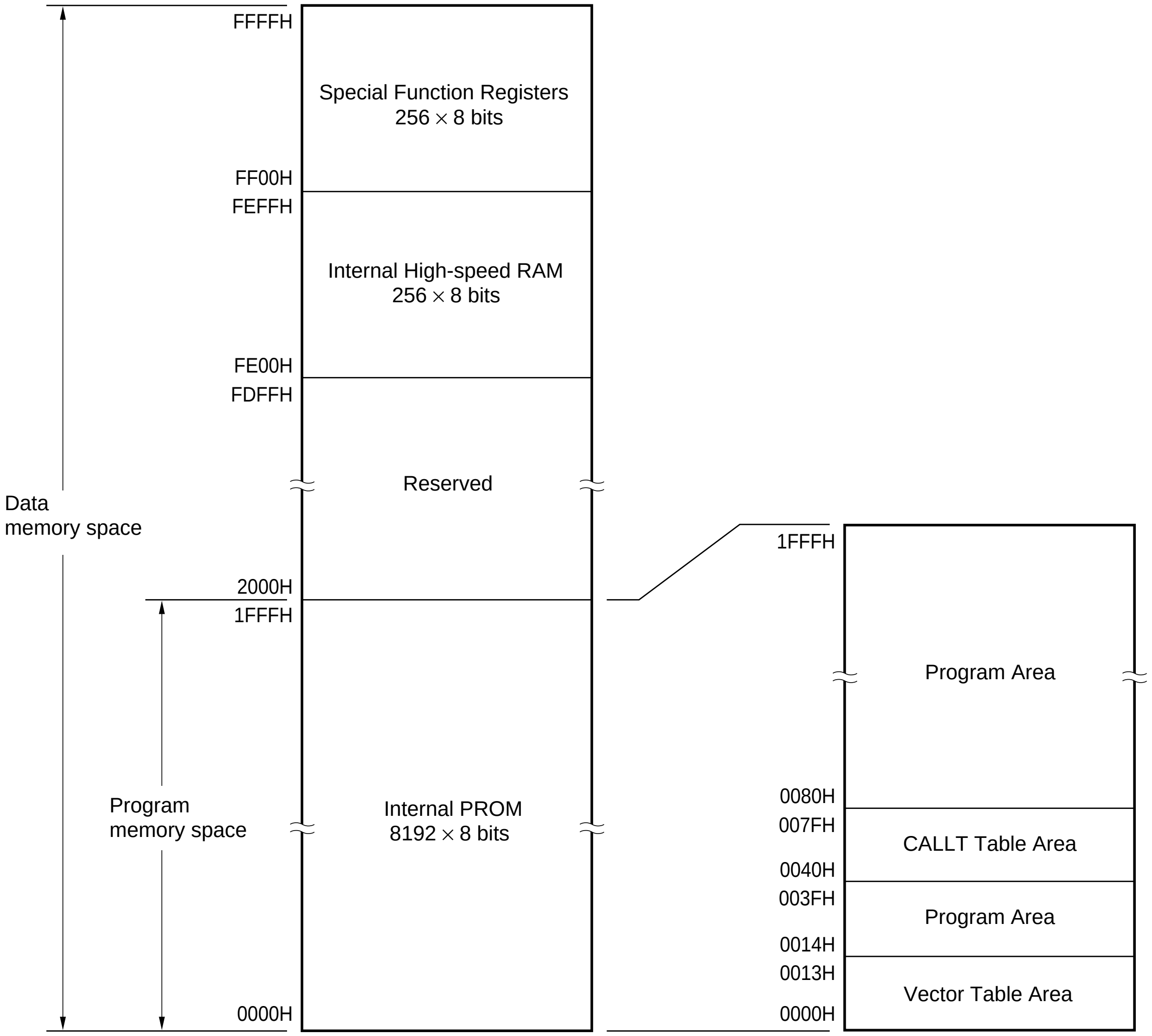


Figure 3-3. Memory Map (μPD78P9014)



3.1.1 Internal program memory space

The internal program memory space stores programs and table data. This space is usually addressed by the program counter (PC).

The μPD789014 Subseries provide the internal ROMs (or PROM) containing the following capacities on each product.

Table 3-1. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
μPD789011	Mask ROM	2048 × 8 bits
μPD789012		4096 × 8 bits
μPD78P9014	PROM	8192 × 8 bits

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

(1) Vector table area

A 20-byte area of addresses 0000H to 0013H is reserved as a vector table area. This area stores program start addresses to be used when branching by the $\overline{\text{RESET}}$ input or an interrupt request generation. Of a 16-bit program address, the lower 8 bits are stored in an even address, and the higher 8 bits are stored in an odd address.

Table 3-2. Vector Table

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	$\overline{\text{RESET}}$ input	000CH	INTSR/INTCSI0
0004H	INTWDT	000EH	INTST
0006H	INTP0	0010H	INTTM0
0008H	INTP1	0012H	INTTM1
000AH	INTP2		

(2) CALLT instruction table area

In a 64-byte area of addresses 0040H to 007FH, the subroutine entry address of a 1-byte call instruction (CALLT) can be stored.

3.1.2 Internal data memory (internal high-speed RAM) space

The μPD789014 Subseries provide internal high-speed RAM containing the following capacities on each product. The internal high-speed RAM can also be used as a stack memory.

Table 3-3. Internal High-Speed RAM Capacity

Part Number	Capacity
μPD789011	128 × 8 bits
μPD789012	
μPD78P9014	256 × 8 bits

3.1.3 Special function register (SFR) area

Special function registers (SFRs) of on-chip peripheral hardware are allocated to an area of FF00H to FFFFH (refer to **Table 3-4**).

3.1.4 Data memory addressing

The μPD789014 Subseries provide a variety of addressing modes which take account of memory manipulability, etc. Especially at addresses corresponding to data memory area (FE80H to FFFFH), particular addressing modes are possible to meet the functions of the special function registers (SFR) and general registers. Figures 3-4 through 3-6 show the data memory addressing modes.

Figure 3-4. Data Memory Addressing (μPD789011)

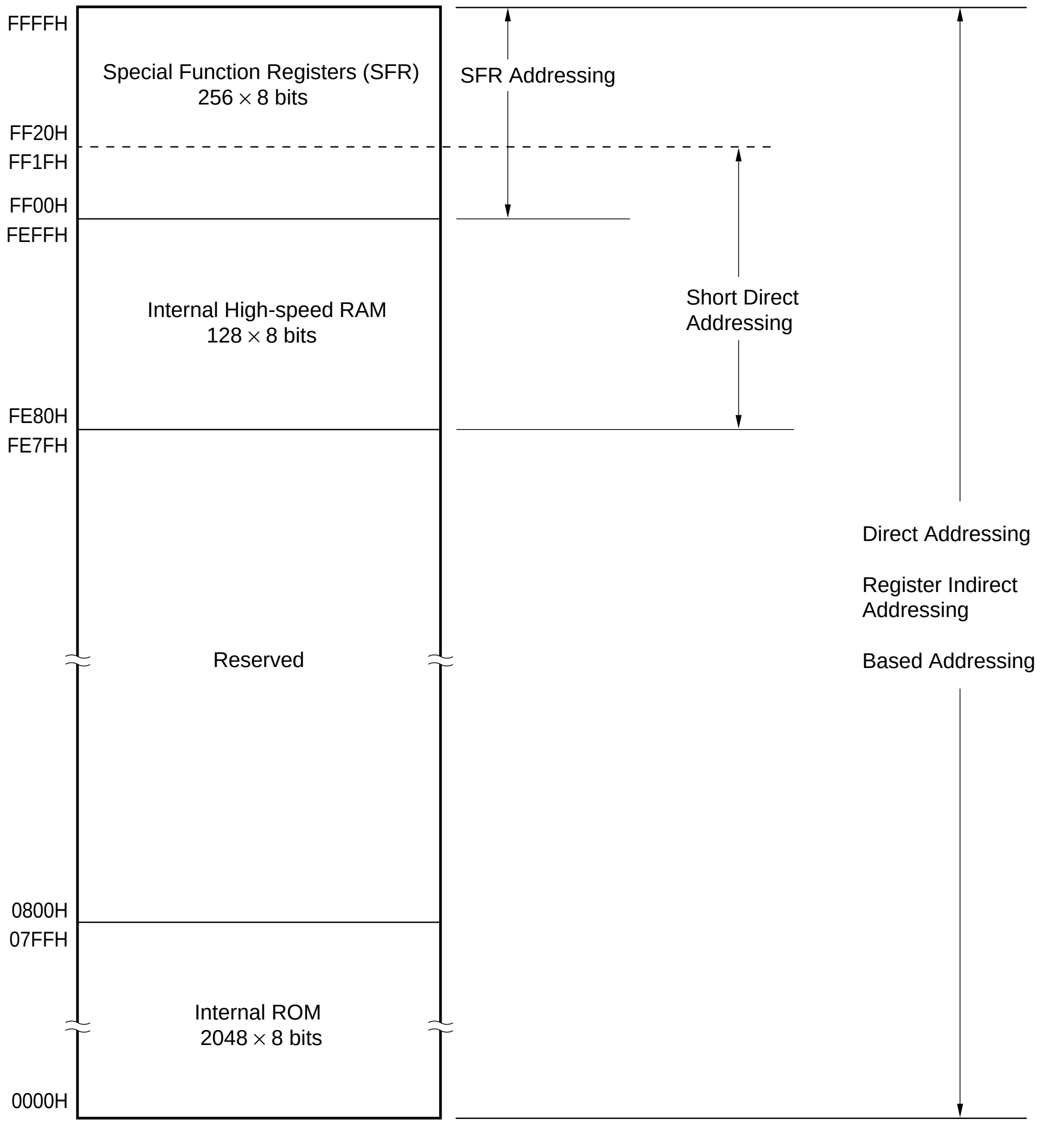
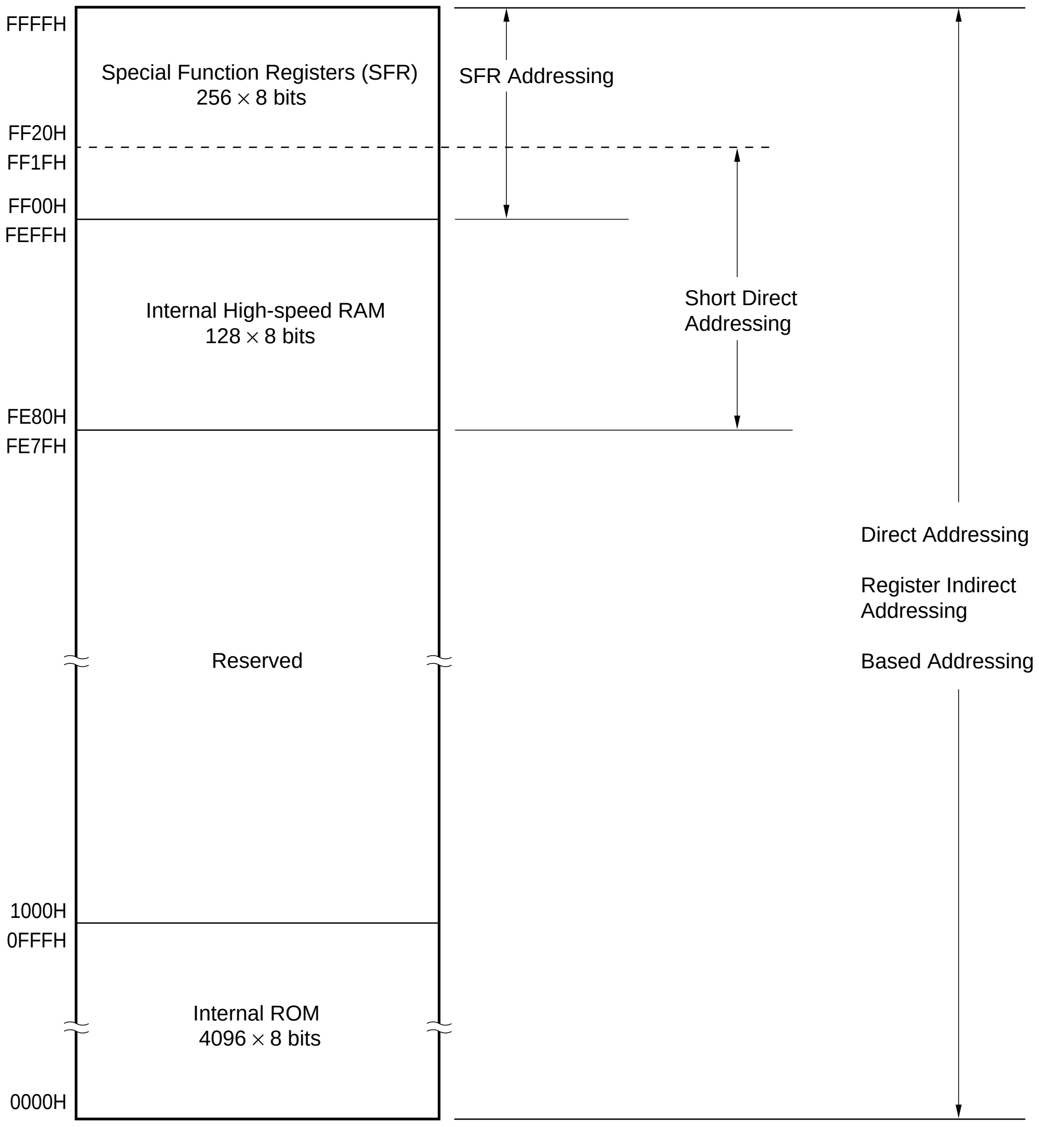
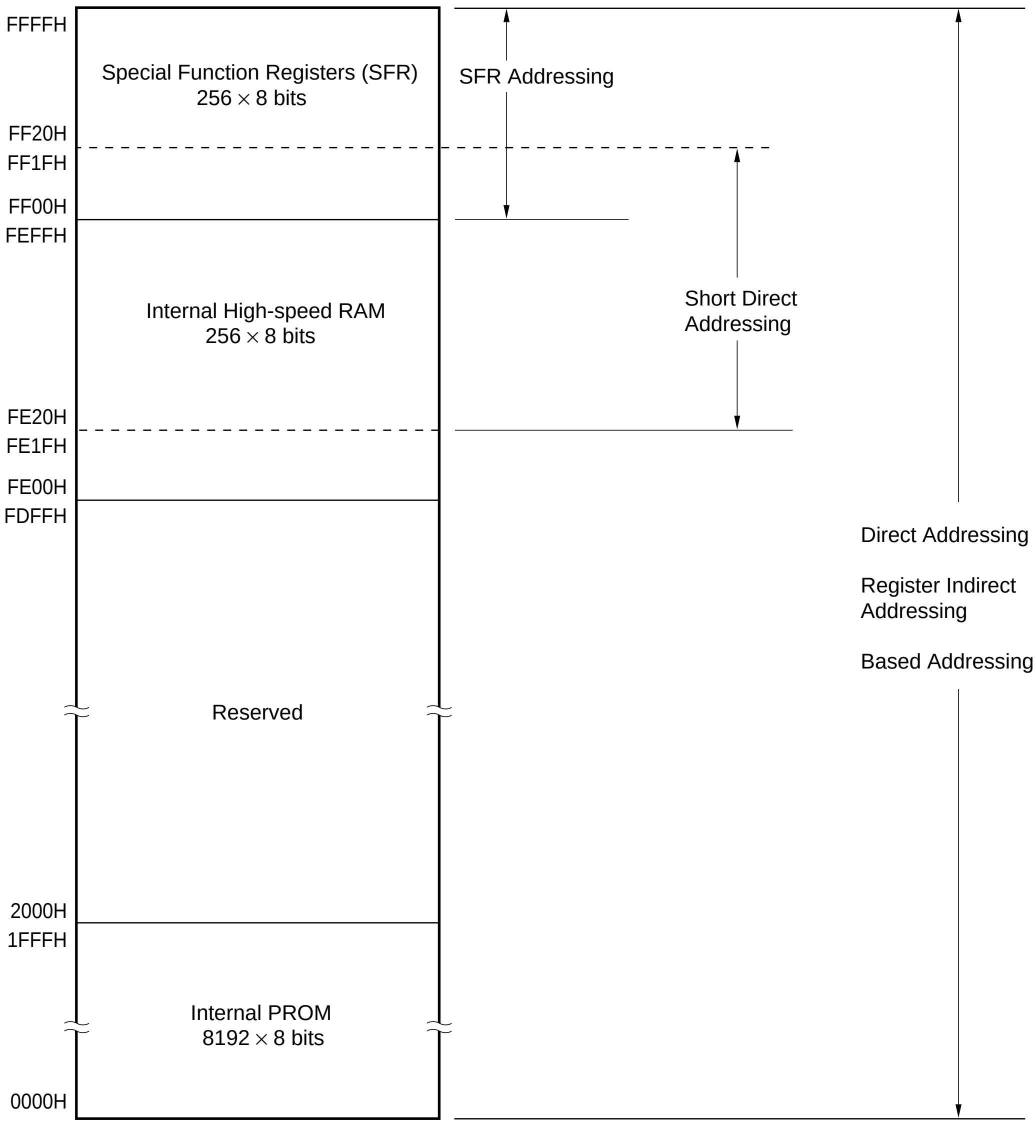


Figure 3-5. Data Memory Addressing (μPD789012)



Phase-out/Discontinued

Figure 3-6. Data Memory Addressing (μPD78P9014)



3.2 Processor Registers

The μPD789014 Subseries provide the following on-chip processor registers:

3.2.1 Control registers

The control registers contain special functions to control the program sequence statuses and stack memory. A program counter, a program status word, and a stack pointer are control registers.

(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 or register contents is set.

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

Figure 3-7. Program Counter Configuration

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

(2) Program status word (PSW)

The program status word is an 8-bit register consisting of various flags 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 RETI and POP PSW instructions.

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

Figure 3-8. Program Status Word Configuration

	7							0
PSW	IE	Z	0	AC	0	0	1	CY

(a) Interrupt enable flag (IE)

This flag controls interrupt request acknowledge operations of CPU.

When IE = 0, the IE is set to interrupt disabled (DI) status. All interrupt requests except non-maskable interrupt are disabled.

When IE = 1, the IE is set to interrupt enabled (EI) status and interrupt request acknowledgement is controlled with an interrupt mask flag for various interrupt sources.

This flag is reset to (0) upon DI instruction execution or interrupt acknowledgment and is set to (1) upon EI instruction execution.

(b) Zero flag (Z)

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

(c) Auxiliary carry flag (AC)

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

(d) 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 can be set as the stack area.

Figure 3-9. Stack Pointer Configuration

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

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

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

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

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

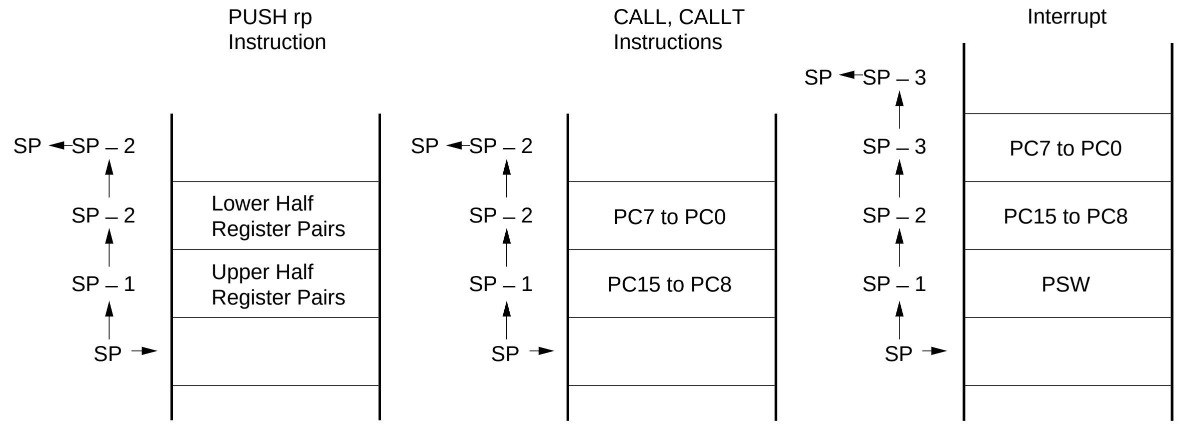
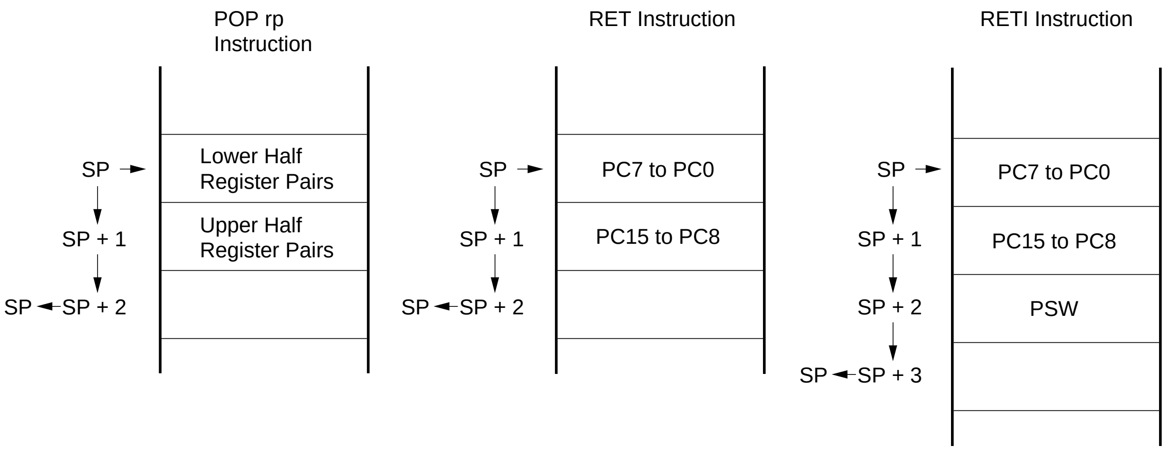


Figure 3-11. Data to be Restored from Stack Memory



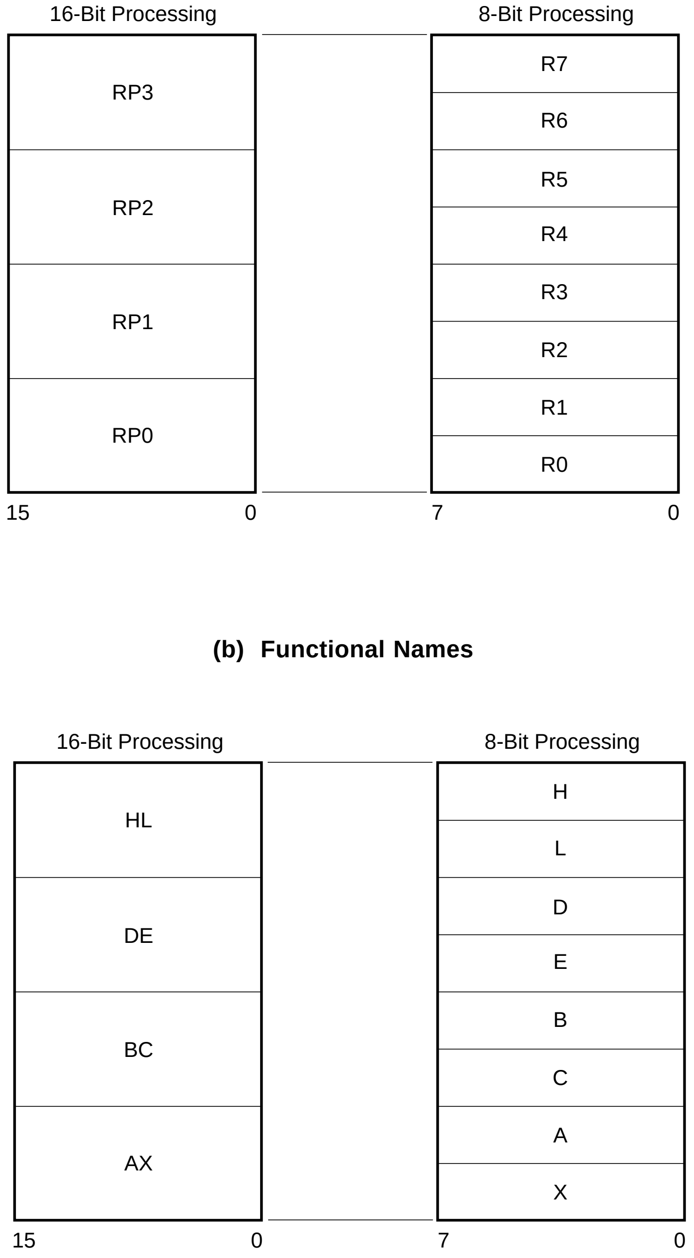
3.2.2 General registers

A general register consists of eight 8-bit registers (X, A, C, B, E, D, L, and H).

In addition that each register can be used as an 8-bit register, two 8-bit registers in pairs can be used as a 16-bit register (AX, BC, DE, and HL).

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

Figure 3-12. General Register Configuration



(a) Absolute Names

(b) Functional Names

3.2.3 Special function register (SFR)

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

It is allocated in the 256-byte area FF00H to FFFFH.

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) differ depending on the special function register type.

Each manipulation bit unit can be specified as follows.

- 1-bit manipulation
  Describes a symbol reserved with assembler for the 1-bit manipulation instruction operand (sfr.bit). This manipulation can also be specified with an address.

- 8-bit manipulation
  Describes a symbol reserved with assembler for the 8-bit manipulation instruction operand (sfr). This manipulation can also be specified with an address.

- 16-bit manipulation
  Describes a symbol reserved with assembler for the 16-bit manipulation instruction operand. When addressing an address, describe an even address.

Table 3-4 lists the special function register. The meanings of the symbols in this table are as follows:

- Symbol
  Indicates the addresses of the implemented special function registers. The symbols shown in this column are the reserved words of the assembler, and have already been defined in the header file called “sfrbit.h” of C compiler. Therefore, these symbols can be used as instruction operands if assembler or integrated debugger is used.

- R/W
  Indicates whether the special function register in question can be read or written.
  R/W : Read/write
  R : Read only
  W : Write only

- Manipulatable bit unit
  Indicates the bit units (1, 8, 16) in which the special function register in question can be manipulated.

- After reset
  Indicates the status of the special function register when the $\overline{\text{RESET}}$ signal is input.

★ **Table 3-4. Special Function Register List**

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		○	○	—	
FF10H	Transmit shift register 00	TXS00 SIO00	W	—	○	—	FFH
	Receive buffer register 00	RXB00	R				Undefined
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		○	○	—	
FF42H	Timer clock select register 2	TCL2		—	○	—	00H
FF50H	Compare register 00	CR00	W	—	○	—	Undefined
FF51H	8-bit timer register 00	TM00	R	—	○	—	00H
FF53H	8-bit timer mode control register 00	TMC00	R/W	○	○	—	
FF54H	Compare register 01	CR01	W	—	○	—	Undefined
FF55H	8-bit timer register 01	TM01	R	—	○	—	00H
FF57H	8-bit timer mode control register 01	TMC01	R/W	○	○	—	
FF70H	Asynchronous serial interface mode register 00	ASIM00		○	○	—	
FF71H	Asynchronous serial interface status register 00	ASIS00	R	○	○	—	
FF72H	Serial operating mode register 00	CSIM00	R/W	○	○	—	
FF73H	Baud rate generator control register 00	BRGC00		—	○	—	
FFE0H	Interrupt request flag register 0	IF0		○	○	—	
FFE4H	Interrupt mask flag register 0	MK0		○	○	—	FFH
FFECH	External interrupt mode register 0	INTM0		—	○	—	00H
FFF7H	Pull-up resistor option register	PUO		○	○	—	
FFF9H	Watchdog timer mode register	WDTM		○	○	—	
FFFAH	Oscillation stabilization time select register	OSTS		—	○	—	04H
FFFBH	Processor clock control register	PCC		○	○	—	02H

3.3 Instruction Address Addressing

An instruction address is determined by program counter (PC) contents. PC contents are 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 each instruction, refer to **78K/0S Series User's Manual — Instruction (U11047E)**).

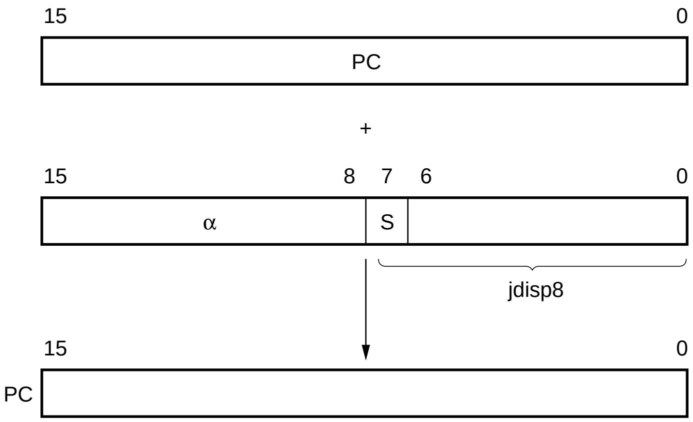
3.3.1 Relative addressing

[Function]

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

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

[Illustration]



··· PC is the start address of
the next instruction of
a BR instruction.

When S = 0, α indicates all bits "0".
When S = 1, α indicates all bits "1".

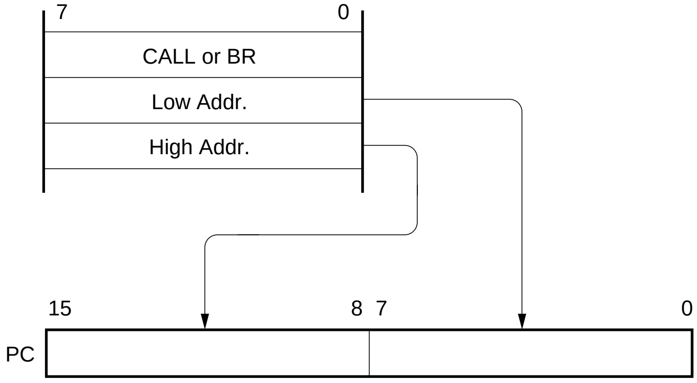
3.3.2 Immediate addressing

[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 and BR !addr16 instructions are executed.
CALL !addr16 and BR !addr16 instructions can branch to all the memory spaces.

[Illustration]

In case of CALL !addr16, BR !addr16 instruction



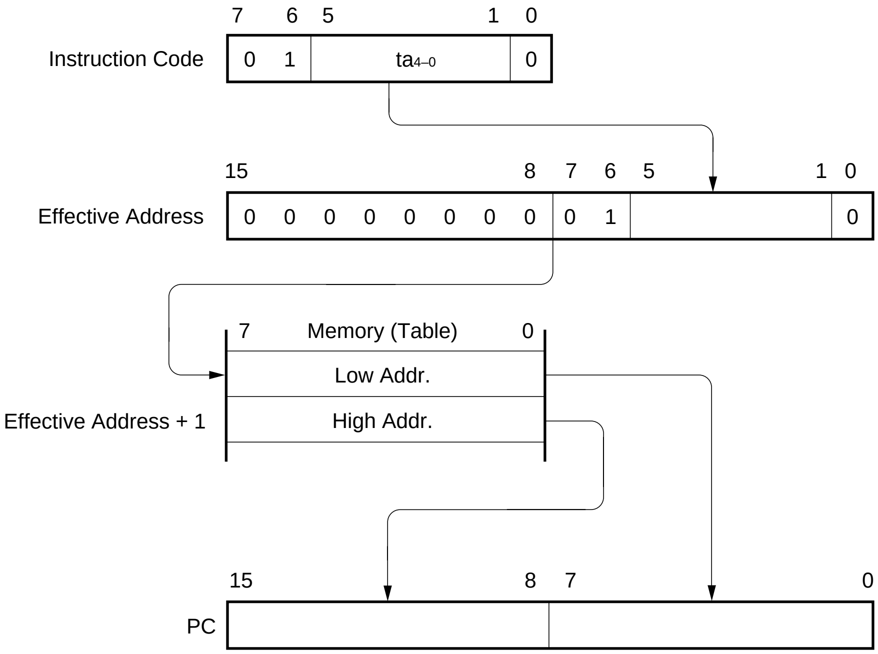
3.3.3 Table indirect addressing

[Function]

Table contents (branch destination address) of the particular location to be addressed by the low-order-5-bit immediate data of an instruction code from bit 1 to bit 5 are transferred to the program counter (PC) and branched.

Table indirect addressing is carried out when the CALLT [addr5] instruction is executed. This instruction can refer to the address stored in the memory table 40H to 7FH and branch to all the memory spaces.

[Illustration]



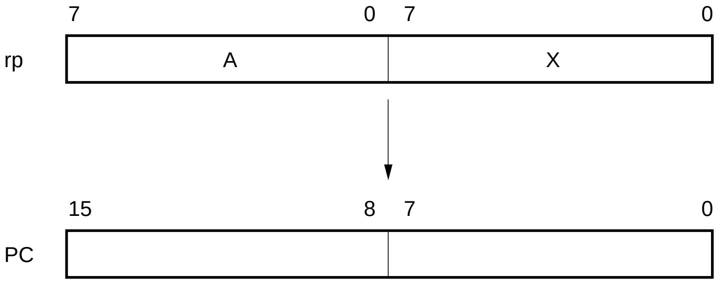
3.3.4 Register addressing

[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 Direct addressing

[Function]

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

[Operand format]

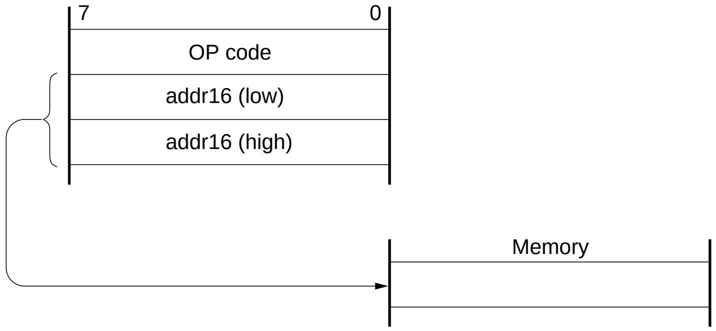
Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

MOV A, !FE00H; When setting !addr16 to FE00H

Instruction code	0	0	1	0	1	0	0	1	OP code
	0	0	0	0	0	0	0	0	00H
	1	1	1	1	1	1	1	0	FEH

[Illustration]



3.4.2 Short direct addressing

[Function]

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

The fixed space where this addressing is applied to is the 160-byte space FE80H to FF1FH. An internal high-speed RAM and a special function register (SFR) are mapped at FE80H to FEFFH and FF00H to FF1FH, respectively.

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

[Operand format]

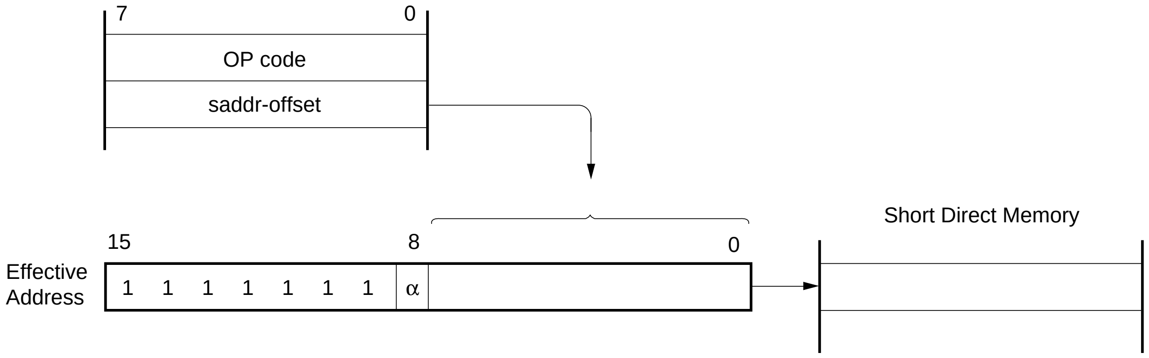
Identifier	Description
saddr	Label or FE80H to FF1FH immediate data
saddrp	Label or FE80H to FF1FH immediate data (even address only)

[Description example]

MOV FE90H, #50H; When setting saddr to FE90H and the immediate data to 50H

Instruction code	1	1	1	1	0	1	0	1	OP code
	1	0	0	1	0	0	0	0	90H (saddr-offset)
	0	1	0	1	0	0	0	0	50H (immediate data)

[Illustration]



When 8-bit immediate data is 20H to FFH, α = 0.
When 8-bit immediate data is 00H to 1FH, α = 1.

3.4.3 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]

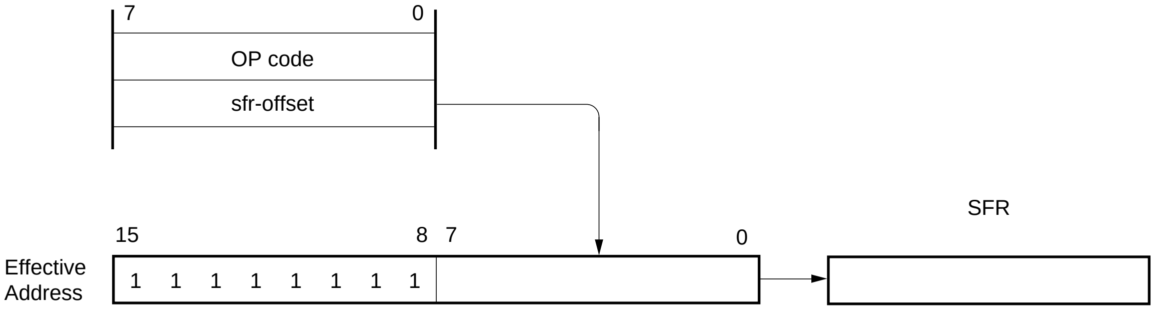
Identifier	Description
sfr	Special function register name

[Description example]

MOV PM0, A; When selecting PM0 for sfr

Instruction code	1	1	1	0	0	1	1	1
	0	0	1	0	0	0	0	0

[Illustration]



3.4.4 Register addressing

[Function]

The general register is accessed as an operand. The general register to be accessed is specified with register specify code and functional name in the instruction code.

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

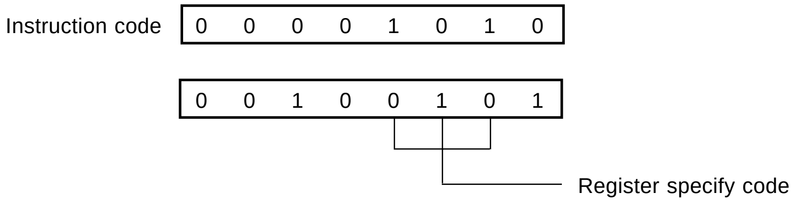
[Operand format]

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

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

[Description example]

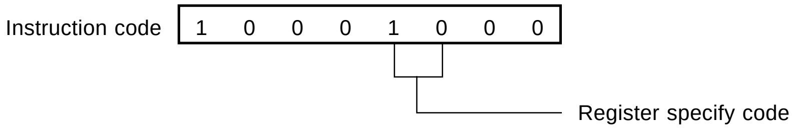
MOV A, C; When selecting the C register for r



Instruction code 0 0 0 0 1 0 1 0

0 0 1 0 0 1 0 1

INCW DE; When selecting the DE register pair for rp



Instruction code 1 0 0 0 1 0 0 0

3.4.5 Register indirect addressing

[Function]

The memory is addressed with the contents of the register pair specified as an operand. The register pair to be accessed is specified with the register pair specify code in the instruction code. This addressing can be carried out for all the memory spaces.

[Operand format]

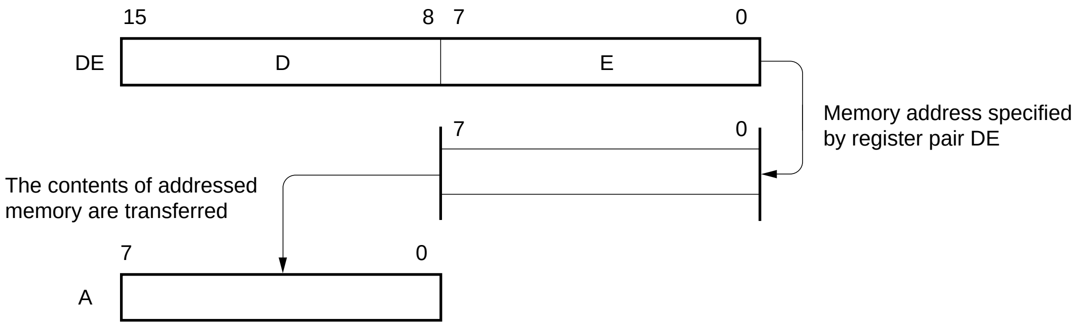
Identifier	Description
—	[DE], [HL]

[Description example]

MOV A, [DE]; When selecting register pair [DE]

Instruction code	0	0	1	0	1	0	1	1

[Illustration]


15
8
7
0
DE
D
E
7
0
Memory address specified by register pair DE
The contents of addressed memory are transferred
7
0
A

3.4.6 Based addressing

[Function]

8-bit immediate data is added to the contents of the base register, that is, the HL register pair, and the sum is used to address the memory. Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be carried out for all the memory spaces.

[Operand format]

Identifier	Description
—	[HL+byte]

[Description example]

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

Instruction code	0	0	1	0	1	1	0	1
	0	0	0	1	0	0	0	0

3.4.7 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.

[Description example]

In the case of PUSH DE

Instruction code	1	0	1	0	1	0	1	0

[MEMO]

CHAPTER 4 PORT FUNCTIONS

4.1 Functions of Ports

The μPD789014 Subseries provides the ports shown in Figure 4-1, enabling various methods of control.

Numerous other functions are provided that can be used in addition to the digital I/O port function. For more information on these additional functions, refer to **CHAPTER 2 PIN FUNCTIONS**.

Figure 4-1. Port Types

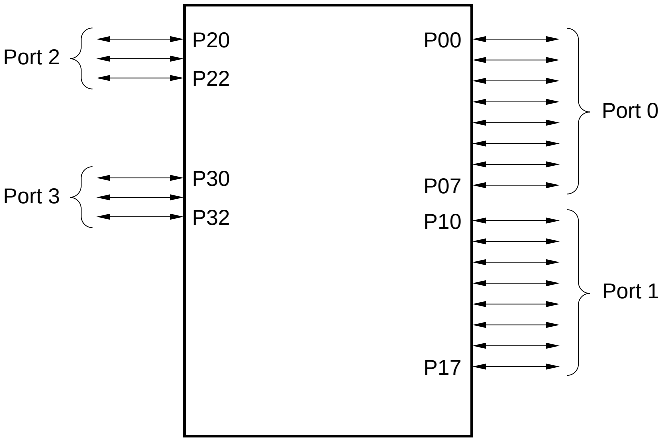


Table 4-1. Port Functions

Pin Name	Input/Output	Function	After Reset	Alternate Function
P00 to P07	Input/output	Port 0 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register (PUO). LEDs can be driven directly.	Input	—
P10 to P17	Input/output	Port 1 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register (PUO). LEDs can be driven directly.	Input	—
P20	Input/output	Port 2 3-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register (PUO). LEDs can be driven directly.	Input	ASCK/$\overline{\text{SCK0}}$
P21				TxD/SO0
P22				RxD/SI0
P30	Input/output	Port 3 3-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register (PUO). LEDs can be driven directly.	Input	INTP0/TI0/TO0
P31				INTP1/TI1/TO1
P32				INTP2

4.2 Port Configuration

Ports have the following hardware configuration.

Table 4-2. Port Configuration

Item	Configuration
Control register	Port mode register (PMm: m = 0 to 3) Pull-up resistor option register (PUO)
Port	Total: 22 (Input/output: 22)
Pull-up resistor	Total: 22 (internal pull-up resistor can be connected by software)

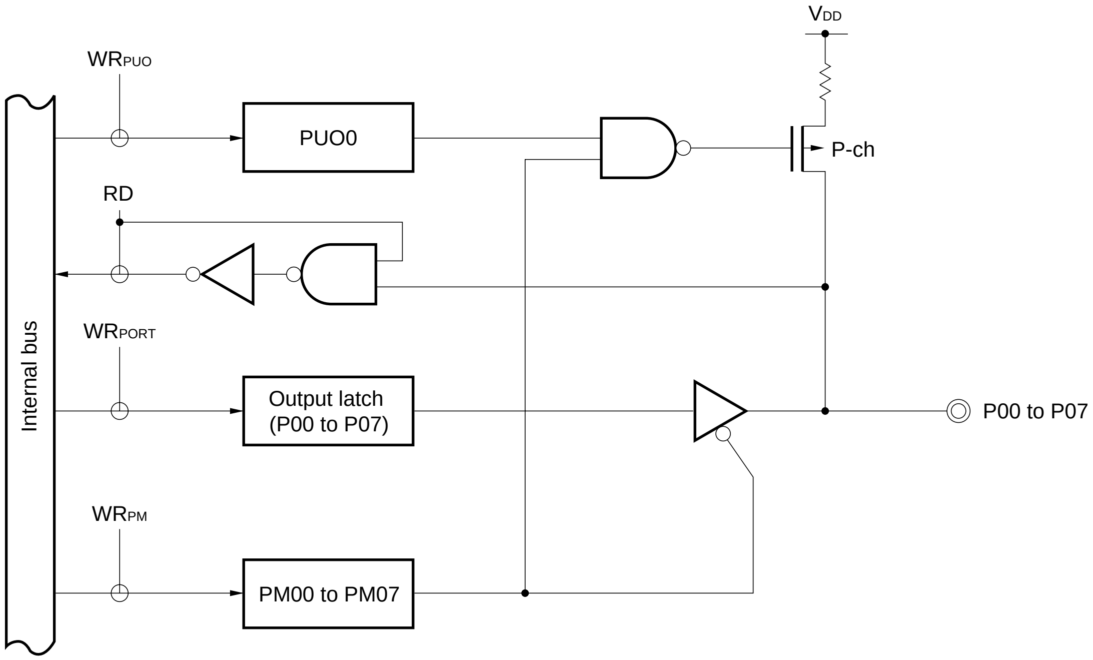
4.2.1 Port 0

This is an 8-bit I/O port with output latch. Port 0 can be specified in the input or output mode in 1-bit units by using the port mode register 0 (PM0). When P00 to P07 pins are used as input port pins, on-chip pull-up resistors can be connected in 8-bit units by using the pull-up resistor option register (PUO).

Port 0 is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

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

Figure 4-2. Block Diagram of P00 to P07



Remark PUO : pull-up resistor option register
PM : port mode register
RD : port 0 read signal
WR : port 0 write signal

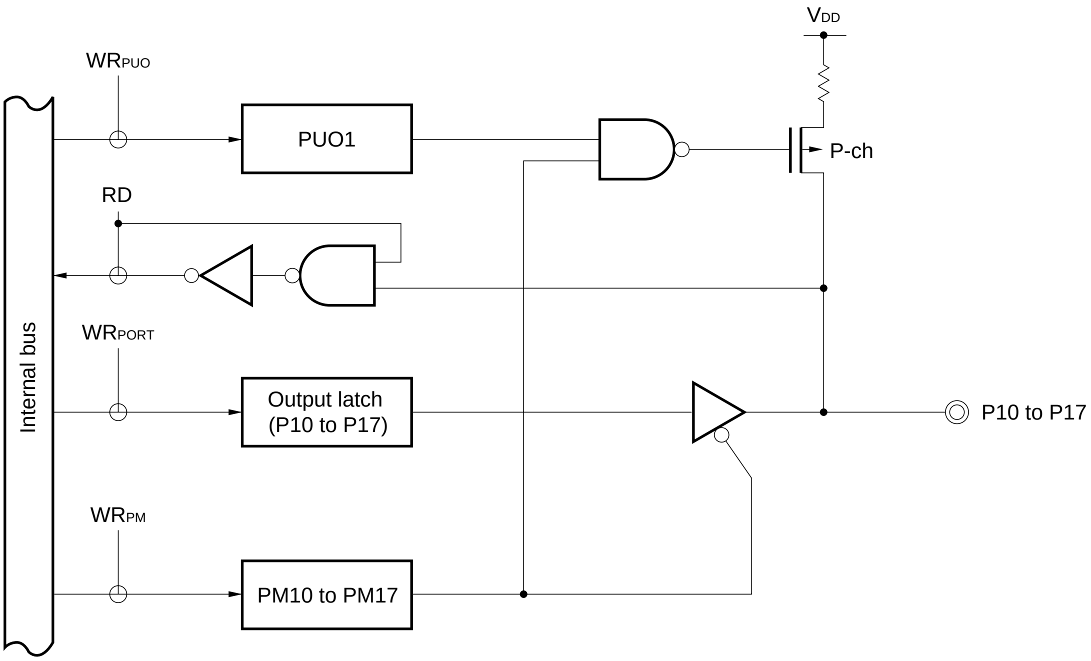
4.2.2 Port 1

This is an 8-bit I/O port with output latch. It can be specified in the input or output mode in 1-bit units by using the port mode register 1 (PM1). When using P10 to P17 pins as input port pins, on-chip pull-up resistors can be connected in 8-bit units by using the pull-up resistor option register (PUO).

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-3 shows the block diagram of port 1.

Figure 4-3. Block Diagram of P10 to P17



Remark	PUO	: pull-up resistor option register
	PM	: port mode register
	RD	: port 1 read signal
	WR	: port 1 write signal

4.2.3 Port 2

This is a 3-bit I/O port with output latch. Port 2 can be specified in the input or output mode in 1-bit units by using the port mode register 2 (PM2). When using P20 to P22 pins as input port pins, internal pull-up resistors can be connected in 1-bit units by using the pull-up resistor option register (PUO).

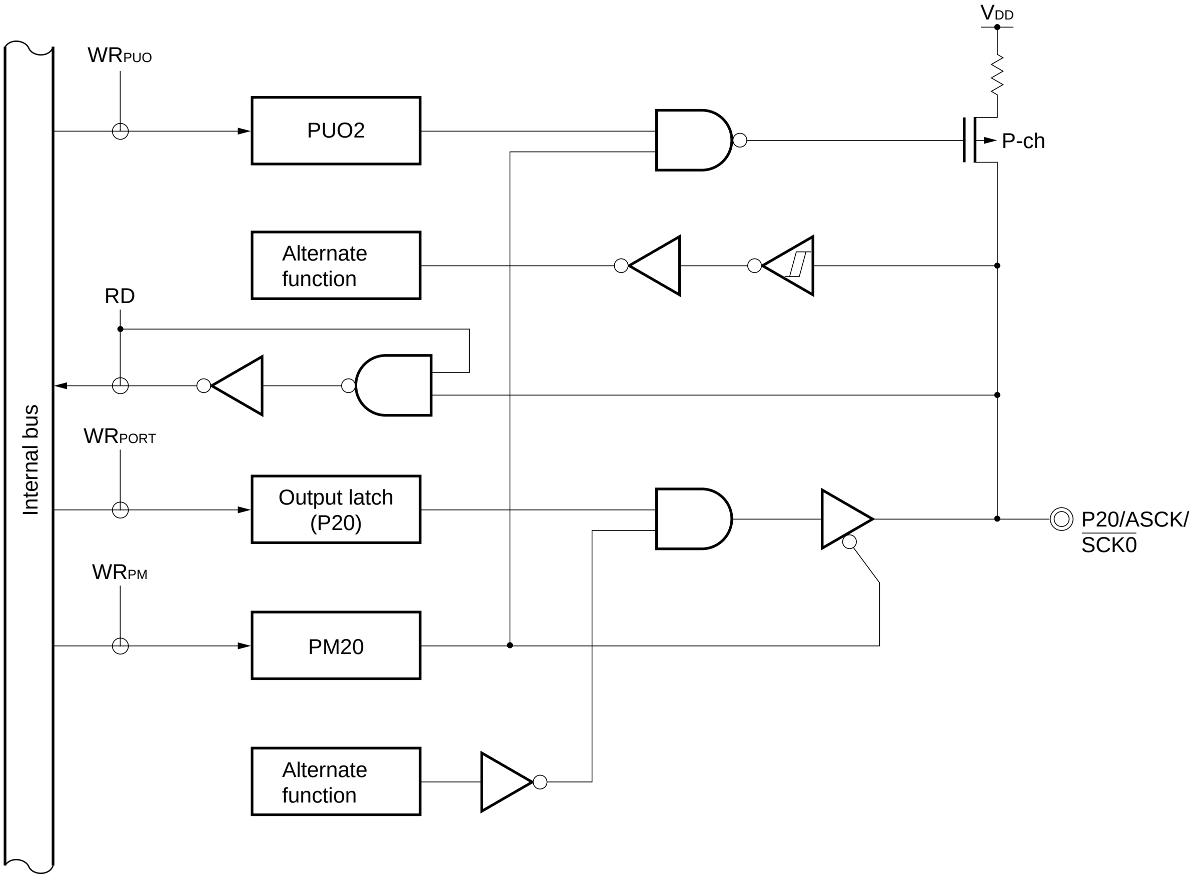
The pins of this port are also used as the data I/O pin of the serial interface.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figures 4-4 through 4-6 show the block diagrams of port 2.

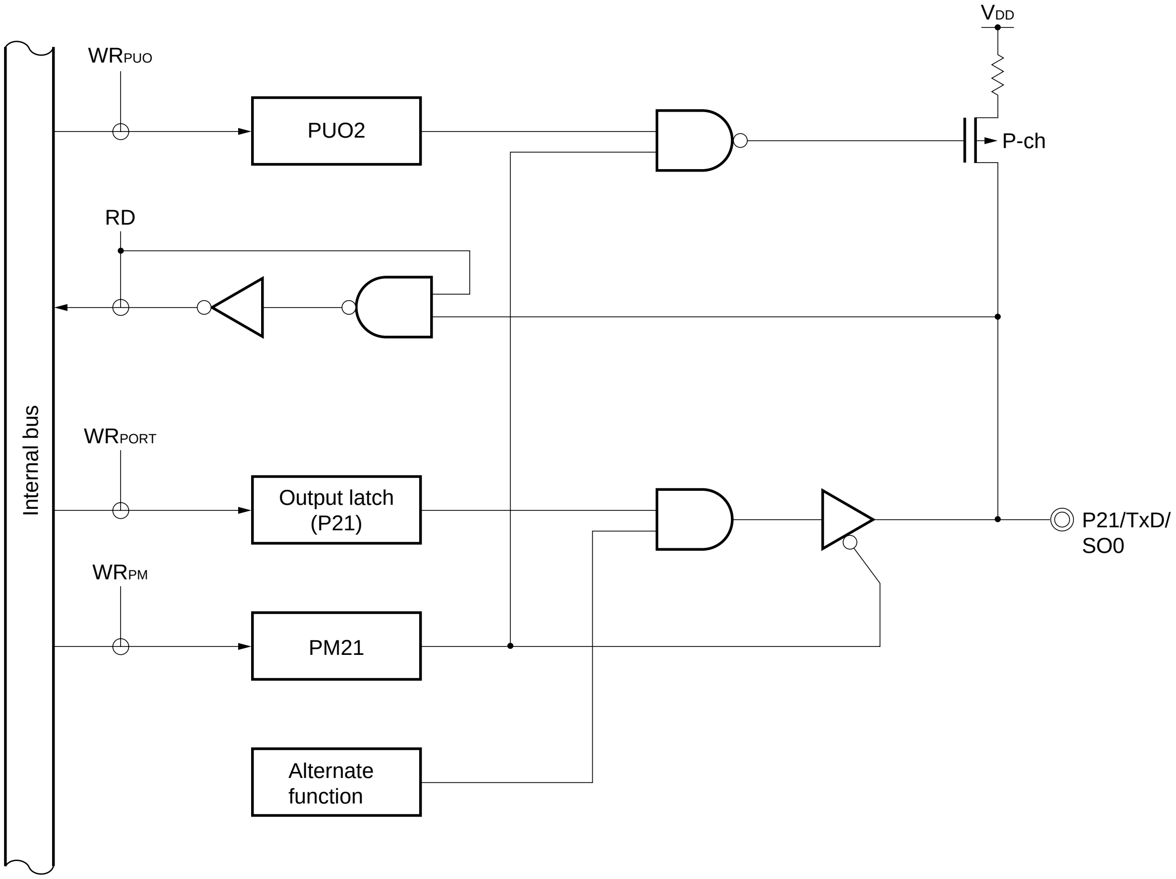
Caution When using the pins of port 2 as the serial interface, the I/O or output latch must be set according to the functions to be used. For the details of the setting, refer to Table 8-2 Serial Interface 00 Operating Mode Settings.

Figure 4-4. Block Diagram of P20



PUO	: pull-up resistor option register
PM	: port mode register
RD	: port 2 read signal
WR	: port 2 write signal

Figure 4-5. Block Diagram of P21



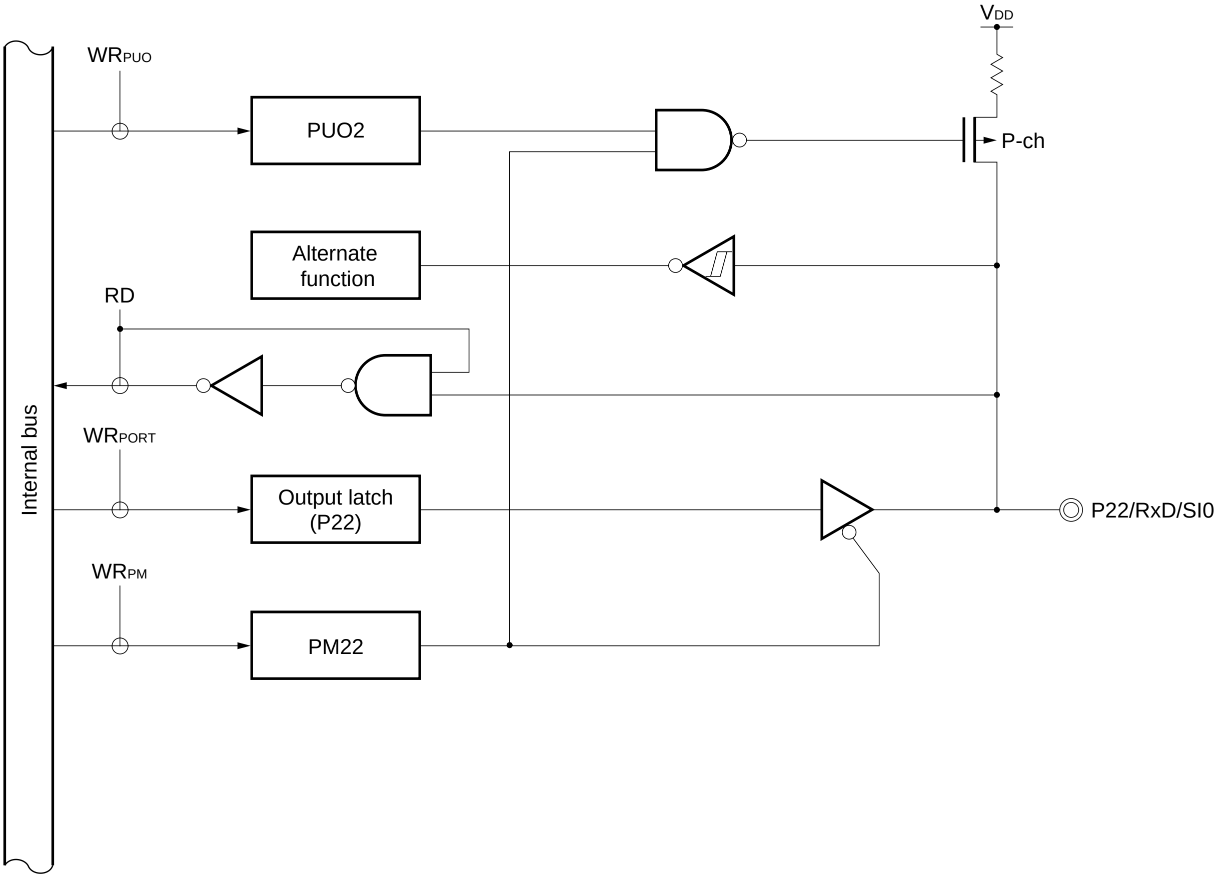
PUO : pull-up resistor option register

PM : port mode register

RD : port 2 read signal

WR : port 2 write signal

Figure 4-6. Block Diagram of P22



PUO : pull-up resistor option register

PM : port mode register

RD : port 2 read signal

WR : port 2 write signal

4.2.4 Port 3

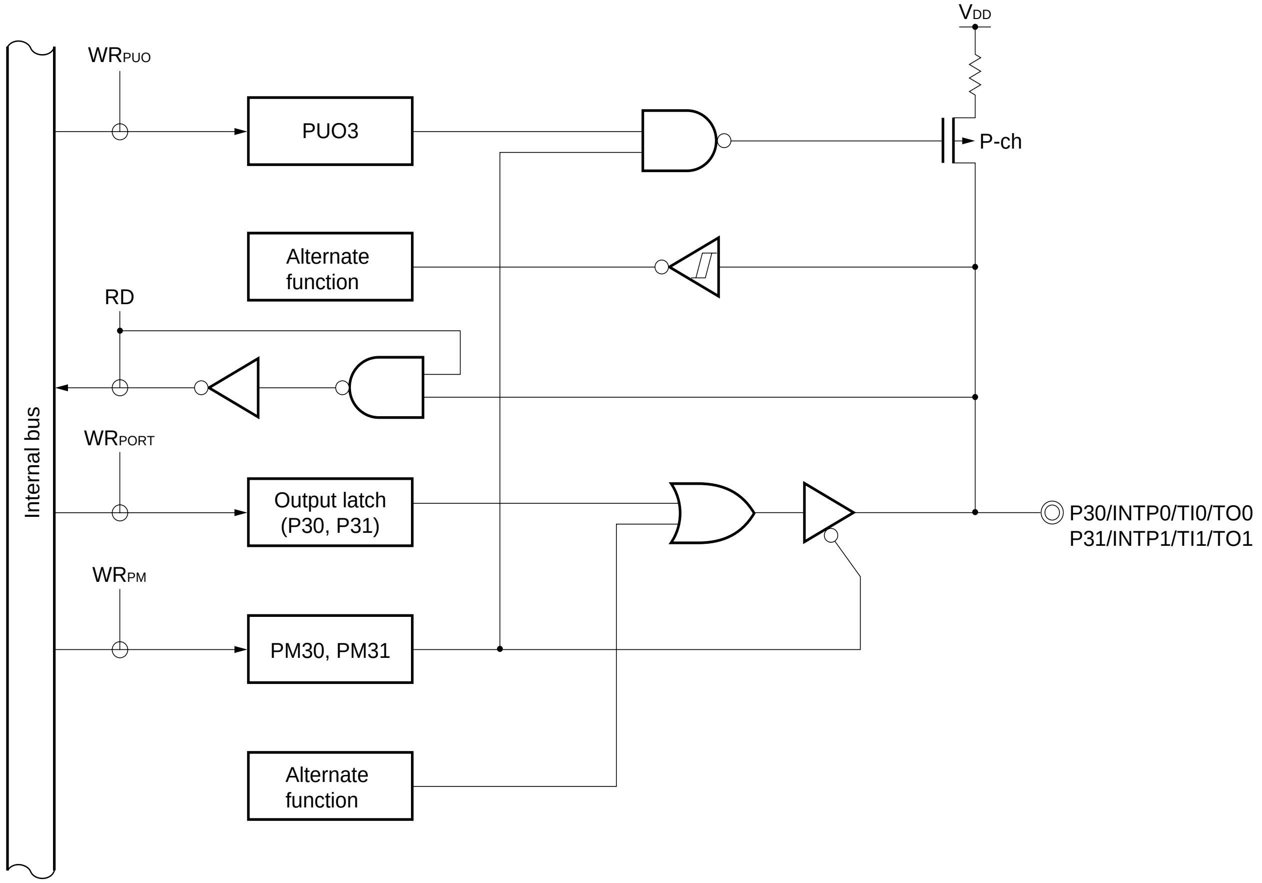
This is a 3-bit I/O port with output latch. Port 3 can be specified in the input or output mode in 1-bit units by using the port mode register 3 (PM3). When using P30 to P32 pins as input port pins, on-chip pull-up resistors can be connected in 3-bit units by using the pull-up resistor option register (PUO).

The pins of this port are also used as the timer I/O and external interrupt pins.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

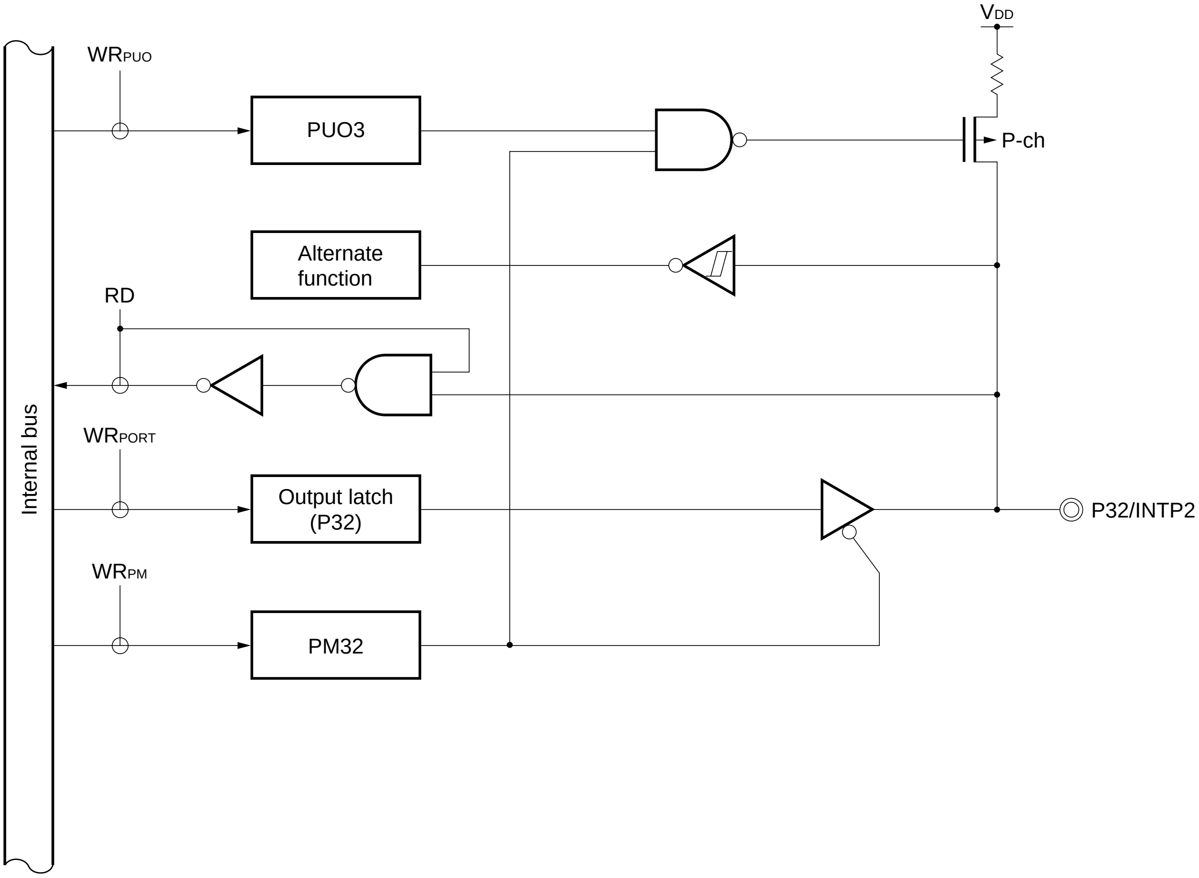
Figures 4-7 and 4-8 show the block diagrams of port 3.

Figure 4-7. Block Diagram of P30 and P31



PUO : pull-up resistor option register
PM : port mode register
RD : port 3 read signal
WR : port 3 write signal

Figure 4-8. Block Diagram of P32



PUO : pull-up resistor option register

PM : port mode register

RD : port 3 read signal

WR : port 3 write signal

4.3 Port Function Control Registers

The following two types of registers control the ports.

- Port mode registers (PM0 to PM3)
- Pull-up resistor option register (PUO)

(1) Port mode registers (PM0 to PM3)

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

PM0 to PM3 are independently set with a 1-bit or 8-bit memory manipulation instruction.

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

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

Caution As port 3 has an alternate function as external interrupt input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. When the output mode is used, therefore, the interrupt mask flag should be set to 1 beforehand.

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

Pin Name	Alternate Function		PM××	P××
	Name	Input/Output		
P30	INTP0	Input	1	×
	TI0	Input	1	×
	TO0	Output	0	0
P31	INTP1	Input	1	×
	TI1	Input	1	×
	TO1	Output	0	0
P32	INTP2	Input	1	×

Caution When using port 2 as serial interface pins, the I/O or output latch must be set according to the functions to be used. For the details of the setting, refer to Table 8-2 Serial Interface 00 Operating Mode Settings.

Remark		
	×	: don't care
	PM××	: port mode register
	P××	: port output latch

Figure 4-9. Port Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FF21H	FFH	R/W
PM2	1	1	1	1	1	PM22	PM21	PM20	FF22H	FFH	R/W
PM3	1	1	1	1	1	PM32	PM31	PM30	FF23H	FFH	R/W

PMmn	Pmn Pin Input/Output Mode Selection (m = 0 to 3, n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(2) Pull-up resistor option register (PUO)

The pull-up resistor option register (PUO) sets whether an on-chip pull-up resistor on each port is used or not. On the port which is specified to use the on-chip pull-up resistor in the PUO, the pull-up resistor can be internally used only for the bits set in the input mode. No on-chip pull-up resistors can be used in the bits set in the output mode in spite of setting the PUO. On-chip pull-up resistors cannot be used even when the pins are used as the alternate-function output pins.

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

$\overline{\text{RESET}}$ input sets the PUO to 00H.

Figure 4-10. Pull-Up Resistor Option Register Format

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	After Reset	R/W
PUO	0	0	0	0	PUO3	PUO2	PUO1	PUO0	FFF7H	00H	R/W

PUOm	Pm On-chip Pull-up Resistor Selection (m = 0 to 3)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

4.4 Operation of Port Functions

The operation of a port differs depending on whether the port is set in the input or output mode, as described below.

4.4.1 Writing to I/O port

(1) In output mode

A value can be written to the output latch by using a transfer instruction. The contents of the output latch can be output from the pins of the port.

The data once written to the output latch is retained until new data is written to the output latch.

Caution Executing a bit manipulation instruction or a logic operation instruction for the port with alternate-output functions may fix the output pin level. Therefore, use an 8-bit data transfer instruction or a 16-bit data transfer instruction.

(2) In input mode

A value can be written to the output latch by using a transfer instruction. However, the status of the pin is not changed because the output buffer is OFF.

The data once written to the output latch is retained until new data is written to the output latch.

Caution A 1-bit memory manipulation instruction is executed to manipulate 1 bit of a port. However, this instruction accesses the port in 8-bit units. When this instruction is executed to manipulate a bit of an input/output port, therefore, the contents of the output latch of the pin that is set in the input mode and not subject to manipulation become undefined.

4.4.2 Reading from I/O port

(1) In output mode

The status of a pin can be read by using a transfer instruction. The contents of the output latch are not changed.

(2) In input mode

The status of a pin can be read by using a transfer instruction. The contents of the output latch are not changed.

4.4.3 Arithmetic operation of I/O port

(1) In output mode

An arithmetic operation can be performed with the status of a pin. The result of the operation is written to the output latch. The contents of the output latch are output from the port pins.

The data once written to the output latch is retained until new data is written to the output latch.

(2) In input mode

The contents of the output latch become undefined. However, the status of the pin is not changed because the output buffer is OFF.

Caution A 1-bit memory manipulation instruction is executed to manipulate 1 bit of a port. However, this instruction accesses the port in 8-bit units. When this instruction is executed to manipulate a bit of an input/output port, therefore, the contents of the output latch of the pin that is set in the input mode and not subject to manipulation become undefined.

CHAPTER 5 CLOCK GENERATOR

5.1 Function of Clock Generator

The clock generator generates the clock to be supplied to the CPU and peripheral hardware. The system clock oscillator consists of the following type.

- System clock oscillator
  This circuit oscillates at frequencies of 1.0 to 5.0 MHz. Oscillation can be stopped by executing the STOP instruction.

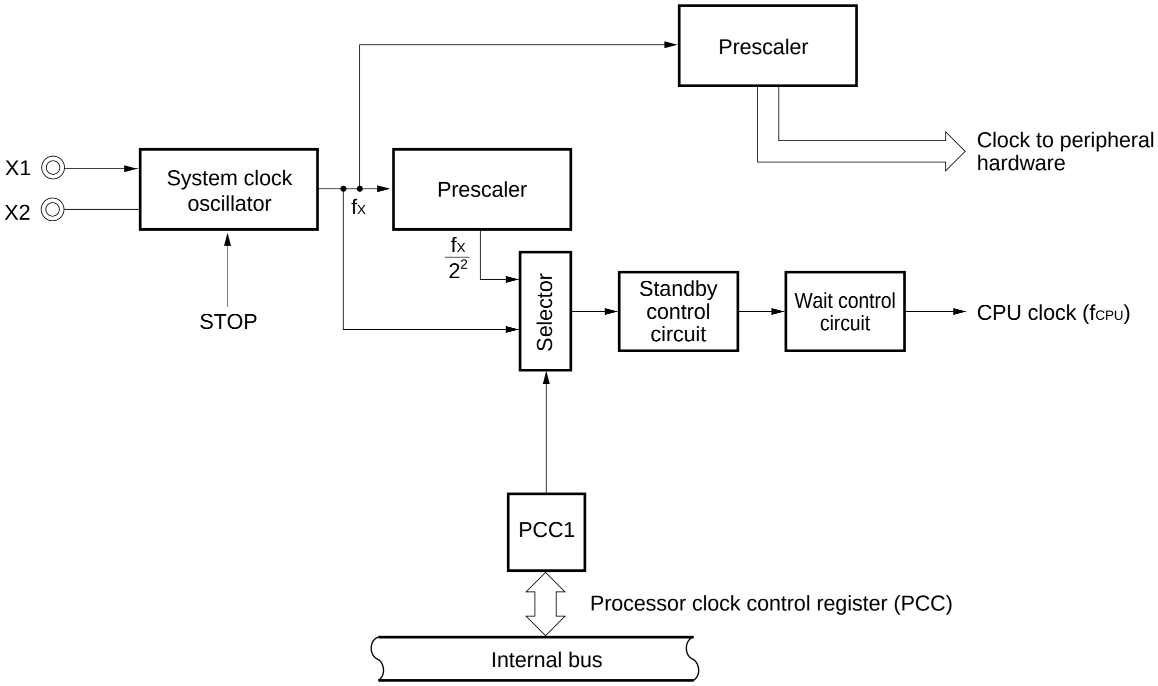
5.2 Configuration of Clock Generator

The clock generator consists of the following hardware.

Table 5-1. Configuration of Clock Generator

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

Figure 5-1. Block Diagram of Clock Generator



5.3 Clock Generator Control Register

The clock generator is controlled by the following register:

- Processor clock control register (PCC)

(1) Processor clock control register (PCC)

The PCC sets CPU clock selection and the ratio of division.
The PCC is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the PCC to 02H.

Figure 5-2. Processor Clock Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PCC	0	0	0	0	0	0	PCC1	0	FFFBH	02H	R/W

PCC1	CPU Clock (f_{CPU}) Selection
0	f_X (0.2 μs)
1	$f_X/2^2$ (0.8 μs)

Caution Bit 0 and bits 2 to 7 must be set to 0.

Remarks **1.** f_X: system clock oscillation frequency

2. Value in parentheses is when operating at f_X = 5.0 MHz.

3. Minimum instruction execution time: $2f_{CPU}$

- When f_{CPU} = 0.2 μs: 0.4 μs
- When f_{CPU} = 0.8 μs: 1.6 μs

5.4 System Clock Oscillator

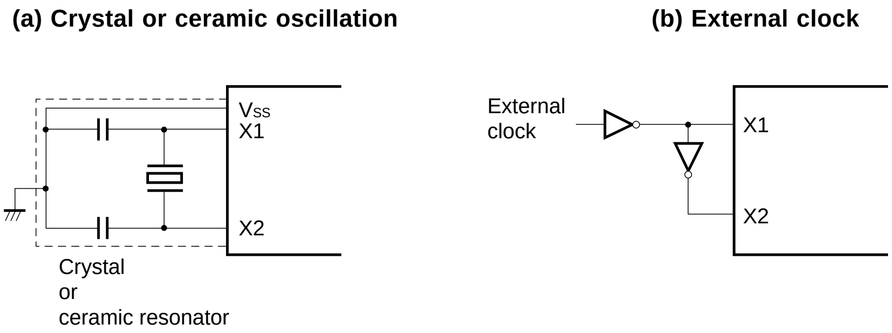
5.4.1 System clock oscillator

The system clock oscillator is oscillated by the crystal or ceramic resonator (5.0 MHz TYP.) connected across the X1 and X2 pins.

An external clock can also be input to the circuit. In this case, input the clock signal to the X1 pin, and input the reversed signal to the X2 pin.

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

Figure 5-3. External Circuit of System Clock Oscillator



(a) Crystal or ceramic oscillation

(b) External clock

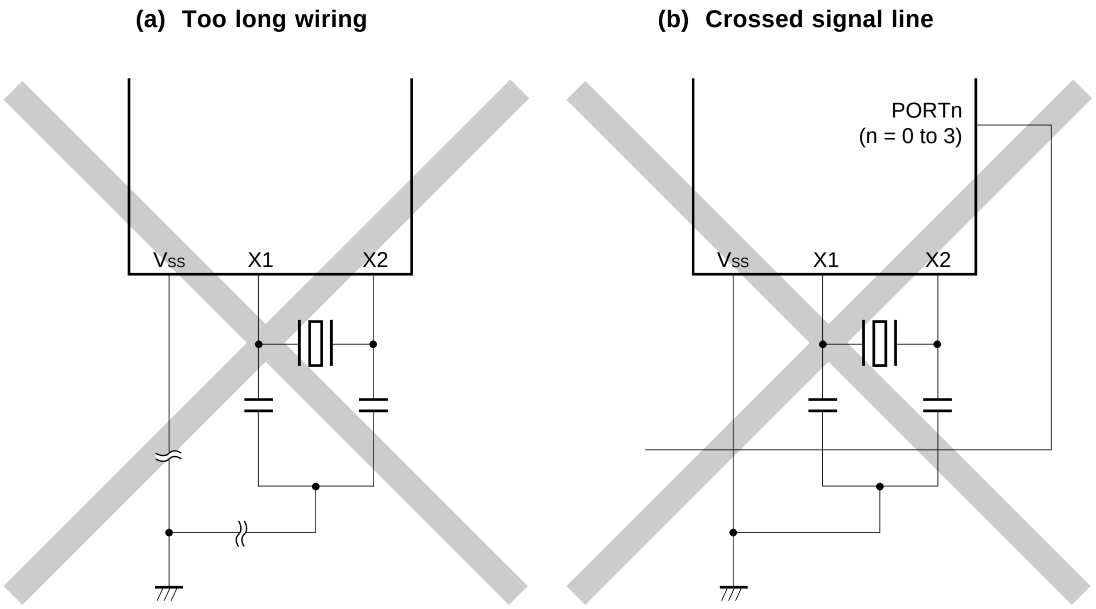
Cautions 1. While an external clock is input to the circuit, do not execute the STOP instruction. Doing so stops the system clock operation and pulls up the X2 pin to V_{DD}.

2. When using the system clock oscillator, wire the area enclosed by the broken line in Figure 5-3 as follows to avoid an adverse effect from wiring capacitance:

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always keep the ground point of the oscillator capacitor to the same potential as V_{SS}. Do not ground the capacitor to a ground pattern in which a high current flows.**
- **Do not fetch signals from the oscillator.**

Figure 5-4 shows incorrect examples of resonator connection.

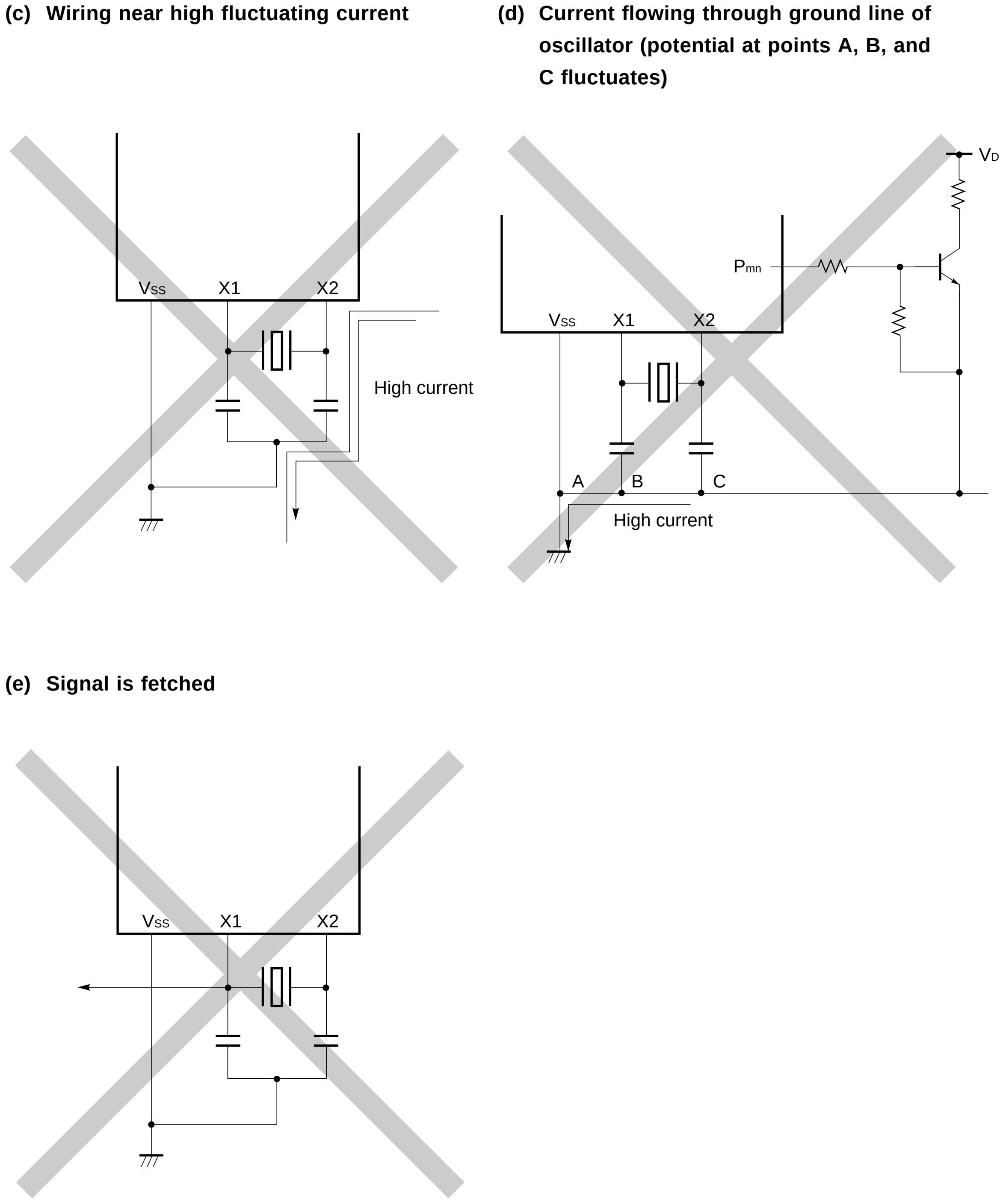
Figure 5-4. Incorrect Examples of Resonator Connection (1/2)



(a) Too long wiring

(b) Crossed signal line

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



(c) Wiring near high fluctuating current

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

(e) Signal is fetched

5.4.2 Divider circuit

The divider circuit divides the output of the system clock oscillator (fX) to generate various clocks.

5.5 Operation of Clock Generator

The clock generator generates the following clocks and controls the operation modes of the CPU, such as the standby mode:

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

The operation of the clock generator is determined by the processor clock control register (PCC), as follows:

(a) The low-speed mode $2f_{CPU}$ (1.6 μs: at 5.0-MHz operation) of the system clock is selected when the $\overline{\text{RESET}}$ signal is generated (PCC = 02H). While a low level is input to the $\overline{\text{RESET}}$ pin, oscillation of the system clock is stopped.

(b) Two types of CPU clocks f_{CPU} (0.2 μs and 0.8 μs: at 5.0-MHz operation) can be selected by the PCC setting.

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

(d) The clock to the peripheral hardware is supplied by dividing the system clock. The other peripheral hardware is stopped when the system clock is stopped (except, however, the external clock input operation).

5.6 Changing Setting of CPU Clock

5.6.1 Time required for switching CPU clock

The CPU clock can be selected by using bit 1 (PCC1) of the processor clock control register (PCC).

Actually, the specified clock is not selected immediately after the setting of PCC has been changed, and the old clock is used for the duration of several instructions after that (refer to **Table 5-2**).

Table 5-2. Maximum Time Required for Switching CPU Clock

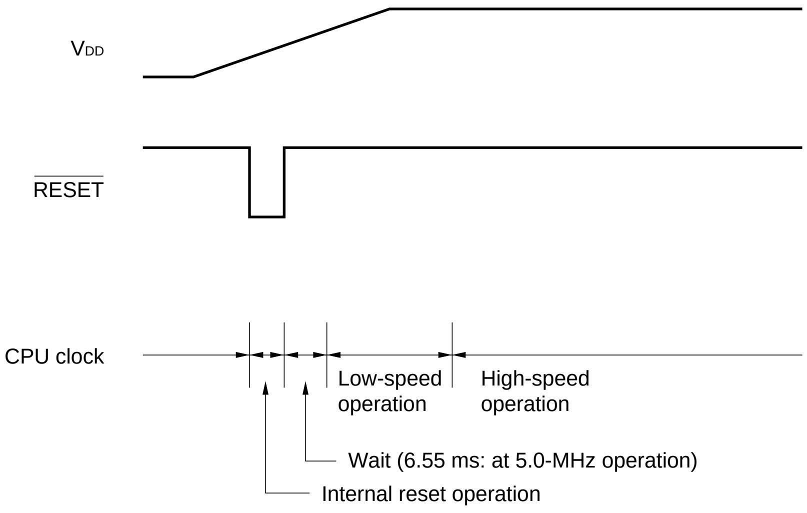
Set Value before Switching	Set Value after Switching	
PCC1	PCC1	PCC1
	0	1
0		4 clocks
1	2 clocks	

Remark Two clocks are the minimum instruction execution time of the CPU clock before switching.

5.6.2 Switching CPU clock

The following figure illustrates how the CPU clock switches.

Figure 5-5. Switching CPU Clock



<1> The CPU is reset when the $\overline{\text{RESET}}$ pin is made low on power application. The effect of resetting is released when the $\overline{\text{RESET}}$ pin is later made high, and the system clock starts oscillating. At this time, the time during which oscillation stabilizes ($2^{15}/f_X$) is automatically secured.
After that, the CPU starts instruction execution at the low speed of the system clock (1.6 μs: at 5.0-MHz operation).

<2> After the time during which the V_{DD} voltage rises to the level at which the CPU can operate at the highest speed has elapsed, the processor clock control register (PCC) is rewritten so that the high speed can be selected.

[MEMO]

CHAPTER 6 8-BIT TIMER/EVENT COUNTER

The μPD789014 Subseries provide the following on-chip timers:

(1) 8-bit timer/event counters (TM00 and TM01)

These counters can be used as interval timers, external event counters, and for output of square waves of any frequency.

(2) Watchdog timer (WDTM)

The watchdog timer can also be used to generate a non-maskable interrupt, maskable interrupt, or $\overline{\text{RESET}}$ signal at any intervals set in advance (refer to **CHAPTER 7 WATCHDOG TIMER**).

6.1 Functions of 8-Bit Timer/Event Counter

The 8-bit timer/event counters have the following functions:

- Interval timer
- External event counter
- Square wave output

(1) 8-bit interval timer

When the 8-bit timer/event counter is used as an interval timer, it generates an interrupt at any time intervals set in advance.

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

Minimum Interval Time	Maximum Interval Time	Resolution
1/fx (200 ns)	2^8/fx (51.2 μs)	1/fx (200 ns)
2^5/fx (6.4 μs)	2^{13}/fx (1.64 ms)	2^5/fx (6.4 μs)

Remarks 1. fx : system clock oscillation frequency

2. ( ) : at fx = 5.0-MHz operation

(2) External event counter

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

(3) Square wave output

A square wave of any frequency can be output.

Table 6-2. Square Wave Output Range of 8-Bit Timer/Event Counter

Minimum Pulse Width	Maximum Pulse Width	Resolution
1/fx (200 ns)	2^8/fx (51.2 μs)	1/fx (200 ns)
2^5/fx (6.4 μs)	2^{13}/fx (1.64 ms)	2^5/fx (6.4 μs)

Remarks 1. fx : system clock oscillation frequency

2. ( ) : at fx = 5.0-MHz operation

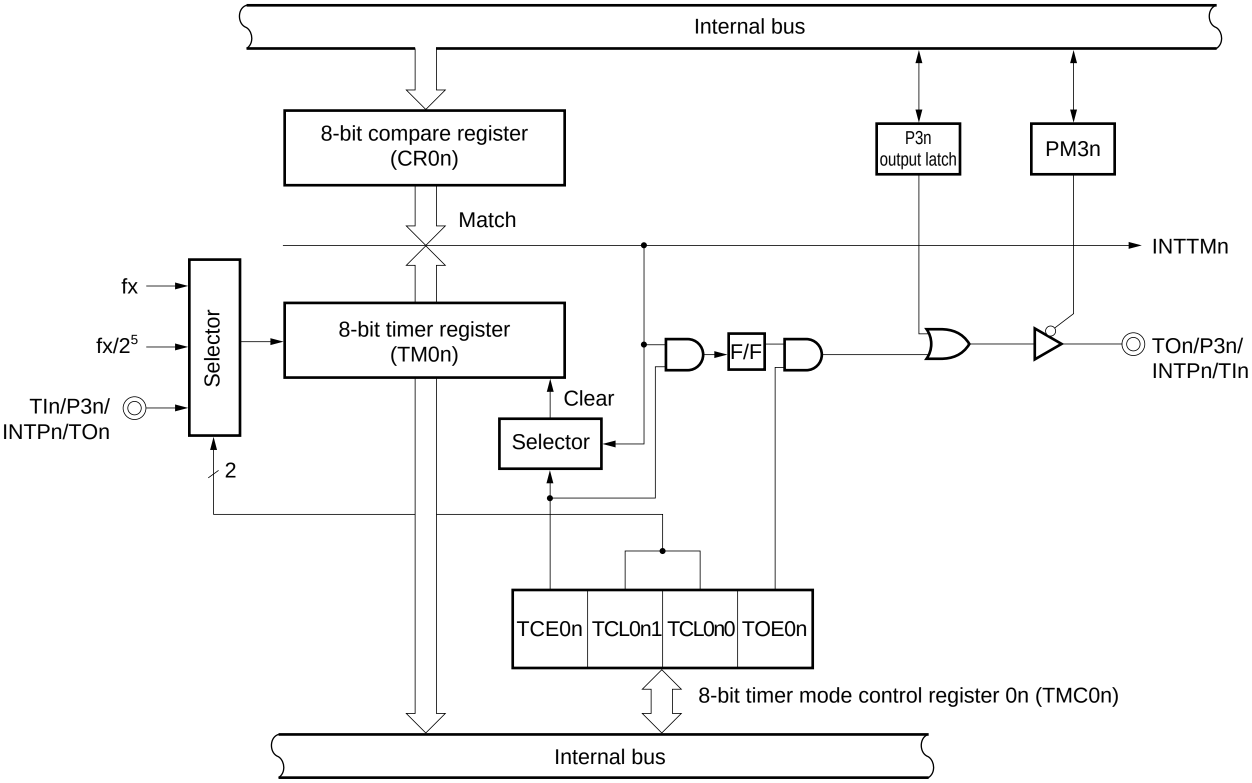
6.2 8-Bit Timer/Event Counter Configuration

The 8-bit timer/event counter consists of the following hardware configuration.

Table 6-3. 8-Bit Timer/Event Counter Configuration

Item	Configuration
Timer register	8 bits × 2 (TM00, TM01)
Register	Compare register: 8 bits × 2 (CR00, CR01)
Timer output	2 (TO0, TO1)
Control register	8-bit timer mode control registers 00, 01 (TMC00, TMC01) Port mode register 3 (PM3)

Figure 6-1. Block Diagram of 8-Bit Timer/Event Counter



n = 0, 1

(1) 8-bit compare register 0n (CR0n)

An 8-bit register to compare the value set to CR0n with the 8-bit timer register 0n (TM0n) count value, and if they match, generate an interrupt request (INTTMn).

CR0n is set with an 8-bit memory manipulation instruction. The 00H to FFH values can be set.

$\overline{\text{RESET}}$ input sets CR0n to undefined.

★ **Caution Be sure to rewrite the CR0n value after stopping timer operation.**

Remark n = 0, 1

(2) 8-bit timer register 0n (TM0n)

This is 8-bit register to count count pulses.

TM0n is read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TM0n to 00H.

Remark n = 0, 1

6.3 8-Bit Timer/Event Counter Control Registers

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

- 8-bit timer mode control registers 00, 01 (TMC00, TMC01)
- Port mode register 3 (PM3)

(1) 8-bit timer mode control register 00 (TMC00)

This register enables/stops operation of 8-bit timer register 00 (TM00), sets the count clock of 8-bit timer/event counter 00, and controls the operation of the output control circuit.

TMC00 is set with a 1-bit or 8-bit memory manipulation instruction.

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

★ **Figure 6-2. 8-Bit Timer Mode Control Register 00 Format**

Symbol	<7>	6	5	4	3	2	1	<0>	Address	After Reset	R/W
TMC00	TCE00	0	0	0	0	TCL001	TCL000	TOE00	FF53H	00H	R/W

TCE00	8-Bit Timer Register 00 Operation Control
0	Operation stop (TM00 cleared to 0)
1	Operation enable

TCL001	TCL000	8-Bit Timer Register 00 Count Clock Selection
0	0	f_X (5.0 MHz)
0	1	$f_X/2^5$ (156 kHz)
1	0	Rising edge of TI0[Note]
1	1	Falling edge of TI0[Note]

TOE00	8-Bit Timer/Event Counter 00 Output Control
0	Output disable (port mode)
1	Output enable

Note When clock is externally input, timer output cannot be used.

Caution Be sure to select the count clock after stopping timer operation (TCE00 = 0). For details, refer to 6.4 Operation of 8-Bit Timer/Event Counter.

Remarks **1.** f_X : system clock oscillation frequency

2. ( ) : at f_X = 5.0-MHz operation

(2) 8-bit timer mode control register 01 (TMC01)

This register enables/stops operation of 8-bit timer register 01 (TM01), sets the count clock of 8-bit timer/event counter 01, and controls the operation of the output control circuit.

TMC01 is set with a 1-bit or 8-bit memory manipulation instruction.

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

★ **Figure 6-3. 8-Bit Timer Mode Control Register 01 Format**

Symbol	<7>	6	5	4	3	2	1	<0>	Address	After Reset	R/W
TMC01	TCE01	0	0	0	0	TCL011	TCL010	TOE01	FF57H	00H	R/W

TCE01	8-Bit Timer Register 01 Operation Control
0	Operation stop (TM01 cleared to 0)
1	Operation enable

TCL011	TCL010	8-Bit Timer Register 01 Count Clock Selection
0	0	f_X (5.0 MHz)
0	1	$f_X/2^5$ (156 kHz)
1	0	Rising edge of TI1$^{\text{Note}}$
1	1	Falling edge of TI1$^{\text{Note}}$

TOE01	8-Bit Timer/Event Counter 01 Output Control
0	Output disable (port mode)
1	Output enable

Note When clock is externally input, timer output cannot be used.

Caution Be sure to select the count clock after stopping timer operation (TCE01 = 0). For details, refer to 6.4 Operation of 8-Bit Timer/Event Counter.

Remarks **1.** f_X : system clock oscillation frequency

2. ( ) : at f_X = 5.0-MHz operation

(3) Port mode register 3 (PM3)

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

When using the P30/INTP0/TI0/TO0 and P31/INTP1/TI1/TO1 pins for timer output, set PM30, PM31 and the output latch of P30, P31 to 0.

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

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

Figure 6-4. Port Mode Register 3 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM3	1	1	1	1	1	PM32	PM31	PM30	FF23H	FFH	R/W

PM3n	P3n Pin Input/Output Mode Selection (n = 0 to 2)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

6.4 Operation of 8-Bit Timer/Event Counter

★ 6.4.1 Operation as interval timer

The interval timer repeatedly generates an interrupt at time intervals specified by the count value set to the 8-bit compare register 00 and 01 (CR00 and CR01) in advance.

To operate the 8-bit timer/event counter as an interval timer, the settings are required in the following order.

<1> Set the 8-bit timer register 0n (TM0n) to operation disable (TCE0n (bit 7 of 8-bit timer mode control register 0n (TMC0n)) = 0).
<2> Set count clock of 8-bit timer/event counter (refer to **Table 6-4**).
<3> Set count value to CR0n.
<4> Set TM0n to operation enable (TCE0n = 1).

When the count value of the 8-bit timer register 0n (TM0n) matches the value set to CR0n, the value of TM0n is cleared to 0 and TM0n continue counting. At the same time, an interrupt request signal (INTTMn) is generated.

Table 6-4 shows interval time, and Figure 6-5 shows the timing of interval timer operation.

Caution When the count clock and TM0n operation enable are set simultaneously with TMC0n using the 8-bit memory manipulation instruction, the error of a cycle from which a timer has been started may become one clock or more. Therefore, settings must be done in the above order to operate the 8-bit timer/event counter as an interval timer.

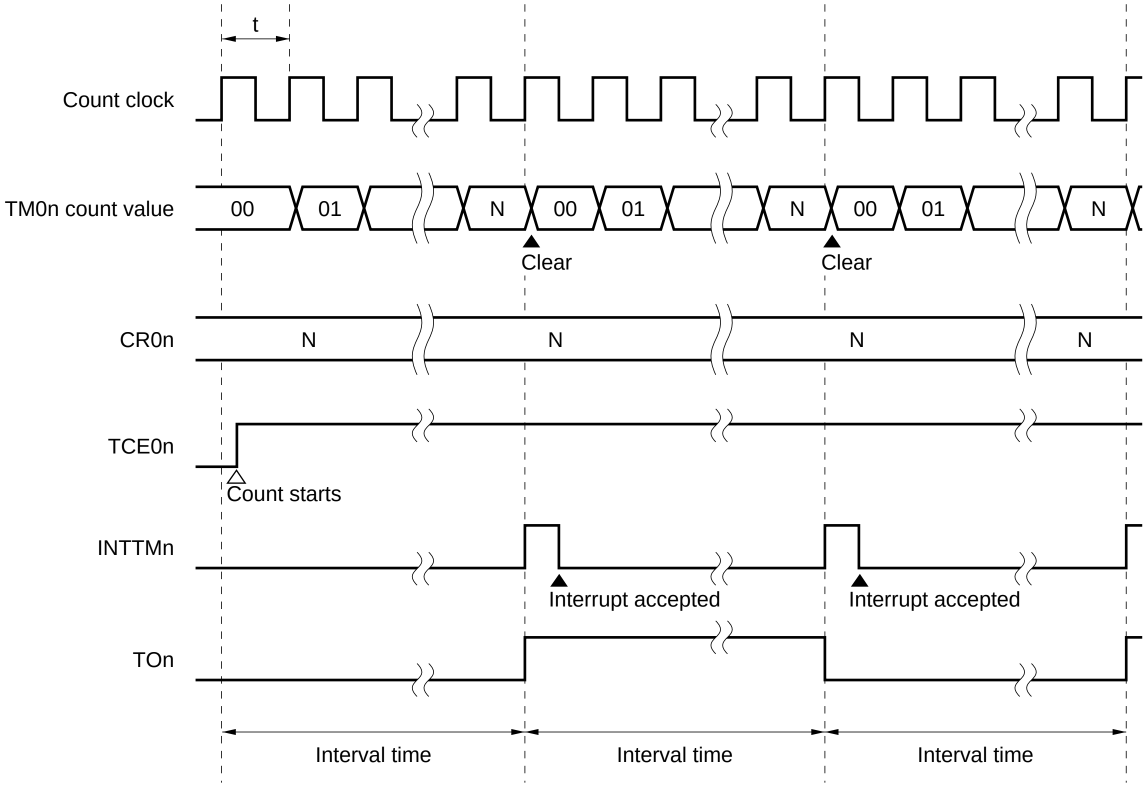
Remark n = 0, 1

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

TCL0n1	TCL0n0	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	1/fx (200 ns)	2^8/fx (51.2 µs)	1/fx (200 ns)
0	1	2^5/fx (6.4 µs)	2^{13}/fx (1.64 ms)	2^5/fx (6.4 µs)
1	0	TIn input cycle	2^8 × TIn input cycle	TIn input edge cycle
1	1	TIn input cycle	2^8 × TIn input cycle	TIn input edge cycle

Remarks 1. fx : system clock oscillation frequency
2. ( ) : at fx = 5.0-MHz operation
3. n = 0, 1

Figure 6-5. Interval Timer Operation Timing



Remarks **1.** Interval time = (N + 1) × t : N = 00H to FFH

2. n = 0, 1

★ 6.4.2 Operation as external event counter

The external event counter counts the number of external clock pulses input to the TI0/P30/INTP0/TO0 and TI1/P31/INTP1/TO1 pins by using the timer register 00 and 01 (TM00 and TM01).

To operate the 8-bit timer/event counter as an external event counter, the settings are required in the following order.

<1> Set P30 and P31 to input mode (PM30 = 1, PM31 = 1).

<2> Set the 8-bit timer register 0n (TM0n) to operation disable (TCE0n (bit 7 of 8-bit timer mode control register 0n (TMC0n)) = 0).

<3> Specify the rising and falling edges of TIn (refer to **Table 6-4**), and set TOn to output disable (TOE0n (bit 0 of TMC0n) = 0).

<4> Set count value to CR0n.

<5> Set TM0n to operation enable (TCE0n = 1).

Each time the valid edge specified by bit 1 (TCL000 and TCL010) of TMC00 and TMC01 is input, the value of the 8-bit timer register 0n (TM00 and TM01) is incremented.

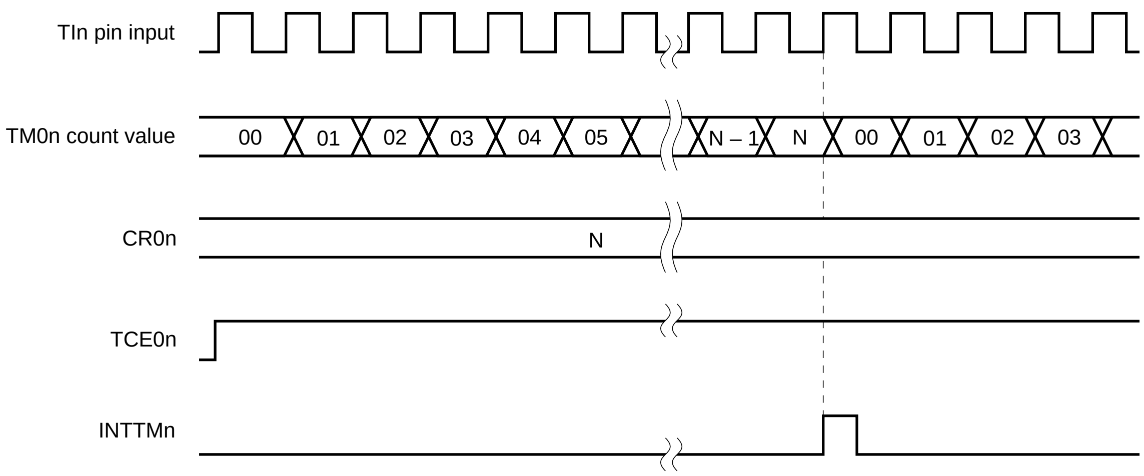
When the count values of TM00 and TM01 match the value set to CR00 and CR01, the values of TM00 and TM01 are cleared to 0 and TM00 and TM01 continue counting. At the same time, an interrupt request signal (INTTM0 and INTTM1) is generated.

Figure 6-6 shows the timing of the external event counter operation (with rising edge specified).

Caution **When the count clock and TM0n operation enable are set simultaneously with TMC0n using the 8-bit memory manipulation instruction, the error of a cycle from which a timer has been started may become one clock or more. Therefore, settings must be done in the above order to operate the 8-bit timer/event counter as an external event counter.**

Remark n = 0, 1

Figure 6-6. External Event Counter Operation Timing (with rising edge specified)



Remarks **1.** N = 00H to FFH

2. n = 0, 1

★ ### 6.4.3 Operation as square wave output

The 8-bit timer/event counter can generate output square waves of a given frequency at intervals specified by the count value set to the 8-bit compare registers 00 and 01 (CR00 and CR01) in advance.

To operate the 8-bit timer/event counter for square wave output, the settings are required in the following order.

<1> Set P30 and P31 to output mode (PM30 = 0, PM31 = 0), and set output latches of P30 and P31 to 0.

<2> Set the 8-bit timer register 0n (TM0n) to operation disable (TCE0n (bit 7 of 8-bit timer mode control register 0n (TMC0n)) = 0).

<3> Set count clock of 8-bit timer/event counter (refer to **Table 6-5**), and set TOn to output enable (TOE0n (bit 0 of TMC0n) = 1).

<4> Set count value to CR0n.

<5> Set TM0n to operation enable (TCE0n = 1).

When the count value of an 8-bit timer register 0n (TM00 and TM01) matches the value set in CR00 and CR01, the TO0/P30/INTP0/TI0 and TO1/P31/INTP1/TI1 pin output will be inverted, respectively. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, the TM00 and TM01 value will be cleared to 0 then resume to count, generating an interrupt request signal (INTTM0 and INTTM1).

Setting 0 to the bit 7 in TMC00 and TMC01, that is, TCE00 and TCE01 clears the square-wave output to 0.

Table 6-5 shows square wave output range, and Figure 6-7 shows timing of square wave output.

Caution When the count clock and TM0n operation enable are set simultaneously with TMC0n using the 8-bit memory manipulation instruction, the error of a cycle from which a timer has been started may become one clock or more. Therefore, settings must be done in the above order to operate the 8-bit timer/event counter for square wave output.

Remark n = 0, 1

Table 6-5. Square Wave Output Range of 8-Bit Timer/Event Counter

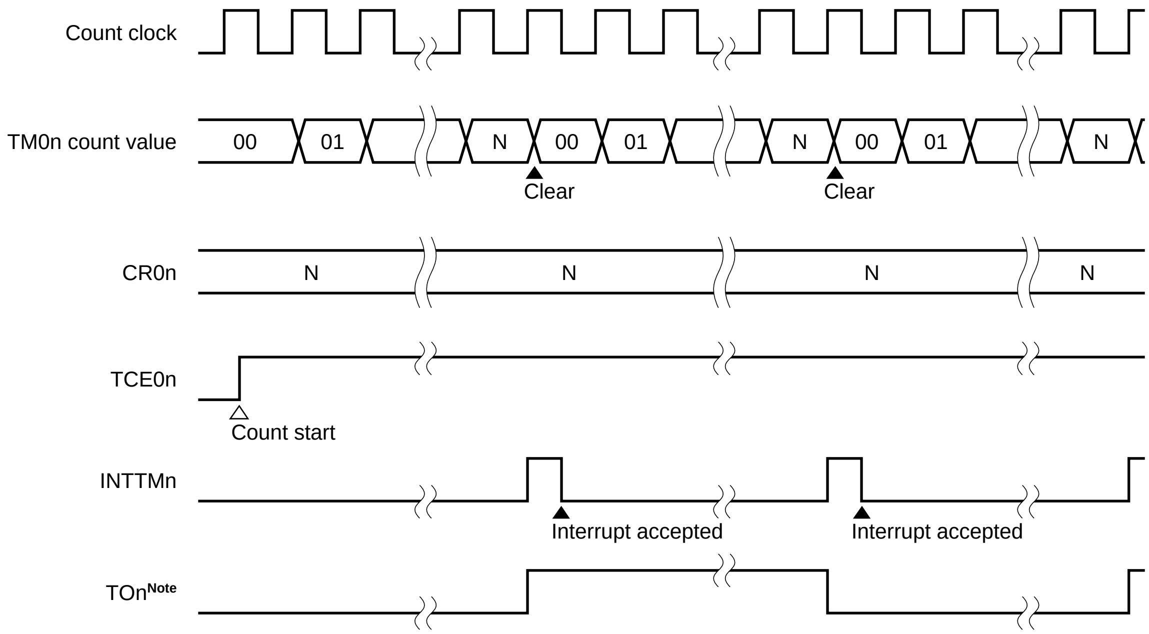
TCL0n1	TCL0n0	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	1/fx (200 ns)	2^8/fx (51.2 μs)	1/fx (200 ns)
0	1	2^5/fx (6.4 μs)	2^{13}/fx (1.64 ms)	2^5/fx (6.4 μs)

Remarks 1. fx : system clock oscillation frequency

2. ( ) : at fx = 5.0-MHz operation

3. n = 0, 1

Figure 6-7. Square Wave Output Timing



Note The initial value of TOn during output enable (TOE0n = 1) becomes low-level.

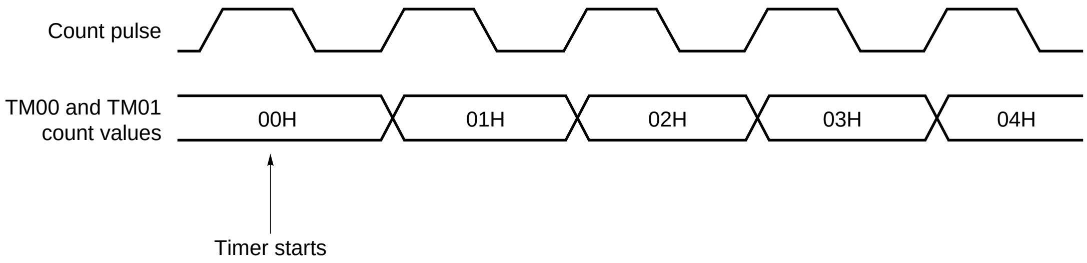
Remark n = 0, 1

6.5 Notes on Using 8-Bit Timer/Event Counters

(1) Error on starting timer

An error of up to 1 clock occurs after the timer has been started until a coincidence signal is generated. This is because the 8-bit timer registers 00 and 01 (TM00 and TM01) are started in asynchronization with the count pulse.

Figure 6-8. Start Timing of 8-Bit Timer Register

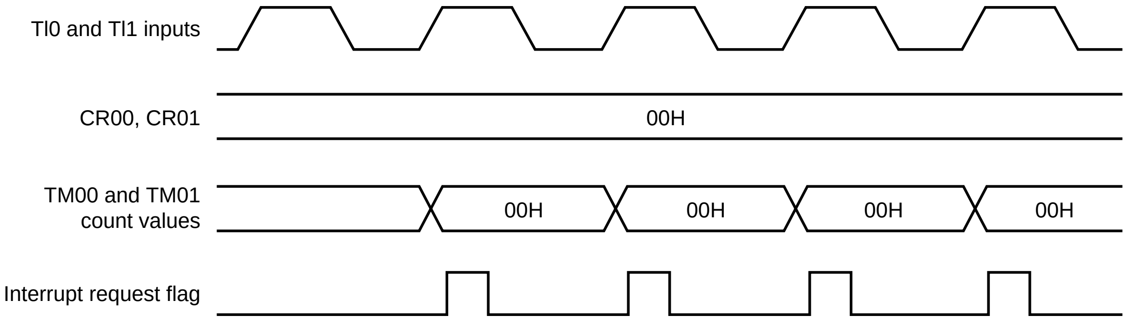


(2) Setting of compare register

The compare registers 00 and 01 (CR00 and CR01) can be set to 00H.

Therefore, one pulse can be counted when an 8-bit timer/event counter operates as an event counter.

Figure 6-9. External Event Counter Operation Timing



Phase-out/Discontinued

[MEMO]

CHAPTER 7 WATCHDOG TIMER

7.1 Functions of Watchdog Timer

The watchdog timer has the following functions:

- Watchdog timer
- Interval timer

Caution Select the watchdog timer mode or interval timer mode by using the watchdog timer mode register (WDTM).

(1) Watchdog timer

The watchdog timer is used to detect program runaway. When a runaway is detected, a non-maskable interrupt or the $\overline{\text{RESET}}$ signal can be generated.

Table 7-1. Runaway Detection Time of Watchdog Timer

Runaway Detection Time	At f_X = 5.0 MHz
$2^{11} \times 1/f_X$	410 μs
$2^{13} \times 1/f_X$	1.64 ms
$2^{15} \times 1/f_X$	6.55 ms
$2^{17} \times 1/f_X$	26.2 ms

f_X: system clock oscillation frequency

(2) Interval timer

The interval timer generates an interrupt at a given interval set in advance.

Table 7-2. Interval Time

Interval Time	At f_X = 5.0 MHz
$2^{11} \times 1/f_X$	410 μs
$2^{13} \times 1/f_X$	1.64 ms
$2^{15} \times 1/f_X$	6.55 ms
$2^{17} \times 1/f_X$	26.2 ms

f_X: system clock oscillation frequency

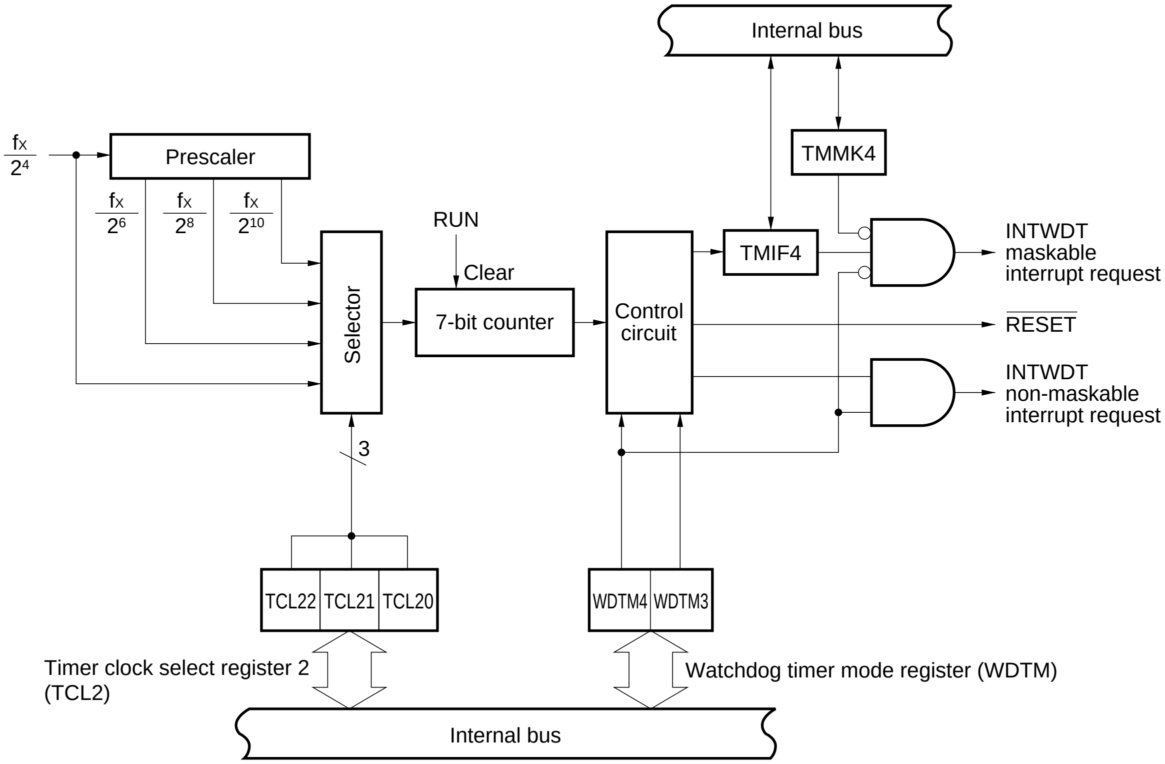
7.2 Configuration of Watchdog Timer

The watchdog timer consists of the following hardware:

Table 7-3. Configuration of Watchdog Timer

Item	Configuration
Control register	Timer clock select register 2 (TCL2) Watchdog timer mode register (WDTM)

Figure 7-1. Block Diagram of Watchdog Timer



7.3 Watchdog Timer Control Registers

The following two types of registers are used to control the watchdog timer.

- Timer clock select register 2 (TCL2)
- Watchdog timer mode register (WDTM)

(1) Timer clock select register 2 (TCL2)

This register sets the watchdog timer count clock.

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

$\overline{\text{RESET}}$ input sets TCL2 to 00H.

Figure 7-2. Timer Clock Select Register 2 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCL2	0	0	0	0	0	TCL22	TCL21	TCL20	FF42H	00H	R/W

TCL22	TCL21	TCL20	Watchdog Timer Count Clock Selection	Interval Time
0	0	0	$f_X/2^4$ (312.5 kHz)	$2^{11}/f_X$ (410 μs)
0	1	0	$f_X/2^6$ (78.1 kHz)	$2^{13}/f_X$ (1.64 ms)
1	0	0	$f_X/2^8$ (19.5 kHz)	$2^{15}/f_X$ (6.55 ms)
1	1	0	$f_X/2^{10}$ (4.88 kHz)	$2^{17}/f_X$ (26.2 ms)
Other than above			Setting prohibited	

Remarks 1. f_X: system clock oscillation frequency

2. Figures in parentheses apply to operation when f_X = 5.0 MHz.

(2) Watchdog timer mode register (WDTM)

This register sets an operation mode of the watchdog timer, and enables/disables counting of the watchdog timer.

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

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

Figure 7-3. Watchdog Timer Mode Register Format

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	FFF9H	00H	R/W

RUN	Selects Operation of Watchdog Timer[Note 1]
0	Stops counting.
1	Clears counter and starts counting.

WDTM4	WDTM3	Selects Operation Mode of Watchdog Timer[Note 2]
0	0	Operation stop
0	1	Interval timer mode (overflow occurs and maskable interrupt occurs)[Note 3]
1	0	Watchdog timer mode 1 (overflow occurs and non-maskable interrupt occurs)
1	1	Watchdog timer mode 2 (overflow occurs and reset operation started)

Notes **1.** Once RUN has been set to (1), it cannot be cleared to (0) by software. Therefore, when counting is started, it cannot be stopped by any means other than $\overline{\text{RESET}}$ input.

2. Once WDTM3 and WDTM4 have been set to (1), they cannot be cleared to (0) by software.

3. The watchdog timer starts operations as an interval timer when RUN is set to 1.

Cautions 1. When the watchdog timer is cleared by setting 1 to RUN, the actual overflow time is up to 0.8% shorter than the time set by the timer clock select register 2 (TCL2).

2. In watchdog timer mode 1 or 2, set WDTM4 to 1 after confirming the TMIF4 (bit 0 of interrupt request flag 0 (IF0)) being set to 0. When watchdog timer mode 1 or 2 is selected under the condition where TMIF4 is 1, a non-maskable interrupt occurs at the completion of rewriting.

7.4 Operation of Watchdog Timer

7.4.1 Operation as watchdog timer

The watchdog timer detects a program runaway when bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1.

The count clock (runaway detection time interval) of the watchdog timer can be selected by bits 0 to 2 (TCL20 to TCL22) of the timer clock select register 2 (TCL2). By setting bit 7 (RUN) of WDTM to 1, the watchdog timer is started. Set RUN to 1 within the set runaway detection time interval after the watchdog timer has been started. By setting RUN to 1, the watchdog timer can be cleared and start counting. If RUN is not set to 1, and the runaway detection time is exceeded, the system is reset or a non-maskable interrupt is generated by the value of bit 3 (WDTM3) of WDTM.

The watchdog timer continues operation in the HALT mode, but stops in the STOP mode. Therefore, set RUN to 1 before entering the STOP mode to clear the watchdog timer, and then execute the STOP instruction.

Caution The actual runaway detection time may be up to 0.8% shorter than the set time.

Table 7-4. Runaway Detection Time of Watchdog Timer

TCL22	TCL21	TCL20	Runaway Detection Time	At f_X = 5.0 MHz
0	0	0	$2^{11} \times 1/f_X$	410 μs
0	1	0	$2^{13} \times 1/f_X$	1.64 ms
1	0	0	$2^{15} \times 1/f_X$	6.55 ms
1	1	0	$2^{17} \times 1/f_X$	26.2 ms

f_X: system clock oscillation frequency

7.4.2 Operation as interval timer

When bits 4 and 3 (WDTM4 and WDTM3) of watchdog timer mode register (WDTM) are set to 0 and 1, the watchdog timer also operates as an interval timer that repeatedly generates an interrupt at time intervals specified by a count value set in advance.

Select a count clock (or interval time) by setting bits 0 through 2 (TCL20 to TCL22) of timer clock select register 2 (TCL2). The watchdog timer starts operation as an interval timer when the RUN bit (bit 7 of WDTM) is set to 1.

In the interval timer mode, the interrupt mask flag (TMMK4) is valid, and a maskable interrupt (INTWDT) can be generated. The priority of INTWDT is set as the highest of all the maskable interrupts.

The interval timer continues operation in the HALT mode, but stops in the STOP mode. Therefore, set RUN to 1 before entering the STOP mode to clear the interval timer, and then execute the STOP instruction.

Cautions 1. Once bit 4 (WDTM4) of WDTM is set to 1 (when the watchdog timer mode is selected), the interval timer mode is not set, unless the $\overline{\text{RESET}}$ signal is input.

2. The interval time immediately after the setting by WDTM may be up to 0.8% shorter than the set time.

Table 7-5. Interval Time of Interval Timer

TCL22	TCL21	TCL20	Interval Time	At f_X = 5.0 MHz
0	0	0	$2^{11} \times 1/f_X$	410 μs
0	1	0	$2^{13} \times 1/f_X$	1.64 ms
1	0	0	$2^{15} \times 1/f_X$	6.55 ms
1	1	0	$2^{17} \times 1/f_X$	26.2 ms

f_X: system clock oscillation frequency

CHAPTER 8 SERIAL INTERFACE 00

8.1 Serial Interface 00 Functions

The serial interface 00 employs the following three modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode
- 3-wire serial I/O mode

(1) Operation stop mode

This mode is used when serial transfer is not carried out. It enables power consumption reduction.

(2) Asynchronous serial interface (UART) mode

In this mode, one byte of data following the start bit is transmitted/received, and full-duplex operation is possible.

A dedicated UART baud rate generator is incorporated, allowing communication over a wide range of baud rates. In addition, the baud rate can be defined by scaling the input clock to the ASCK pin.

(3) 3-wire serial I/O mode (MSB/LSB start bit switchable)

In this mode, 8-bit data transfer is carried out with three lines, one for serial clock ($\overline{\text{SCK0}}$) and two for serial data (SI0, SO0).

The 3-wire serial I/O mode supports simultaneous transmit and receive operation, reducing data transfer processing time.

It is possible to switch the start bit of 8-bit data to be transmitted between the MSB and the LSB, thus allowing connection to devices with either start bit.

The 3-wire serial I/O mode is effective for connecting display controllers and peripheral I/Os such as the 75XL Series, 78K Series, and 17K Series, which have internal conventional clock synchronous serial interface.

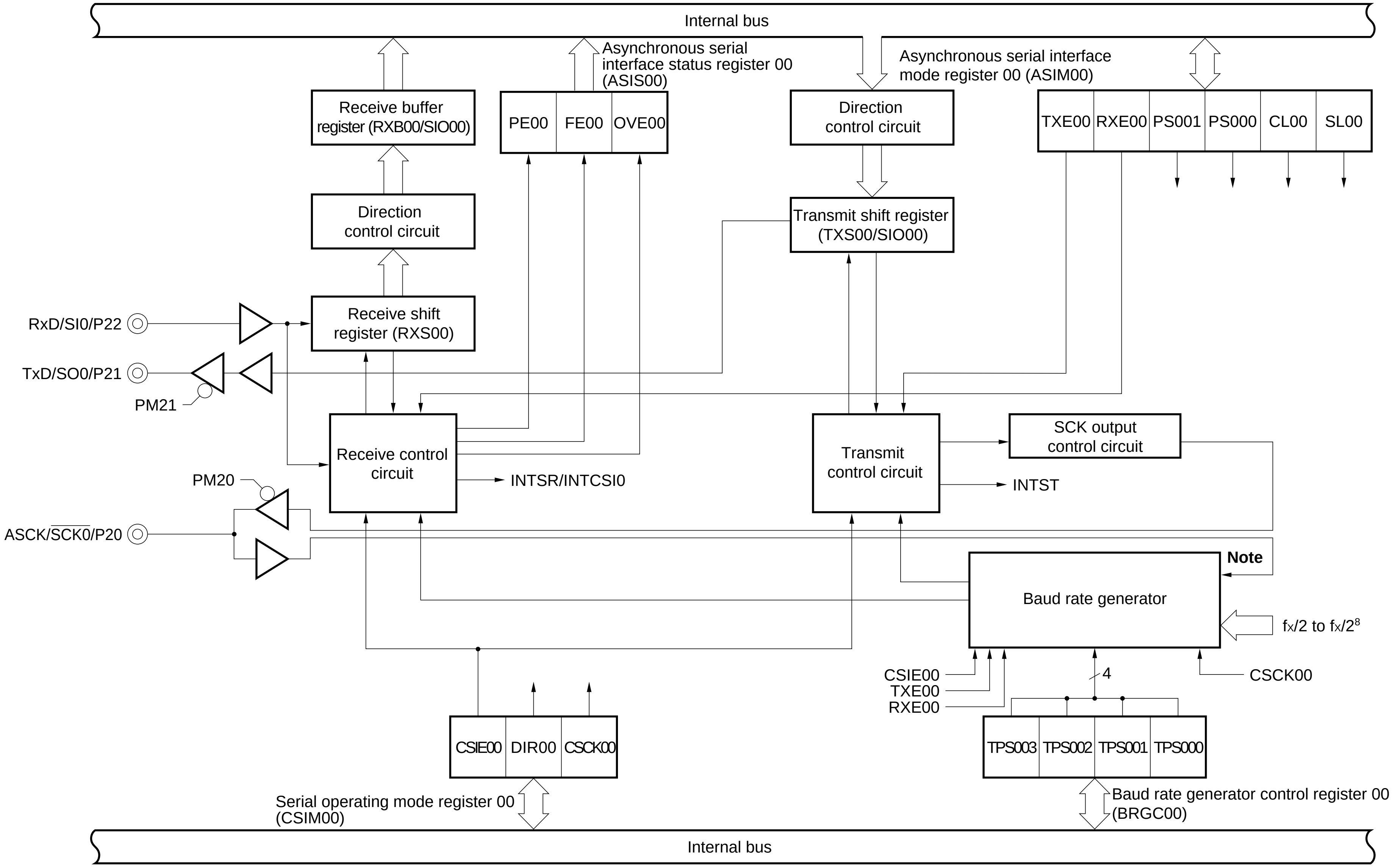
8.2 Serial Interface 00 Configuration

Serial interface 00 has the following hardware configuration.

Table 8-1. Serial Interface 00 Configuration

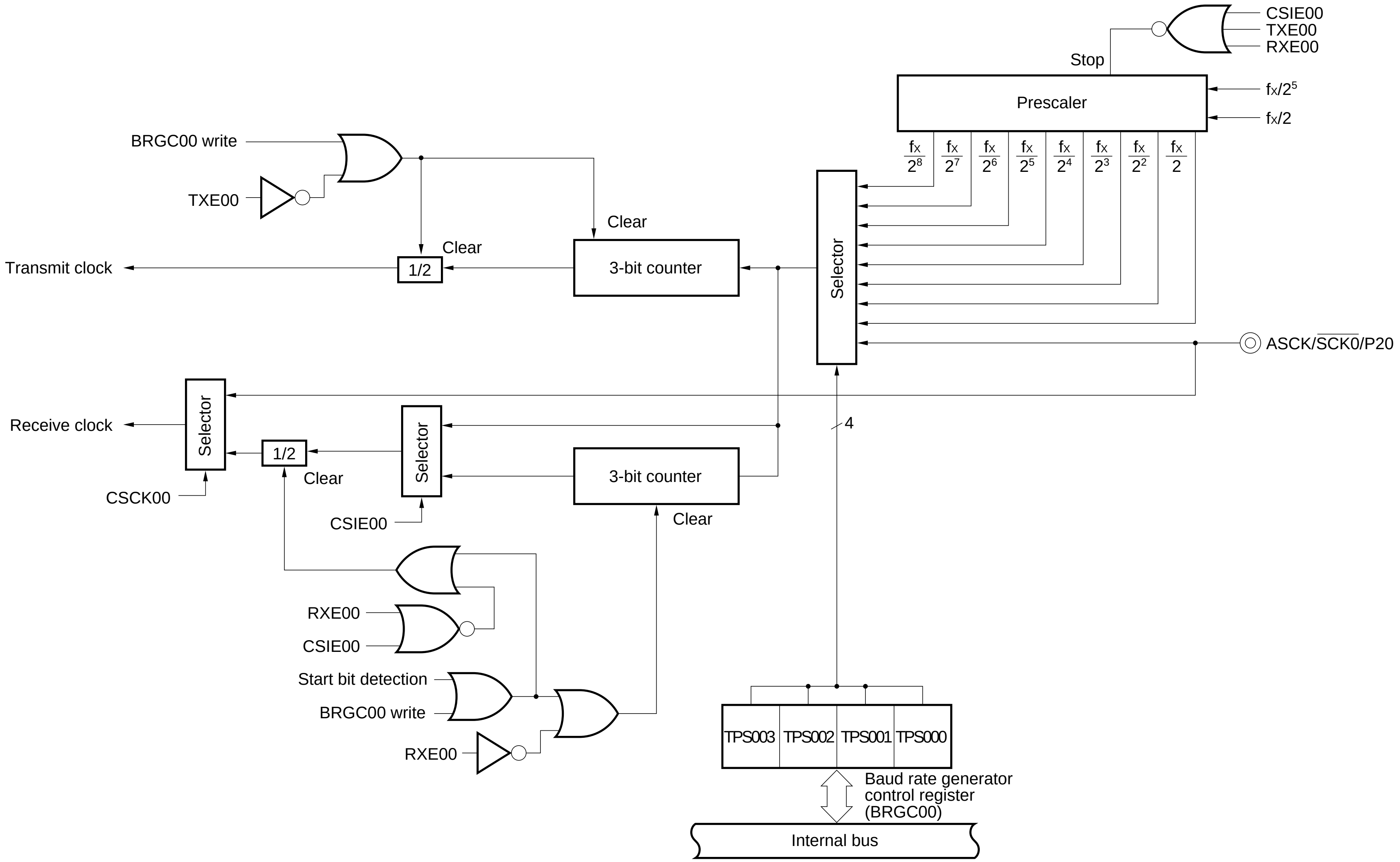
Item	Configuration
Register	Transmit shift register 00 (TXS00) Receive shift register 00 (RXS00) Receive buffer register 00 (RXB00)
Control register	Serial operating mode register 00 (CSIM00) Asynchronous serial interface mode register 00 (ASIM00) Asynchronous serial interface status register 00 (ASIS00) Baud rate generator control register 00 (BRGC00)

Figure 8-1. Block Diagram of Serial Interface 00


Internal bus
Asynchronous serial interface status register 00 (ASIS00)
Asynchronous serial interface mode register 00 (ASIM00)
Receive buffer register (RXB00/SIO00)
PE00
FE00
OVE00
Direction control circuit
TXE00
RXE00
PS001
PS000
CL00
SL00
Direction control circuit
Transmit shift register (TXS00/SIO00)
RxD/SI0/P22
Receive shift register (RXS00)
TxD/SO0/P21
PM21
Receive control circuit
INTSR/INTCSI0
Transmit control circuit
SCK output control circuit
INTST
PM20
ASCK/SCK0/P20
Note
Baud rate generator
fx/2 to fx/2^8
CSIE00
TXE00
RXE00
4
CSCK00
CSIE00
DIR00
CSCK00
TPS003
TPS002
TPS001
TPS000
Serial operating mode register 00 (CSIM00)
Baud rate generator control register 00 (BRGC00)
Internal bus

Note For the baud rate generator configuration, see Figure 8-2.

Figure 8-2. Block Diagram of Baud Rate Generator



(1) Transmit shift register 00 (TXS00)

This register is used to specify data to be transmitted. Data written to TXS00 is transmitted as serial data. If the data length is specified as 7 bits, bits 0 to 6 of the data written to TXS00 are transferred as the transmit data. The transmit operation is started by writing data to TXS00.

TXS00 is written to with an 8-bit memory manipulation instruction. It cannot be read.

$\overline{\text{RESET}}$ input sets TXS00 to FFH.

Caution Do not write to TXS00 during transmission.
TXS00 and the receive buffer register 00 (RXB00) are allocated to the same address, and when reading is performed, RXB00 values are read.

(2) Receive shift register 00 (RXS00)

This register is used to convert serial data input to the RxD pin into parallel data. Each time one byte of data is received, it is transferred to the receive buffer register 00 (RXB00).

The RXS00 cannot be manipulated directly by program.

(3) Receive buffer register 00 (RXB00)

This register is used to hold received data. Each time one byte of data is received, a new receive data is transferred from the receive shift register 00 (RXS00).

If the data length is specified as 7 bits, receive data is transferred to bits 0 to 6 of RXB00, and the MSB of RXB00 always becomes 0.

RXB00 can be read with an 8-bit memory manipulation instruction. It cannot be written to.

$\overline{\text{RESET}}$ input becomes undefined.

Caution RXB00 and the transmit shift register 00 (TXS00) are allocated to the same address, and when an interrupt is executed, the values are written to TXS00.

(4) Transmit control circuit

This circuit controls transmit operations by adding a start bit, parity bit, and stop bit to data written to the transmit shift register 00 (TXS00), according to the data set to the asynchronous serial interface mode register 00 (ASIM00).

(5) Receive control circuit

This circuit controls receive operations according to the data set to the asynchronous serial interface mode register 00 (ASIM00). It performs also parity error check, etc., during receive operations, and when an error is detected, it sets the value to the asynchronous serial interface status register 00 (ASIS00) depending on the nature of the error.

8.3 Serial Interface 00 Control Registers

The following four types of registers are used to control the serial interface 00.

- Serial operating mode register 00 (CSIM00)
- Asynchronous serial interface mode register 00 (ASIM00)
- Asynchronous serial interface status register 00 (ASIS00)
- Baud rate generator control register 00 (BRGC00)

(1) Serial operating mode register 00 (CSIM00)

This register is set when using the serial interface 00 in the 3-wire serial I/O mode.
CSIM00 is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CSIM00 to 00H.

★ **Figure 8-3. Serial Operating Mode Register 00 Format**

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM00	CSIE00	0	0	0	0	DIR00	CSCK00	0	FF72H	00H	R/W

CSIE00	Operation Control in 3-Wire Serial I/O Mode
0	Operation stop
1	Operation enable

DIR00	Start Bit Specification
0	MSB
1	LSB

CSCK00	Clock Selection in 3-Wire Serial I/O Mode
0	Input clock to $\overline{\text{SCK0}}$ pin from external
1	Dedicated baud rate generator output

Cautions 1. Be sure to set 0 to bit 0 and bits 3 to 6.

2. Set 00H to the CSIM00 at the UART mode.

★ **3. In the 3-wire serial I/O mode, if the operation is interrupted (CSIE00 = 0) during data transmission/reception, and if the operation control flag is cleared (CSIE00 = 0) when not performing data transmit/receive, P21, which is the alternate-function I/O port pin of SO0, cannot be used as a general-purpose output port.**

(2) Asynchronous serial interface mode register 00 (ASIM00)

This register is set when using the serial interface 00 in the asynchronous serial interface mode.
ASIM00 is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets ASIM00 to 00H.

Figure 8-4. Asynchronous Serial Interface Mode Register 00 Format

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After Reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	0	0	FF70H	00H	R/W

TXE00	Transmit Operation Control
0	Transmit operation stop
1	Transmit operation enable

RXE00	Receive Operation Control
0	Receive operation stop
1	Receive operation enable

PS001	PS000	Parity Bit Specification
0	0	No parity
0	1	Always add 0 parity at transmission Parity check is not performed at reception (No parity error is generated)
1	0	Odd parity
1	1	Even parity

CL00	Character Length Specification
0	7 bits
1	8 bits

SL00	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

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

2. Set 00H to the ASIM00 at the 3-wire serial I/O mode.

3. Switching operation modes must be performed after the halt of serial transmit/receive operation.

★ **4. In the UART mode, if the operation is interrupted (TXE00 = 0) during data transmission, P21, which is the alternate-function I/O port pin of TxD, cannot be used as a general-purpose output port.**

Table 8-2. Serial Interface 00 Operating Mode Settings (1/2)

(1) Operation stop mode

ASIM00		CSIM00			PM22	P22	PM21	P21	PM20	P20	Start	Shift	P22/SI0/RxD	P21/SO0/TxD	P20/$\overline{SCK0}$/ASCK
TXE00	RXE00	CSIE00	DIR00	CSCK00							Bit	Clock	Pin Function	Pin Function	Pin Function
0	0	0	×	×	×[Note 1]	×[Note 1]	×[Note 1]	×[Note 1]	×[Note 1]	×[Note 1]	—	—	P22	P21	P20
Other than above											Setting prohibited				

(2) Asynchronous serial interface mode

ASIM00		CSIM00			PM22	P22	PM21	P21	PM20	P20	Start	Shift	P22/SI0/RxD	P21/SO0/TxD	P20/$\overline{SCK0}$/ASCK
TXE00	RXE00	CSIE00	DIR00	CSCK00							Bit	Clock	Pin Function	Pin Function	Pin Function
1	0	0	0	0	×[Note 1]	×[Note 1]	0	1	1	×	LSB	External clock	P22	TxD (CMOS output)	ASCK input
									×[Note 1]	×[Note 1]		Internal clock			P20
0	1	0	0	0	1	×	×[Note 1]	×[Note 1]	1	×		External clock	RxD	P21	ASCK input
									×[Note 1]	×[Note 1]		Internal clock			P20
1	1	0	0	0	1	×	0	1	1	×		External clock		TxD (CMOS output)	ASCK input
									×[Note 1]	×[Note 1]		Internal clock			P20
Other than above											Setting prohibited				

Table 8-2. Serial Interface 00 Operating Mode Settings (2/2)

(3) 3-wire serial I/O mode

ASIM00		CSIM00			PM22	P22	PM21	P21	PM20	P20	Start	Shift	P22/SI0/RxD	P21/SO0/TxD	P20/$\overline{\text{SCK0}}$/ASCK
TXE00	RXE00	CSIE00	DIR00	CSCK00							Bit	Clock	Pin Function	Pin Function	Pin Function
0	0	1	0	0	1[Note 2]	×[Note 2]	0	1	1	×	MSB	External clock	SI0[Note 2]	SO0 (CMOS output)	$\overline{\text{SCK0}}$ input
				1					0	1		Internal clock			$\overline{\text{SCK0}}$ output
		1	1	0					1	×	LSB	External clock			$\overline{\text{SCK0}}$ input
				1					0	1		Internal clock			$\overline{\text{SCK0}}$ output
Other than above											Setting prohibited				

Notes **1.** Can be used as port function.

2. If used only for transmission, can be used as P22 (CMOS input/output).

Remark ×: don’t care

(3) Asynchronous serial interface status register 00 (ASIS00)

This register indicates types of error when a reception error is generated in the asynchronous interface mode.

★ ASIS00 is read with a 1-bit or 8-bit memory manipulation instruction.

The contents of ASIS00 become undefined in the 3-wire serial I/O mode.

$\overline{\text{RESET}}$ input sets ASIS00 to 00H.

★ **Figure 8-5. Asynchronous Serial Interface Status Register 00 Format**

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
ASIS00	0	0	0	0	0	PE00	FE00	OVE00	FF71H	00H	R

PE00	Parity Error Flag
0	Parity error not generated
1	Parity error generated (when the parity of transmit data does not coincide.)

FE00	Flaming Error Flag
0	Flaming error not generated
1	Flaming error generated (when stop bit is not detected.)[Note 1]

OVE00	Overrun Error Flag
0	Overrun error not generated
1	Overrun error generated[Note 2] (when the next receive operation is completed before the data is read from the receive buffer register.)

Notes

1. Even when the stop bit length is set to 2 bits by setting bit 2 (SL00) of the asynchronous serial interface mode register 00 (ASIM00), the stop bit detection in the case of reception is performed with 1 bit.
2. When an overrun error occurs, be sure to read out the receive buffer register 00 (RXB00). Unless RXB00 is read out, overrun errors occur at each data reception.

(4) Baud rate generator control register 00 (BRGC00)

This register is used to set the serial clock of serial interface 00.

BRGC00 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC00 to 00H.

★ **Figure 8-6. Baud Rate Generator Control Register 00 Format**

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
BRGC00	TPS003	TPS002	TPS001	TPS000	0	0	0	0	FF73H	00H	R/W

TPS003	TPS002	TPS001	TPS000	3-Bit Counter Source Clock Selection	n
0	0	0	0	$f_X/2$ (2.5 MHz)	1
0	0	0	1	$f_X/2^2$ (1.25 MHz)	2
0	0	1	0	$f_X/2^3$ (625 kHz)	3
0	0	1	1	$f_X/2^4$ (313 kHz)	4
0	1	0	0	$f_X/2^5$ (156 kHz)	5
0	1	0	1	$f_X/2^6$ (78.1 kHz)	6
0	1	1	0	$f_X/2^7$ (39.1 kHz)	7
0	1	1	1	$f_X/2^8$ (19.5 kHz)	8
1	0	0	0	Input clock from external to ASCK pin[Note]	–
Other than above				Setting prohibited	

Note Only used in UART mode.

Cautions 1. When writing to BRGC00 is performed during a communication operation, the baud rate generator output is disrupted and communications cannot be performed normally. Be sure not to write to BRGC00 during communication operation.

2. Be sure not to select n = 1 during an operation at f_X = 5.0 MHz because n = 1 exceeds the baud rate limit.

Remarks 1. f_X: system clock oscillation frequency

2. Figure in parentheses applies to operation when f_X = 5.0 MHz.

The baud rate transmit/receive clock to be generated is either a signal scaled from the system clock, or a signal scaled from the clock input from the ASCK pin.

(a) Generation of baud rate transmit/receive clock by means of system clock

The transmit/receive clock is generated by scaling the system clock. The baud rate generated from the system clock is found from the following expression.

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1} \times 8} \quad [\text{Hz}]$$

f_X : system clock oscillation frequency

Table 8-3. Example of Relationship between System Clock and Baud Rate

Baud Rate	BRGC00 Set Value	Error (%)	
(bps)		f_X = 5.0 MHz	f_X = 4.9152 MHz
1200	70H	1.73	0
2400	60H		
4800	50H		
9600	40H		
19200	30H		
38400	20H		
76800	10H		

(b) Generation of baud rate transmit/receive clock by means of external clock from ASCK pin

The transmit/receive clock is generated by scaling the clock input from the ASCK pin. The baud rate generated from the clock input from the ASCK pin is found from the following expression.

$$[\text{Baud rate}] = \frac{f_{\text{ASCK}}}{16}\ [\text{Hz}]$$

f_{ASCK}: frequency of clock input to ASCK pin

Table 8-4. Relationship between ASCK Pin Input Frequency and Baud Rate (When BRGC00 is set to 80H)

Baud Rate (bps)	ASCK Pin Input Frequency (kHz)
75	1.2
150	2.4
300	4.8
600	9.6
1200	19.2
2400	38.4
4800	76.8
9600	153.6
19200	307.2
31250	500.0
38400	614.4

8.4 Serial Interface 00 Operation

Serial interface 00 provides the following three types of modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode
- 3-wire serial I/O mode

8.4.1 Operation stop mode

In the operation stop mode, serial transfer is not executed, therefore, the power consumption can be reduced. The P20/$\overline{\text{SCK0}}$/ASCK, P21/SO0/TxD, and P22/SI0/RxD pins can be used as normal I/O ports.

(1) Register setting

Operation stop mode is set by serial operating mode register 00 (CSIM00) and asynchronous serial interface mode register 00 (ASIM00).

(a) Serial operating mode register 00 (CSIM00)

CSIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CSIM00 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM00	CSIE00	0	0	0	0	DIR00	CSCK00	0	FF72H	00H	R/W

CSIE00	Operation Control in 3-Wire Serial I/O Mode
0	Operation stop
1	Operation enable

Caution Be sure to set 0 to bit 0 and bits 3 to 6.

(b) Asynchronous serial interface mode register 00 (ASIM00)

ASIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIM00 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After Reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	0	0	FF70H	00H	R/W

TXE00	Transmit Operation Control
0	Transmit operation stop
1	Transmit operation enable

RXE00	Receive Operation Control
0	Receive operation stop
1	Receive operation enable

Caution Be sure to set 0 to bits 0 and 1.

8.4.2 Asynchronous serial interface (UART) mode

In this mode, the one-byte data following the start bit is transmitted/received and thus full-duplex communication is possible.

This device incorporates a UART-dedicated baud rate generator that enables communications at a desired transfer rate from many options. In addition, the baud rate can also be defined by dividing the input clock to the ASCK pin.

The UART-dedicated baud rate generator also can output the 31.25-kbps baud rate that complies with the MIDI standard.

(1) Register setting

The UART mode is set by serial operating mode register 00 (CSIM00), asynchronous serial interface mode register 00 (ASIM00), asynchronous serial interface status register 00 (ASIS00), and baud rate generator control register 00 (BRGC00).

(a) Serial operating mode register 00 (CSIM00)

CSIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CSIM00 to 00H.

Set 00H to CSIM00 when UART mode is selected.

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM00	CSIE00	0	0	0	0	DIR00	CSCK00	0	FF72H	00H	R/W

CSIE00	Operation Control in 3-Wire Serial I/O Mode
0	Operation stop
1	Operation enable

DIR00	Start Bit Specification
0	MSB
1	LSB

CSCK00	Clock Selection in 3-Wire Serial I/O Mode
0	Input clock to $\overline{\text{SCK0}}$ pin from external
1	Dedicated baud rate generator output

Caution Be sure to set 0 to bit 0 and bits 3 to 6.

(b) Asynchronous serial interface mode register 00 (ASIM00)

ASIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIM00 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After Reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	0	0	FF70H	00H	R/W

TXE00	Transmit Operation Control
0	Transmit operation stop
1	Transmit operation enable

RXE00	Receive Operation Control
0	Receive operation stop
1	Receive operation enable

PS001	PS000	Parity Bit Specification
0	0	No parity
0	1	Always add 0 parity at transmission Parity check is not performed at reception (No parity error is generated)
1	0	Odd parity
1	1	Even parity

CL00	Character Length Specification
0	7 bits
1	8 bits

SL00	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

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

2. Switching operation modes must be performed after the halt of serial transmit/ receive operation.

(c) Asynchronous serial interface status register 00 (ASIS00)

★ ASIS00 is read with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIS00 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
ASIS00	0	0	0	0	0	PE00	FE00	OVE00	FF71H	00H	R

PE00	Parity Error Flag
0	Parity error not generated
1	Parity error generated (when the parity of transmit data does not coincide.)

FE00	Flaming Error Flag
0	Framing error not generated
1	Framing error generated (when stop bit is not detected.)Note 1

OVE00	Overrun Error Flag
0	Overrun error not generated
1	Overrun error generatedNote 2 (when the next receive operation is completed before the data is read from the receive buffer register.)

Notes **1.** Even when the stop bit length is set to 2 bits by setting bit 2 (SL00) of the asynchronous serial interface mode register 00 (ASIM00), the stop bit detection in the case of reception is performed with 1 bit.

2. When an overrun error occurs, be sure to read out the receive buffer register 00 (RXB00). Unless RXB00 is read out, overrun errors occur at each data reception.

(d) Baud rate generator control register 00 (BRGC00)

BRGC00 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC00 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
BRGC00	TPS003	TPS002	TPS001	TPS000	0	0	0	0	FF73H	00H	R/W

TPS003	TPS002	TPS001	TPS000	3-Bit Counter Source Clock Selection	n
0	0	0	0	$f_X/2$ (2.5 MHz)	1
0	0	0	1	$f_X/2^2$ (1.25 MHz)	2
0	0	1	0	$f_X/2^3$ (625 kHz)	3
0	0	1	1	$f_X/2^4$ (313 kHz)	4
0	1	0	0	$f_X/2^5$ (156 kHz)	5
0	1	0	1	$f_X/2^6$ (78.1 kHz)	6
0	1	1	0	$f_X/2^7$ (39.1 kHz)	7
0	1	1	1	$f_X/2^8$ (19.5 kHz)	8
1	0	0	0	Input clock from external to ASCK pin	–
Other than above				Setting prohibited	

Cautions 1. When writing to BRGC00 is performed during a communication operation, the output of baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC00 during communication operation.

2. Be sure not to select n = 1 during an operation at f_X = 5.0 MHz because n = 1 exceeds the baud rate limit.

Remarks 1. f_X: system clock oscillation frequency

2. Figure in parentheses applies to operation when f_X = 5.0 MHz.

The baud rate transmit/receive clock to be generated is either a signal scaled from the system clock, or a signal scaled from the clock input from the ASCK pin.

(i) Generation of baud rate transmit/receive clock by means of system clock

The transmit/receive clock is generated by scaling the system clock. The baud rate generated from the system clock is found from the following expression.

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1} \times 8} \ [\text{Hz}]$$

f_X: system clock oscillation frequency

n: value set in TPS000 to TPS003 ($2 \leq n \leq 8$)

Table 8-5. Example of Relationship between System Clock and Baud Rate

Baud Rate	BRGC00 Set Value	Error (%)	
(bps)		f_X = 5.0 MHz	f_X = 4.9152 MHz
1200	70H	1.73	0
2400	60H		
4800	50H		
9600	40H		
19200	30H		
38400	20H		
76800	10H		

(ii) Generation of baud rate transmit/receive clock by means of external clock from ASCK pin

The transmit/receive clock is generated by scaling the clock input from the ASCK pin. The baud rate generated from the clock input from the ASCK pin is found from the following expression.

$$[\text{Baud rate}] = \frac{f_{ASCK}}{16} \ [\text{Hz}]$$

f_{ASCK}: frequency of clock input to ASCK pin

Table 8-6. Relationship between ASCK Pin Input Frequency and Baud Rate (When BRGC00 is set to 80H)

Baud Rate (bps)	ASCK Pin Input Frequency (kHz)
75	1.2
150	2.4
300	4.8
600	9.6
1200	19.2
2400	38.4
4800	76.8
9600	153.6
19200	307.2
31250	500.0
38400	614.4

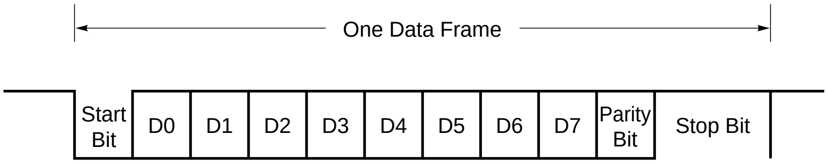
(2) Communication operation

(a) Data format

The transmit/receive data format is as shown in Figure 8-7. One data frame consists of a start bit, character bits, parity bit and stop bit(s).

The specification of character bit length, parity selection, and specification of stop bit length for each data frame is carried out with asynchronous serial interface mode register 00 (ASIM00).

Figure 8-7. Asynchronous Serial Interface Transmit/Receive Data Format



• Start bit	1 bit
• Character bits	7 bits/8 bits
• Parity bits	Even parity/odd parity/0 parity/no parity
• Stop bit(s)	1 bit/2 bits

When 7 bits are selected as the number of character bits, only the lower 7 bits (bits 0 to 6) are valid; in transmission the most significant bit (bit 7) is ignored, and in reception the most significant bit (bit 7) is always “0”.

The serial transfer rate is selected by means of ASIM00 and the baud rate generator control register 00 (BRGC00).

If a serial data receive error is generated, the receive error contents can be determined by reading the status of the asynchronous serial interface status register 00 (ASIS00).

(b) Parity types and operation

The parity bit is used to detect a bit error in the communication data. Normally, the same kind of parity bit is used on the transmitting side and the receiving side. With even parity and odd parity, a one-bit (odd number) error can be detected. With 0 parity and no parity, an error cannot be detected.

(i) Even parity

- **At transmission**

  The transmission operation is controlled so that the number of bits with a value of “1” in the transmit data including parity bit may be even. The parity bit value should be as follows.

  The number of bits with a value of “1” is an odd number in transmit data : 1
  The number of bits with a value of “1” is an even number in transmit data : 0

- **At reception**

  The number of bits with a value of “1” in the receive data including parity bit is counted, and if the number is odd, a parity error is generated.

(ii) Odd parity

- **At transmission**

  Conversely to the even parity, the transmission operation is controlled so that the number of bits with a value of “1” in the transmit data including parity bit may be odd. The parity bit value should be as follows.

  The number of bits with a value of “1” is an odd number in transmit data : 0
  The number of bits with a value of “1” is an even number in transmit data : 1

- **At reception**

  The number of bits with a value of “1” in the receive data including parity bit is counted, and if the number is even, a parity error is generated.

(iii) 0 Parity

When transmitting, the parity bit is set to “0” irrespective of the transmit data.
At reception, a parity bit check is not performed. Therefore, a parity error is not generated, irrespective of whether the parity bit is set to “0” or “1”.

(iv) No parity

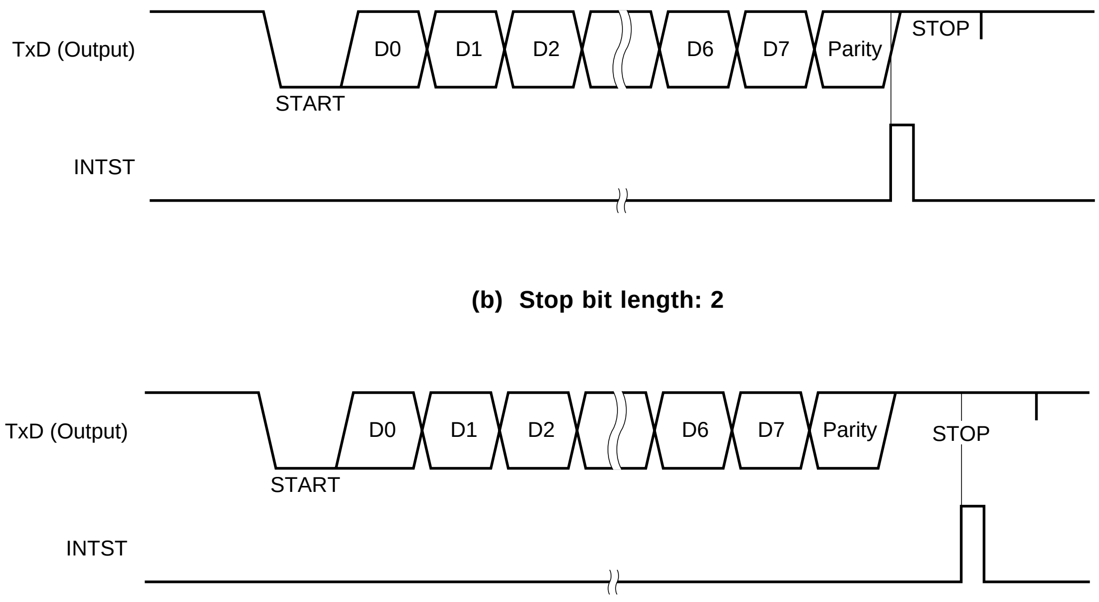
A parity bit is not added to the transmit data. At reception, data is received assuming that there is no parity bit. Since there is no parity bit, a parity error is not generated.

(c) Transmission

A transmit operation is started by writing transmit data to the transmit shift register 00 (TXS00). The start bit, parity bit and stop bit(s) are added automatically.

When the transmit operation starts, the data in TXS00 is shifted out, and when TXS00 is empty, a transmission completion interrupt (INTST) is generated.

Figure 8-8. Asynchronous Serial Interface Transmission Completion Interrupt Timing



(a) Stop bit length: 1

(b) Stop bit length: 2

Caution Do not replace the asynchronous serial interface mode register 00 (ASIM00) during a transmit operation. If the ASIM00 register is replaced during transmission, subsequent transmission may not be performed (the normal state is restored by $\overline{\text{RESET}}$ input).

It is possible to determine whether transmission is in progress by software by using a transmission completion interrupt (INTST) or the interrupt request flag (STIF00) set by INTST.

(d) Reception

When bit 6 (RXE) of the asynchronous serial interface mode register 00 (ASIM00) is set (1), a receive operation is enabled and sampling of the RxD pin input is performed.

RxD pin input sampling is performed using the serial clock specified by ASIM00.

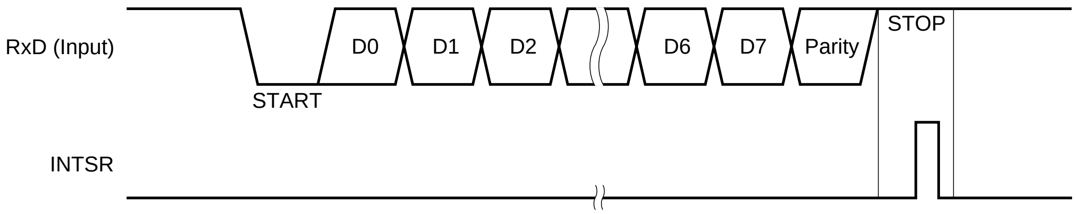
When the RxD pin input becomes low, the 3-bit counter starts counting, and at the time when the half time determined by the specified baud rate has passed, the data sampling start timing signal is output. If the RxD pin input sampled again as a result of this start timing signal is low, it is identified as a start bit, the 3-bit counter is initialized and starts counting, and data sampling is performed. When character data, a parity bit and one stop bit are detected after the start bit, reception of one frame of data ends.

When one frame of data has been received, the receive data in the shift register is transferred to the receive buffer register 00 (RXB00), and a reception completion interrupt (INTSR) is generated.

If an error is generated, the receive data in which the error was generated is still transferred to RXB00, and INTSR is generated.

If the RXE00 bit is reset (0) during the receive operation, the receive operation is stopped immediately. In this case, the contents of RXB00 and asynchronous serial interface status register 00 (ASIS00) are not changed, and INTSR is not generated.

Figure 8-9. Asynchronous Serial Interface Reception Completion Interrupt Timing



Caution Be sure to read the receive buffer register 00 (RXB00) even if a receive error occurs. If RXB00 is not read, an overrun error will be generated when the next data is received, and the receive error state will continue indefinitely.

(e) Receive errors

The following three errors may occur during a receive operation: a parity error, framing error, or overrun error. The data reception result error flag is set in the asynchronous serial interface status register 00 (ASIS00). Receive error causes are shown in Table 8-7.

It is possible to determine what kind of error was generated during reception by reading the contents of ASIS00 in the reception error interrupt servicing (see Figures 8-9 and 8-10).

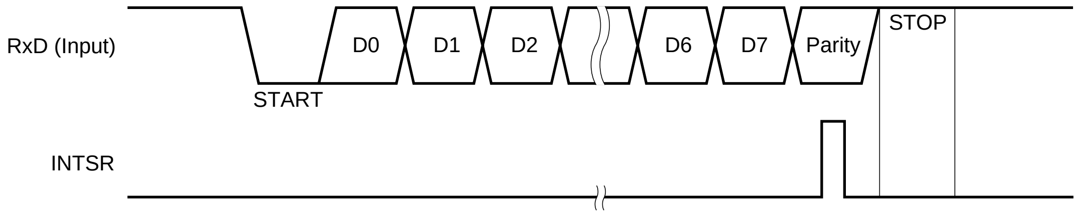
The contents of ASIS00 are reset (0) by reading the receive buffer register 00 (RXB00) or receiving the next data (if there is an error in the next data, the corresponding error flag is set).

Table 8-7. Receive Error Causes

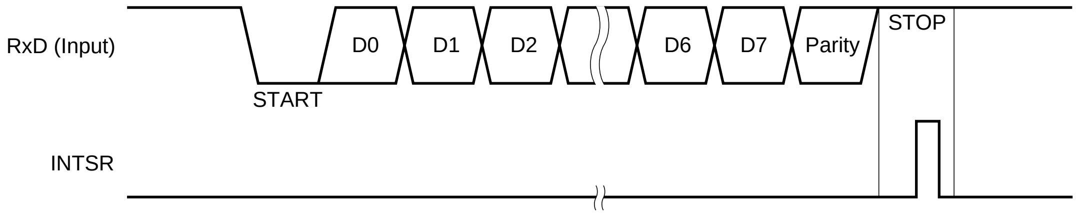
Receive Errors	Cause
Parity error	Transmission-time parity specification and reception data parity do not match
Framing error	Stop bit not detected
Overrun error	Reception of next data is completed before data is read from receive buffer register

Figure 8-10. Receive Error Timing

(a) Parity error generated



(b) Framing error or overrun error generated



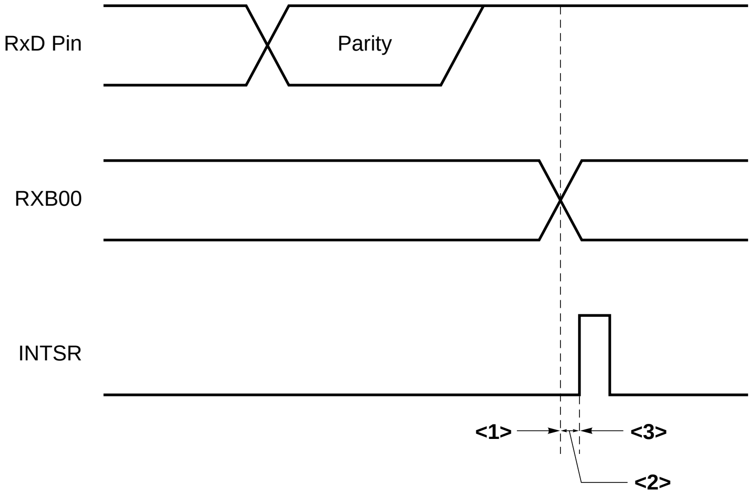
Cautions 1. The contents of the ASIS00 register are reset (0) by reading the receive buffer register 00 (RXB00) or receiving the next data. To ascertain the error contents, read ASIS00 before reading RXB00.

2. Be sure to read the receive buffer register 00 (RXB00) even if a receive error occurs. If RXB00 is not read, an overrun error will be generated when the next data is received, and the receive error state will continue indefinitely.

(3) UART mode cautions

(a) When bit 7 (TXE00) of the asynchronous serial interface mode register 00 (ASIM00) is cleared during transmission, be sure to set the transmit shift register 00 (TXS00) to FFH, then set TXE00 to 1 before executing the next transmission.

(b) When bit 6 (RXE00) of the asynchronous serial interface mode register 00 (ASIM00) is cleared during reception, receive buffer register 00 (RXB00) and receive completion interrupt (INTSR) are as follows.



When RXE00 is set to 0 at a time indicated by **<1>**, RXB00 holds the previous data and does not generate INTSR.
When RXE00 is set to 0 at a time indicated by **<2>**, RXB00 renews the data and does not generate INTSR.
When RXE00 is set to 0 at a time indicated by **<3>**, RXB00 renews the data and generates INTSR.

★ (c) If the operation is interrupted (TXE00 = 0) during data transmission, P21, which is the alternate-function I/O port pin of TxD, cannot be used as a general-purpose output port. Therefore, do not clear the transmit operation enable flag (TXE00 = 0) during data transmission in the transmit operation enabled state (TXE00 = 1). When switching to the general-purpose output port, clear the transmit operation enable flag (TXE00 = 0) upon completion of data transmission.

An example of a program switching to the general-purpose output port following UART transmit completion is shown below.

```
        MOV     CSIM00, #00H
        MOV     BRGC00, #40H    ; Baud rate           : 9600 bps
        MOV     ASIM00, #88H    ; Character length    : 8 bits
                                  Stop bit length     : 1bit, no parity
WAIT:
        BF      STIF00, $WAIT
        CLR1    TXE00
```

★ (d) Upon occurrence of INTSR, if the RXB00 register has been read before the clock shown in Table 8-8 from the start of this interrupt routine, an overrun error is generated. To read the receive data, do so after letting the number of clocks indicated in Table 8-8 elapse prior to reading RXB00.

Table 8-8. Number of Clocks Required to Read RXB00 Register

BRGC00 Set Value	Transfer Rate[Note]	System Clock High-Speed Operation (PCC1 = 0)	System Clock Low-Speed Operation (PCC1 = 1)
10H	76800	0	0
20H	38400	0	0
30H	19200	7	2
40H	9600	23	6
50H	4800	55	14
60H	2400	119	30
70H	1200	247	62
80H	In the case of external clock, satisfy the following equation. EXCL1 (Hz) > f_{CPU} (Hz)/(9 clocks + × clocks)		

Note When f_X = 4.9152-MHz operation

EXCL1 : External clock frequency (transfer rate × 2^4)

f_{CPU} : CPU operation frequency

9 clocks : Because interrupt processing is started one clock after the occurrence of an interrupt (interrupt processing uses 8 clocks), a total of 9 clocks are required.

× clocks : Number of clocks until RXB00 read

Example Number of clocks required to read RXB00 register if clock signal is input from external when 300-bps transfer is desired (when f_{CPU} = 1-MHz operation)

$$\begin{aligned} \text{EXCL1} &= 300 \times 2^4 \\ &= 4.8\text{ kHz} \end{aligned}$$

$$\begin{aligned} 4.8\text{ kHz} &> 1\text{ MHz}/(9 + \times) \\ \times &> (1\text{ MHz}/4.8\text{ kHz}) - 9 \\ \times &> 199.3 \end{aligned}$$

Therefore, in this case, wait 200 clocks before reading the RXB00 register within the interrupt routine.

8.4.3 3-wire serial I/O mode

The 3-wire serial I/O mode is useful for connection of peripheral I/Os and display controllers, etc., which incorporate a conventional synchronous clocked serial interface, such as the 75XL Series, 78K Series, 17K Series, etc.

Communication is performed using three lines: the serial clock ($\overline{\text{SCK0}}$), serial output (SO0), and serial input (SI0).

(1) Register setting

3-wire serial I/O mode settings are performed using serial operating mode register 00 (CSIM00), the asynchronous serial interface mode register 00 (ASIM00), and the baud rate generator control register 00 (BRGC00).

(a) Serial operating mode register 00 (CSIM00)

CSIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CSIM00 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM00	CSIE00	0	0	0	0	DIR00	CSCK00	0	FF72H	00H	R/W

CSIE00	Operation Control in 3-Wire Serial I/O Mode
0	Operation stop
1	Operation enable

DIR00	Start Bit Specification
0	MSB
1	LSB

CSCK00	Clock Selection in 3-Wire Serial I/O Mode
0	Input clock to $\overline{\text{SCK0}}$ pin from external
1	Dedicated baud rate generator output

Caution Be sure to set 0 to bit 0 and bits 3 to 6.

(b) Asynchronous serial interface mode register 00 (ASIM00)

ASIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIM00 to 00H.

When the 3-wire serial I/O mode is selected, 00H must be set to ASIM00.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After Reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	0	0	FF70H	00H	R/W

TXE00	Transmit Operation Control
0	Transmit operation stop
1	Transmit operation enable

RXE00	Receive Operation Control
0	Receive operation stop
1	Receive operation enable

PS001	PS000	Parity Bit Specification
0	0	No parity
0	1	Always add 0 parity at transmission Parity check is not performed at reception (No parity error is generated)
1	0	Odd parity
1	1	Even parity

CL00	Character Length Specification
0	7 bits
1	8 bits

SL00	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

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

2. Switching operation modes must be performed after serial transmit/receive operation is halted.

(c) Baud rate generator control register 00 (BRGC00)

BRGC00 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC00 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
BRGC00	TPS003	TPS002	TPS001	TPS000	0	0	0	0	FF73H	00H	R/W

TPS003	TPS002	TPS001	TPS000	3-Bit Counter Source Clock Selection	n
0	0	0	0	$f_X/2$ (2.5 MHz)	1
0	0	0	1	$f_X/2^2$ (1.25 MHz)	2
0	0	1	0	$f_X/2^3$ (625 kHz)	3
0	0	1	1	$f_X/2^4$ (313 kHz)	4
0	1	0	0	$f_X/2^5$ (156 kHz)	5
0	1	0	1	$f_X/2^6$ (78.1 kHz)	6
0	1	1	0	$f_X/2^7$ (39.1 kHz)	7
0	1	1	1	$f_X/2^8$ (19.5 kHz)	8
Other than above				Setting prohibited	

Cautions 1. When writing to BRGC00 is performed during a communication operation, the baud rate generator output is disrupted and communications cannot be performed normally. Be sure not to write to BRGC00 during communication operation.

2. Be sure not to select n = 1 during an operation at f_X = 5.0 MHz because n = 1 exceeds the baud rate limit.

Remarks 1. f_X: system clock oscillation frequency

2. Figure in parentheses applies to operation when f_X = 5.0 MHz.

If the internal clock is used as the serial clock for the 3-wire serial I/O mode, set the TPS000 to TPS003 bits to set the frequency of the serial clock. To obtain the frequency to be set, use the following formula. When the serial clock is input from off-chip, setting BRGC00 is not necessary.

$$\text{Serial clock frequency} = \frac{f_X}{2^{n+1}} \text{ [Hz]}$$

f_X : system clock oscillation frequency

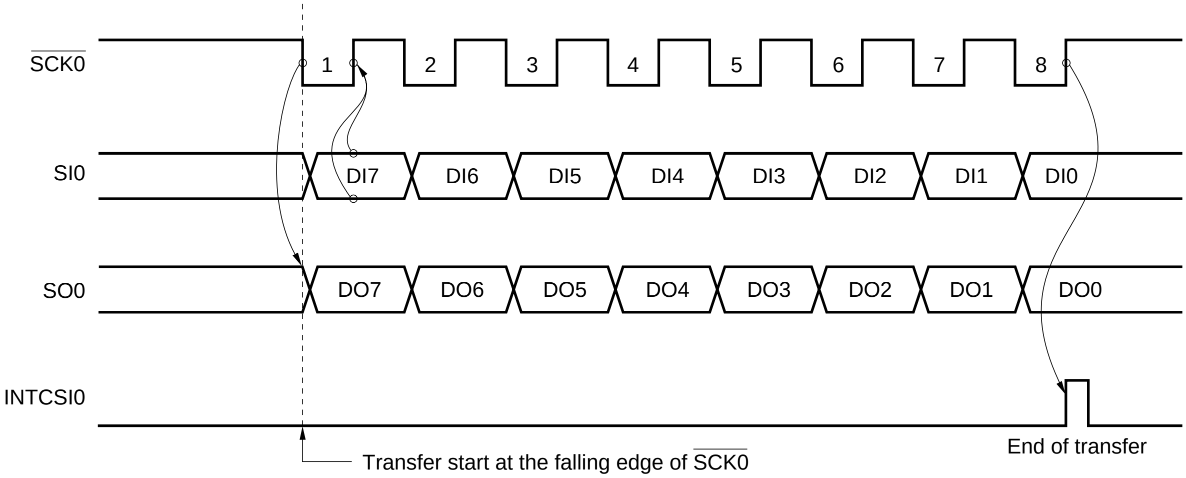
(2) **Communication operation**

In the 3-wire serial I/O mode, data transmission/reception is performed in 8-bit units. Data is transmitted/ received bit by bit in synchronization with the serial clock.

Transmit shift register 00 (TXS00/SIO00) and receive shift register 00 (RXS00) shift operations are performed in synchronization with the fall of the serial clock ($\overline{\text{SCK0}}$). Then transmit data is held in the SO0 latch and output from the SO0 pin. Also, receive data input to the SI0 pin is latched in the receive buffer register 00 (RXB00/SIO00) on the rise of $\overline{\text{SCK0}}$.

At the end of an 8-bit transfer, the operation of TXS00/SIO00 or RXS00 stops automatically, and the interrupt request signal (INTCSI0) is generated.

Figure 8-11. 3-Wire Serial I/O Mode Timing



★ **Caution In the 3-wire serial I/O mode, if the operation is interrupted (CSIM00 = 0) during data transmission/reception, and if the operation control flag is cleared (CSIE00 = 0) when not performing data transmit/receive, P21, which is the alternate-function I/O port pin of SO0, cannot be used as a general-purpose output port.**

To use P21 as a general-purpose output port, do the following.

- **Do not clear the operation control flag (CSIE00) until transmit/receive ends.**
- **To stop the 3-wire serial I/O mode, clear the operation control flag (CSIE00 = 0) after transmitting FFH once. Moreover, clear the transmit operation enable flag (TXE00 = 0) after transmitting FFH as a UART mode transmission.**

An example of a program for stopping the 3-wire serial I/O mode is shown below.

Example 1. Transmission in 3-wire serial I/O mode

```
MOV     CSIM00, #02H
MOV     BRGC00, #00H
MOV     ASIM00, #80H
MOV     TXS00, #0FFH
CLR1    CSIE00
```

When TXS00 is written to, the SO0 pin immediately becomes high level (after 4 clocks). However, a clock rides the $\overline{\text{SCK0}}$ clock signal.

Example 2. Transmission in UART mode

```
MOV     CSIM00, #00H
MOV     BRGC00, #00H
MOV     ASIM00, #80H
MOV     TXS00, #0FFH
CLR1    TXE00
```

After 16 to 32 clocks following writing to TXS00, the SO0 pin becomes high level. By using this method, the $\overline{\text{SCK0}}$ pin remains low level.

(3) Transfer start

Serial transfer is started by setting transfer data to the transmit shift register 00 (TXS00/SIO00) when the following two conditions are satisfied.

- Serial operating mode register 00 (CSIM00) bit 7 (CSIE00) = 1
- Internal serial clock is stopped or $\overline{\text{SCK0}}$ is a high level after 8-bit serial transfer.

Caution If CSIE00 is set to “1” after data write to TXS00/SIO00, transfer does not start.

A termination of 8-bit transfer stops the serial transfer automatically and generates the interrupt request signal (INTCSI0).

[MEMO]

CHAPTER 9 INTERRUPT FUNCTIONS

9.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged unconditionally even in the interrupt disabled state. It does no undergo interrupt priority control and is given top priority over all other interrupt requests.
A standby release signal is generated.
The non-maskable interrupt has one source of interrupt from the watchdog timer.

(2) Maskable interrupt

These interrupts undergo mask control. If two or more interrupts with the same priority are simultaneously generated, each interrupt has a predetermined priority (priority) as shown in Table 9-1.
A standby release signal is generated.
★ The maskable interrupt has three sources of external interrupts and five sources of internal interrupts.

9.2 Interrupt Sources and Configuration

There are total of 9 non-maskable and maskable interrupts in the interrupt sources (see **Table 9-1**).

Table 9-1. Interrupt Source List

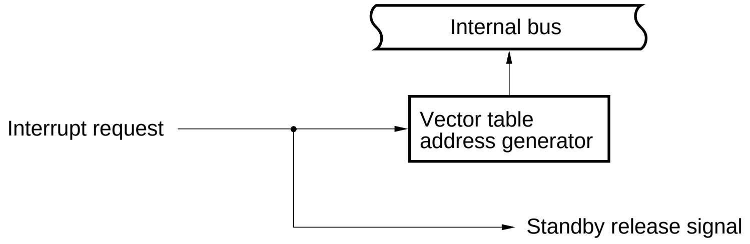
Interrupt Type	Priority[Note 1]	Interrupt Source		Internal /External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	—	INTWDT	Watchdog timer overflow (with watchdog timer mode 1 selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (with interval timer mode selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTSR	End of serial interface 00 UART reception	Internal	000CH	(B)
		INTCSI0	End of serial interface 00 3-wire transfer			
	5	INTST	End of serial interface 00 UART transmission		000EH	
	6	INTTM0	Generation of matching signal of 8-bit timer/event counter 00		0010H	
	7	INTTM1	Generation of matching signal of 8-bit timer/event counter 01		0012H	

Notes 1. Priority is the priority order when several maskable interrupts are generated at the same time. 0 is the highest order and 7 is the lowest order.

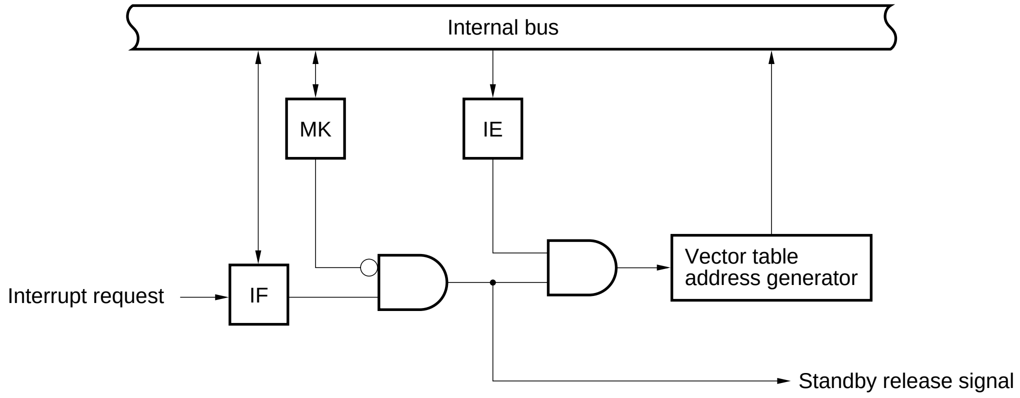
2. Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 9-1.

Figure 9-1. Basic Configuration of Interrupt Function

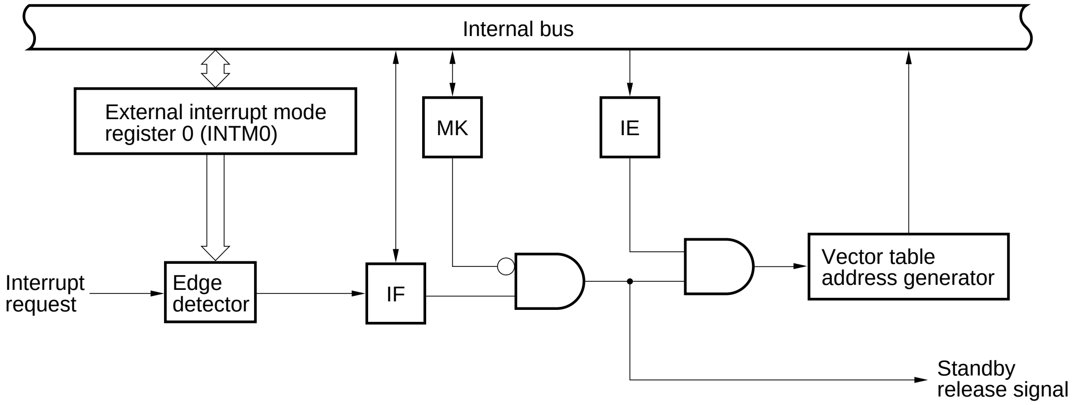
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt


Internal bus
External interrupt mode register 0 (INTM0)
MK
IE
Interrupt request
Edge detector
IF
Vector table address generator
Standby release signal

IF	:	interrupt request flag
IE	:	interrupt enable flag
MK	:	interrupt mask flag

9.3 Interrupt Function Control Registers

The following four registers are used to control the interrupt functions.

- Interrupt request flag register 0 (IF0)
- Interrupt mask flag register 0 (MK0)
- External interrupt mode register 0 (INTM0)
- Program status word (PSW)

Table 9-2 gives a listing of interrupt request flag and interrupt mask flag names corresponding to interrupt requests.

Table 9-2. Flags Corresponding to Interrupt Request Signal Name

Interrupt Request Signal Name	Interrupt Request Flag	Interrupt Mask Flag
INTWDT	TMIF4	TMMK4
INTP0	PIF0	PMK0
INTP1	PIF1	PMK1
INTP2	PIF2	PMK2
INTSR/INTCSI0	SRIF00	SRMK00
INTST	STIF00	STMK00
INTTM0	TMIF00	TMMK00
INTTM1	TMIF01	TMMK01

(1) Interrupt request flag register 0 (IF0)

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 acknowledgement of an interrupt request or upon $\overline{\text{RESET}}$ input.

IF0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets IF0 to 00H.

★ **Figure 9-2. Interrupt Request Flag Register 0 Format**

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After Reset	R/W
IF0	TMIF01	TMIF00	STIF00	SRIF00	PIF2	PIF1	PIF0	TMIF4	FFE0H	00H	R/W

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

Cautions 1. TMIF4 flag is R/W enabled only when a watchdog timer is used as an interval timer. If the watchdog timer mode 1 and 2 are used, set TMIF4 flag to 0.

2. Because port 3 has an alternate function as the external interrupt 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) Interrupt mask flag register 0 (MK0)

The interrupt mask flag is used to enable/disable the corresponding maskable interrupt service.

MK0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets MK0 to FFH.

★ **Figure 9-3. Interrupt Mask Flag Register 0 Format**

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After Reset	R/W
MK0	TMMK01	TMMK00	STMK00	SRMK00	PMK2	PMK1	PMK0	TMMK4	FFE4H	FFH	R/W

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

Cautions 1. If the TMMK4 flag is read when a watchdog timer is used in watchdog timer mode 1 and 2, its value becomes undefined.

2. Because port 3 has an alternate function as the external interrupt 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.

(3) External interrupt mode register 0 (INTM0)

This register is used to set the valid edge of INTP0 to INTP2.

INTM0 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets INTM0 to 00H.

Figure 9-4. External Interrupt Mode Register 0 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	0	FFECH	00H	R/W

ES21	ES20	INTP2 Valid Edge Selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES11	ES10	INTP1 Valid Edge Selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES01	ES00	INTP0 Valid Edge Selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

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

2. Before setting INTM0 register, be sure to set the corresponding interrupt mask flag (××MK× = 1) to disable interrupts. After setting INTM0 register, clear the interrupt request flag (××IF× = 0), clear the interrupt mask flag (××MK× = 0) to enable interrupts.

(4) Program status word (PSW)

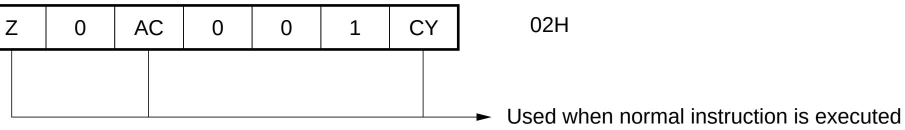
The program status word is a register used to hold the instruction execution result and the current status for interrupt requests. The IE flag to set maskable interrupt enable/disable is mapped.

Besides 8-bit unit read/write, this register can carry out operations with a bit manipulation instruction and dedicated instructions (EI, DI). When a vectored interrupt request is acknowledged, PSW is automatically saved into a stack, and the IE flag is reset to 0. It is reset from the stack with the RETI and POP PSW instructions.

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

Figure 9-5. Program Status Word Configuration

Symbol	7	6	5	4	3	2	1	0	After Reset
PSW	IE	Z	0	AC	0	0	1	CY	02H



IE	Interrupt Acknowledge Enable/Disable
0	Disable
1	Enable

9.4 Interrupt Processing Operation

9.4.1 Non-maskable interrupt request acceptance operation

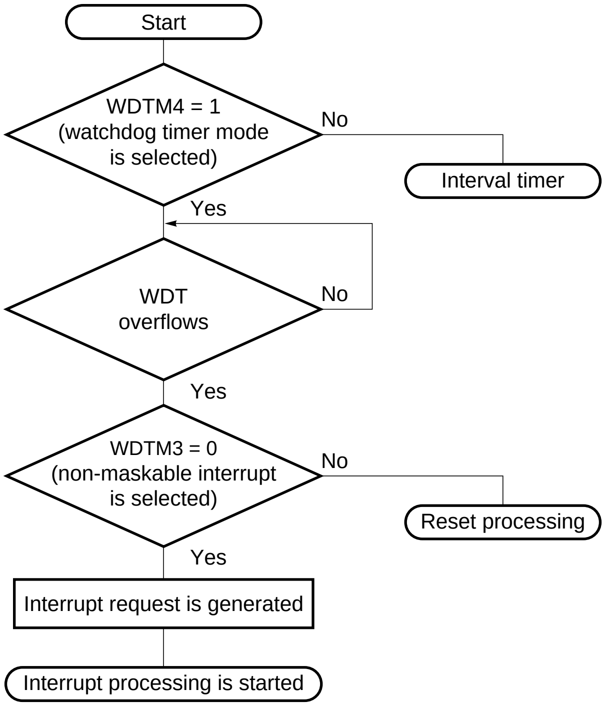
The non-maskable interrupt request is unconditionally accepted even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

When the non-maskable interrupt request is acknowledged, PSW and PC are saved to the stack in that order, the IE flag is reset to 0, the contents of the vector table are loaded to the PC, and then program execution branches.

Figure 9-6 shows the flowchart from non-maskable interrupt request generation to acceptance. Figure 9-7 shows the timing of non-maskable interrupt request acceptance. Figure 9-8 shows the acceptance operation if multiple non-maskable interrupts are generated.

Caution During a non-maskable interrupt service program execution, do not input another non-maskable interrupt request; if it is input, the service program will be interrupted and the new non-maskable interrupt request will be acknowledged.

★ **Figure 9-6. Flowchart from Non-Maskable Interrupt Request Generation to Acceptance**



WDTM	:	watchdog timer mode register
WDT	:	watchdog timer

Figure 9-7. Timing of Non-Maskable Interrupt Request Acceptance

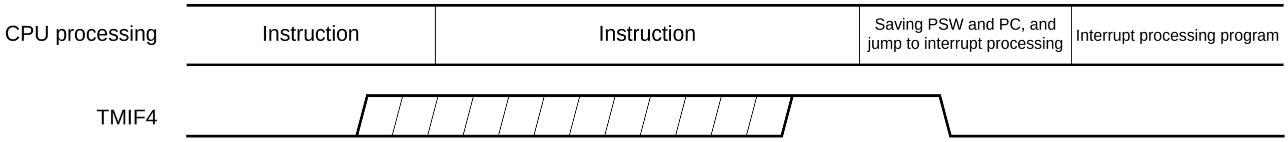
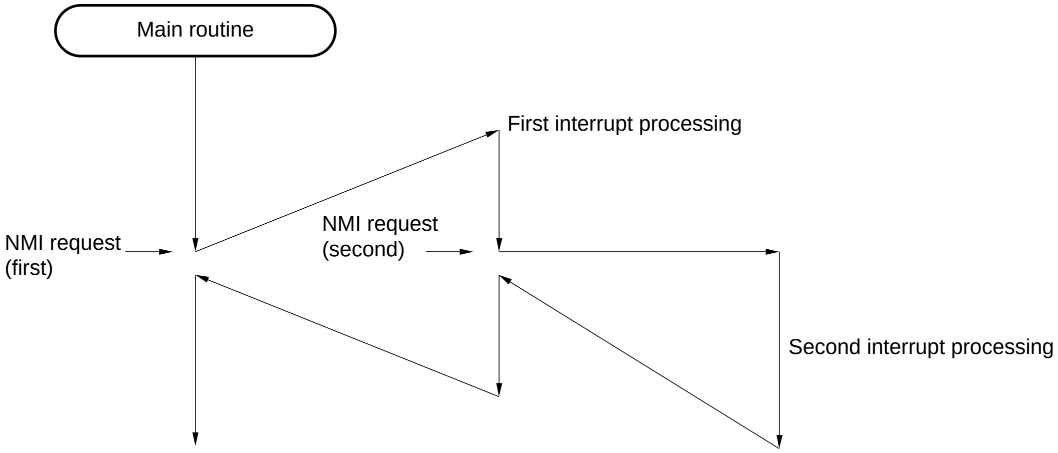


Figure 9-8. Accepting Non-Maskable Interrupt Request



9.4.2 Maskable interrupt acceptance operation

A maskable interrupt request can be accepted when the interrupt request flag is set to 1 and the corresponding interrupt mask flag is cleared to 0. A vectored interrupt request is accepted in the interrupt enabled status (when the IE flag is set to 1).

The time required to start the interrupt processing after a maskable interrupt request has been generated is shown in Table 9-3.

Refer to Figures 9-10 and 9-11 for the interrupt request acceptance timing.

Table 9-3. Time from Generation of Maskable Interrupt Request to Processing

Minimum Time	Maximum Time[Note]
9 clocks	19 clocks

Note The wait time is maximum when an interrupt request is generated immediately before BT and BF instruction.

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

When two or more maskable interrupt requests are generated at the same time, they are accepted starting from the interrupt request assigned the highest priority.

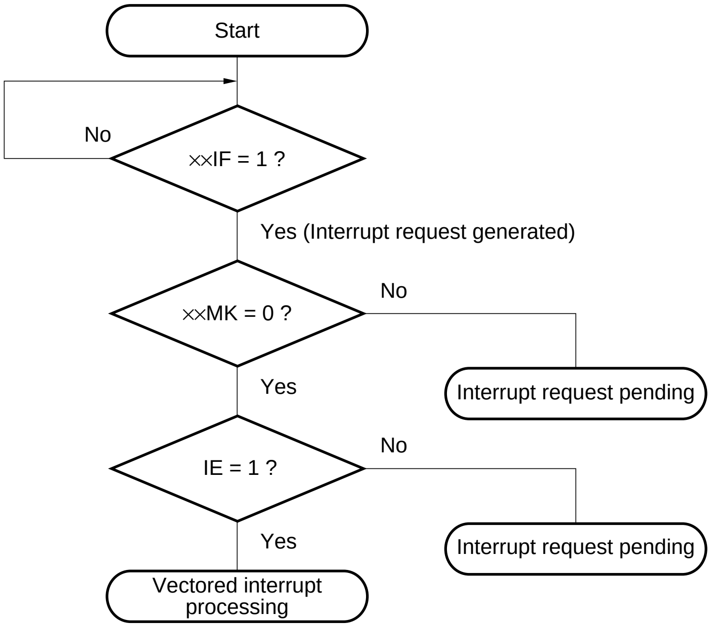
A pended interrupt is accepted when the status where it can be accepted is set.

Figure 9-9 shows the algorithm of accepting interrupt requests.

When a maskable interrupt request is accepted, the contents of PSW and PC are saved to the stack in that order, the IE flag is reset to 0, and the data in the vector table determined for each interrupt request is loaded to the PC, and execution branches.

To return from interrupt processing, use the RETI instruction.

Figure 9-9. Interrupt Request Acceptance Program Algorithm



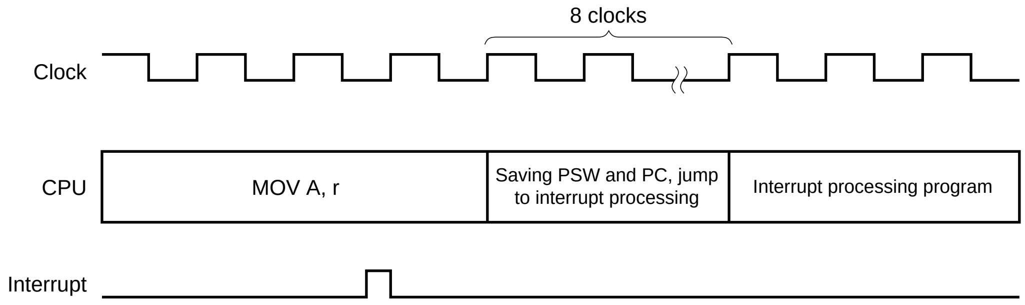
Start
No
××IF = 1 ?
Yes (Interrupt request generated)
××MK = 0 ?
No
Yes
Interrupt request pending
IE = 1 ?
No
Yes
Interrupt request pending
Vectored interrupt processing

××IF : interrupt request flag

××MK : interrupt mask flag

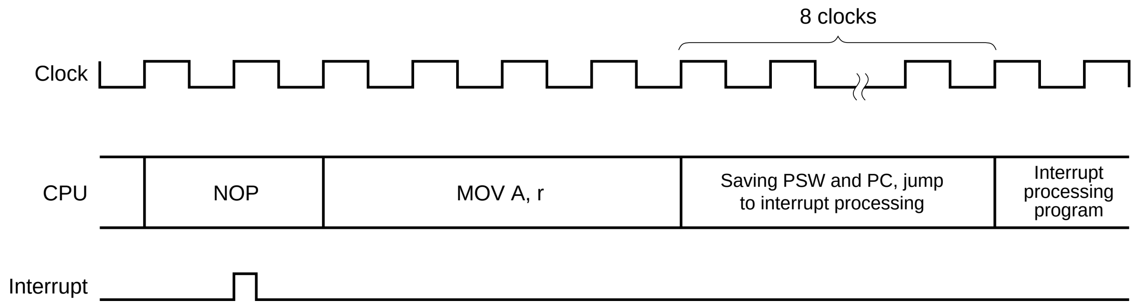
IE : flag to control maskable interrupt request acceptance (1 = enable, 0 = disable)

Figure 9-10. Interrupt Request Acceptance Timing (example of MOV A, r)


8 clocks
Clock
CPU
MOV A, r
Saving PSW and PC, jump to interrupt processing
Interrupt processing program
Interrupt

If an interrupt request flag (××IF) is set before an instruction clock n (n = 4 to 10) under execution becomes n–1, the interrupt is accepted after the instruction under execution completes. Figure 9-10 shows an example of the interrupt request acceptance timing for an 8-bit data transfer instruction MOV A, r. Since this instruction is executed for 4 clocks, if an interrupt occurs for 3 clocks after the execution starts, the interrupt acceptance processing is performed after the MOV A, r instruction is completed.

Figure 9-11. Interrupt Request Acceptance Timing (When interrupt request flag generates at the last clock during instruction execution)


8 clocks
Clock
CPU
NOP
MOV A, r
Saving PSW and PC, jump to interrupt processing
Interrupt processing program
Interrupt

If an interrupt request flag (××IF) is set at the last clock of the instruction, the interrupt acceptance processing starts after the next instruction is executed. Figure 9-11 shows an example of the interrupt acceptance timing for an interrupt request flag that is set at the second clock of NOP (2-clock instruction). In this case, the MOV A, r instruction after the NOP instruction is executed, and then the interrupt acceptance processing is performed.

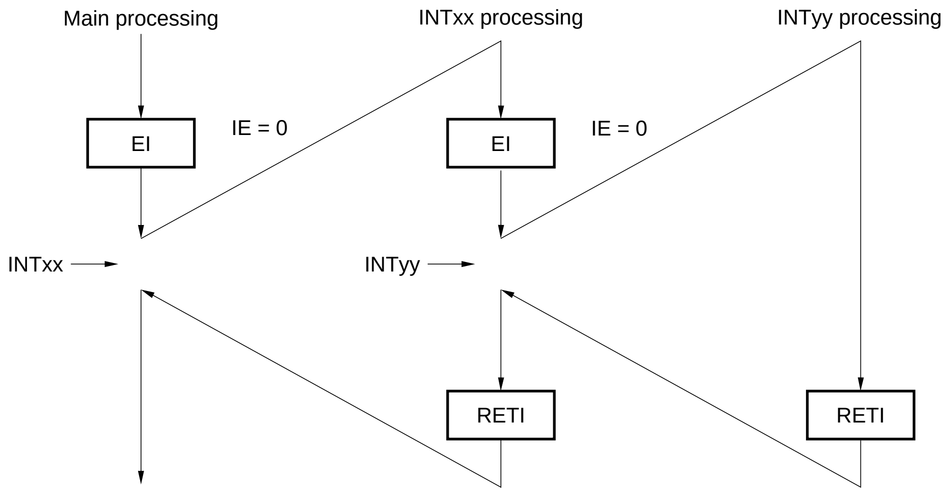
Caution Interrupt requests are reserved while the interrupt request flag register 0 (IF0) or the interrupt mask flag register 0 (MK0) is being accessed.

9.4.3 Multiplexed interrupt processing

Multiplexed interrupt processing in which another interrupt is accepted while an interrupt is processed can be processed by priority. When the priority is controlled by the default priority and two or more interrupts are generated at once, interrupt processing is performed according to the priority assigned to each interrupt request in advance (refer to **Table 9-1**).

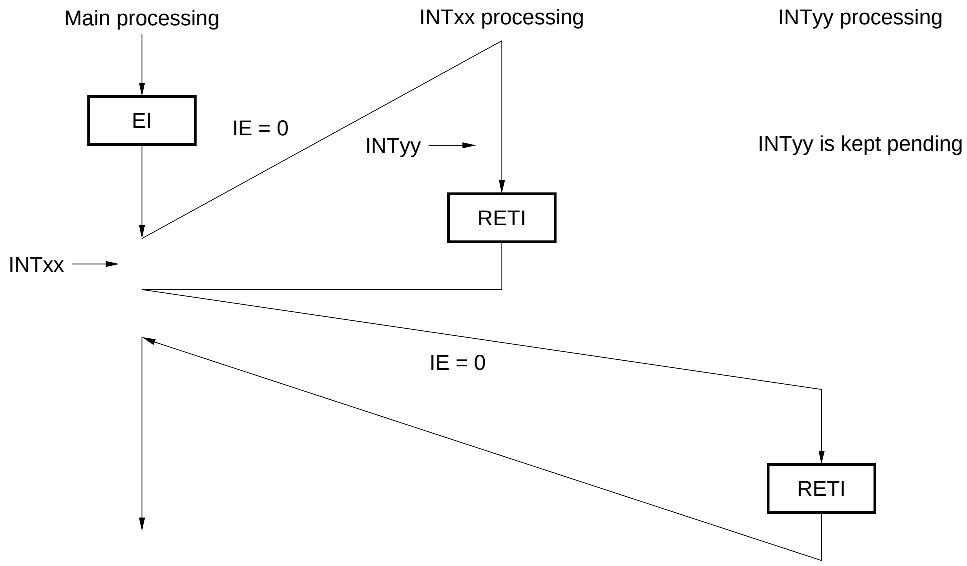
Figure 9-12. Example of Multiplexed Interrupt

Example 1. Multiplexed interrupt is accepted



During interrupt INTxx servicing, interrupt request INTyy is accepted, and a multiple interrupt is generated. An EI instruction is issued before each interrupt request acceptance, and the interrupt request acceptance enable state is set.

Example 2. A multiple interrupt is not generated because interrupts are not enabled



Because interrupts are not enabled in interrupt INTxx servicing (an EI instruction is not issued), interrupt request INTyy is not accepted, and a multiple interrupt is not generated. The INTyy request is reserved and accepted after the INTxx processing is performed.

IE = 0: Interrupt request acceptance disabled

9.4.4 Interrupt request reserve

Some instructions may reserve the acceptance of an instruction request until the completion of the execution of the next instruction even if the interrupt request (maskable interrupt, non-maskable interrupt, and external interrupt) is generated during the execution. The following shows such instructions (interrupt request reserve instruction).

- Manipulation instruction for the interrupt request flag register 0 (IF0)
- Manipulation instruction for the interrupt mask flag register 0 (MK0)

[MEMO]

CHAPTER 10   STANDBY FUNCTION

10.1 Standby Function and Configuration

10.1.1 Standby function

The standby function is to reduce the power dissipation of the system and can be effected in the following two modes:

(1) HALT mode

This mode is set when the HALT instruction is executed. The HALT mode stops the operation clock of the CPU. The system clock oscillator continues oscillating. This mode does not reduce the power dissipation as much as the STOP mode, but is useful for resuming processing immediately when an interrupt request is generated, or for intermittent operations.

(2) STOP mode

This mode is set when the STOP instruction is executed. The STOP mode stops the system clock oscillator and stops the entire system. The power dissipation of the CPU can be substantially reduced in this mode. The low voltage (V_{DD} = 1.8 V) of the data memory can be retained. Therefore, this mode is useful for retaining the contents of the data memory at an extremely low current.

The STOP mode can be released by an interrupt request, so that this mode can be used for intermittent operation. However, some time is required until the system clock oscillator stabilizes after the STOP mode has been released. If processing must be resumed immediately by using an interrupt request, therefore, use the HALT mode.

In both modes, the previous contents of the registers, flags, and data memory before setting the standby mode are all retained. In addition, the statuses of the output latch of the I/O ports and output buffer are also retained.

Caution To set the STOP mode, be sure to stop the operations of the peripheral hardware, and then execute the STOP instruction.

10.1.2 Standby function control register

The wait time after the STOP mode is released upon interrupt request until the oscillation stabilizes is controlled with the oscillation stabilization time select register (OSTS).

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

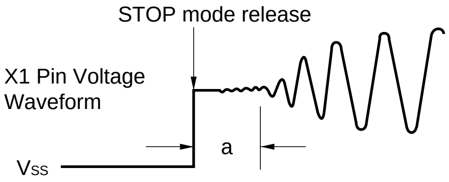
$\overline{\text{RESET}}$ input sets OSTS to 04H. However, the oscillation stabilization time after $\overline{\text{RESET}}$ input is $2^{15}/f_X$, instead of $2^{17}/f_X$.

Figure 10-1. Oscillation Stabilization Time Select Register 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	Oscillation Stabilization Time Selection
0	0	0	$2^{12}/f_X$ (819 μs)
0	1	0	$2^{15}/f_X$ (6.55 ms)
1	0	0	$2^{17}/f_X$ (26.2 ms)
Other than above			Setting prohibited

Caution The wait time after the STOP mode is released does not include the time from STOP mode release to clock oscillation start ("a" in the figure below), regardless of release by $\overline{\text{RESET}}$ input or by interrupt generation.



Remarks **1.** f_X : system clock oscillation frequency

2. Figures in parentheses applies to operation when f_X = 5.0 MHz.

10.2 Operation of Standby Function

10.2.1 HALT mode

(1) HALT mode

The HALT mode is set by executing the HALT instruction.

The operation status in the HALT mode is shown in the following table.

Table 10-1. HALT Mode Operating Status

Item	HALT Mode Operating Status
Clock generator	Enables system clock oscillation Stops clock supply to CPU
CPU	Stops operation
Port (Output latch)	Retains the status before setting the HALT mode
8-bit timer/event counter	Enables operation
Watchdog timer	Enables operation
Serial interface	Enables operation
External interrupt	Enables operation

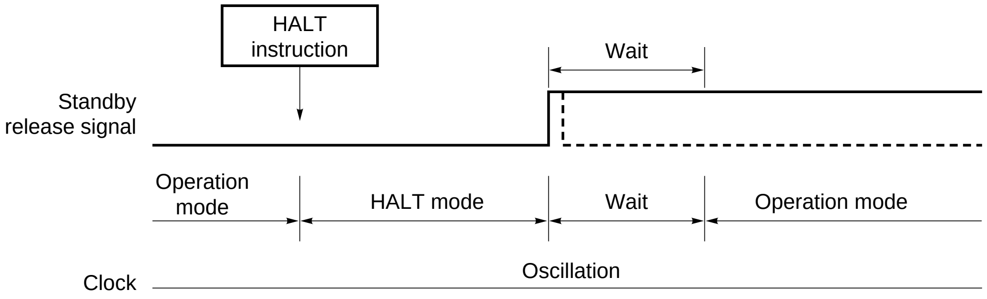
(2) Releasing HALT mode

The HALT mode can be released by the following three types of sources:

(a) Releasing by unmasked interrupt request

The HALT mode is released by an unmasked interrupt request. In this case, if the interrupt request is enabled to be accepted, vectored interrupt processing is performed. If the interrupt is disabled, the instruction at the next address is executed.

Figure 10-2. Releasing HALT Mode by Interrupt



Remarks 1. The broken line indicates the case where the interrupt request that has released the standby mode is accepted.

2. The wait time is as follows:

- When vectored interrupt processing is performed : 9 to 10 clocks
- When vectored interrupt processing is not performed : 1 to 2 clocks

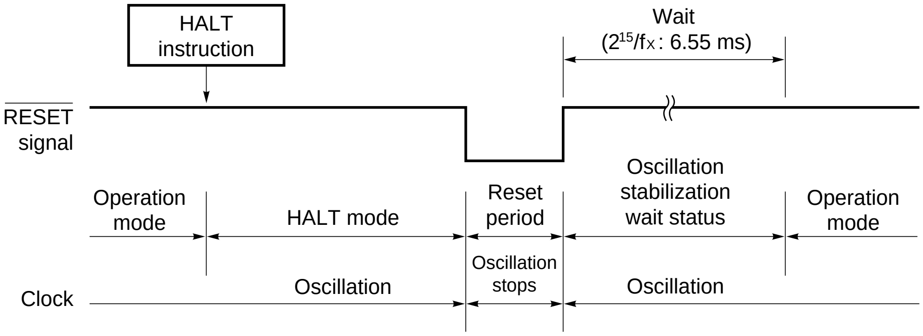
(b) Releasing by non-maskable interrupt request

The HALT mode is released regardless of whether the interrupt is enabled or disabled, and vectored interrupt processing is performed.

(c) Releasing by $\overline{\text{RESET}}$ input

When the HALT mode is released by the $\overline{\text{RESET}}$ signal, execution branches to the reset vector address in the same manner as the ordinary reset operation, and program execution is started.

Figure 10-3. Releasing HALT Mode by $\overline{\text{RESET}}$ Input



Remarks 1. fx : system clock oscillation frequency

2. ( ) : at fx = 5.0-MHz operation

Table 10-2. Operation after Release of HALT Mode

Releasing Source	MK××	IE	Operation
Maskable interrupt request	0	0	Executes next address instruction
	0	1	Executes interrupt processing
	1	×	Retains HALT mode
Non-maskable interrupt request	—	×	Executes interrupt processing
$\overline{\text{RESET}}$ input	—	—	Reset processing

×: don't care

10.2.2 STOP mode

(1) Setting and operation status of STOP mode

The STOP mode is set by executing the STOP instruction.

Cautions 1. When the STOP mode is set, the X2 pin is internally pulled up to V_{DD} to suppress the current leakage of the crystal oscillation circuit block. Therefore, do not use the STOP mode in a system where the external clock is used as the system clock.

2. Because the standby mode can be released by an interrupt request signal, the standby mode is released as soon as it is set if there is an interrupt source whose interrupt request flag is set and interrupt mask flag is reset. When the STOP mode is set, therefore, the HALT mode is set immediately after the STOP instruction has been executed, the wait time set by the oscillation stabilization time select register (OSTS) elapses, and then an operation mode is set.

The operation status in the STOP mode is shown in the following table.

Table 10-3. STOP Mode Operating Status

Item	STOP Mode Operating Status
Clock generator	Stops system clock oscillation
CPU	Stops operation
Port (Output latch)	Retains the status before setting the STOP mode
8-bit timer/event counter	Enables operation only when TI0 or TI1 is selected as count clock
Watchdog timer	Stops operation
Serial interface	Enables operation only when input clock from external is selected as serial clock
External interrupt	Enables operation

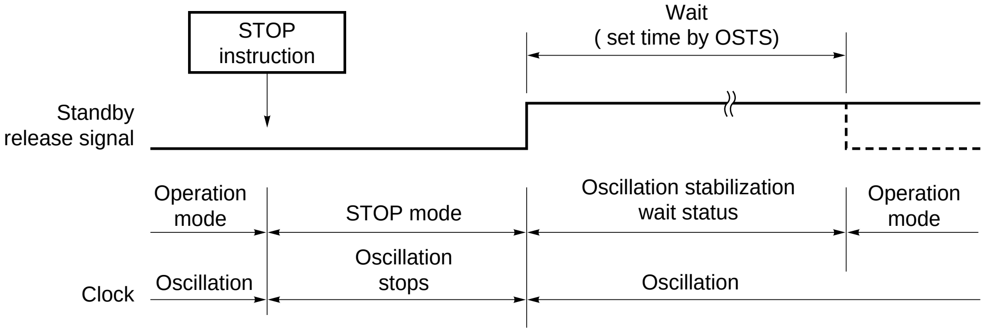
(2) Releasing STOP mode

The STOP mode can be released by the following two types of sources:

(a) Releasing by unmasked interrupt request

The STOP mode can be released by an unmasked interrupt request. In this case, if the interrupt is enabled to be accepted, vectored interrupt processing is performed, after the oscillation stabilization time has elapsed. If the interrupt acceptance is disabled, the instruction at the next address is executed.

Figure 10-4. Releasing STOP Mode by Interrupt

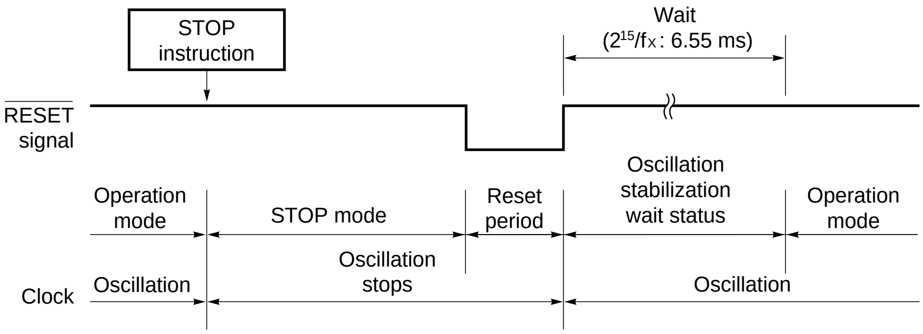


Remark The broken line indicates the case where the interrupt request that has released the standby mode is accepted.

(b) Releasing by $\overline{\text{RESET}}$ input

When the STOP mode is released by the $\overline{\text{RESET}}$ signal, the reset operation is performed after the oscillation stabilization time has elapsed.

Figure 10-5. Releasing STOP Mode by $\overline{\text{RESET}}$ Input



Remarks 1. fx : system clock oscillation frequency

2. ( ) : at fx = 5.0-MHz operation

Table 10-4. Operation after Release of STOP Mode

Releasing Source	MK××	IE	Operation
Maskable interrupt request	0	0	Executes next address instruction
	0	1	Executes interrupt processing
	1	×	Retains STOP mode
$\overline{\text{RESET}}$ input	—	—	Reset processing

×: don't care

CHAPTER 11 RESET FUNCTION

The following two operations are available to generate reset signals.

(1) External reset input with $\overline{\text{RESET}}$ pin

(2) Internal reset by program runaway time detected with watchdog timer

External and internal reset have no functional differences. In both cases, program execution starts at the address at 0000H and 0001H by reset signal input.

When a low level is input to the $\overline{\text{RESET}}$ pin or the watchdog timer overflows, a reset is applied and each hardware is set to the status shown in Table 11-1. Each pin has a high impedance during reset input or during oscillation stabilization time just after reset clear.

When a high level is input to the $\overline{\text{RESET}}$ pin, the reset is cleared and program execution is started after the oscillation stabilization time (2^{15}/fx) has elapsed. The reset applied by the watchdog timer overflow is automatically cleared after reset, and program execution is started after the oscillation stabilization time (2^{15}/fx) has elapsed (see Figures 11-2 through 11-4).

Cautions 1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.

2. When the STOP mode is cleared by reset, the STOP mode contents are held during reset input. However, the port pins become high impedance.

Figure 11-1. Block Diagram of Reset Function

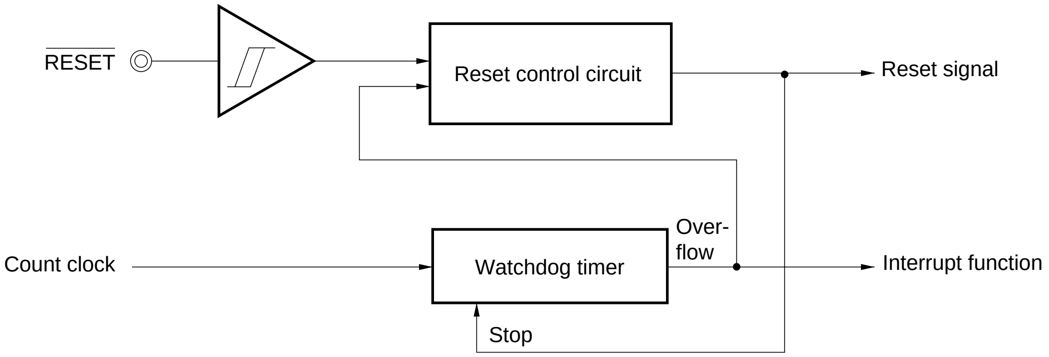


Figure 11-2. Reset Timing by $\overline{\text{RESET}}$ Input

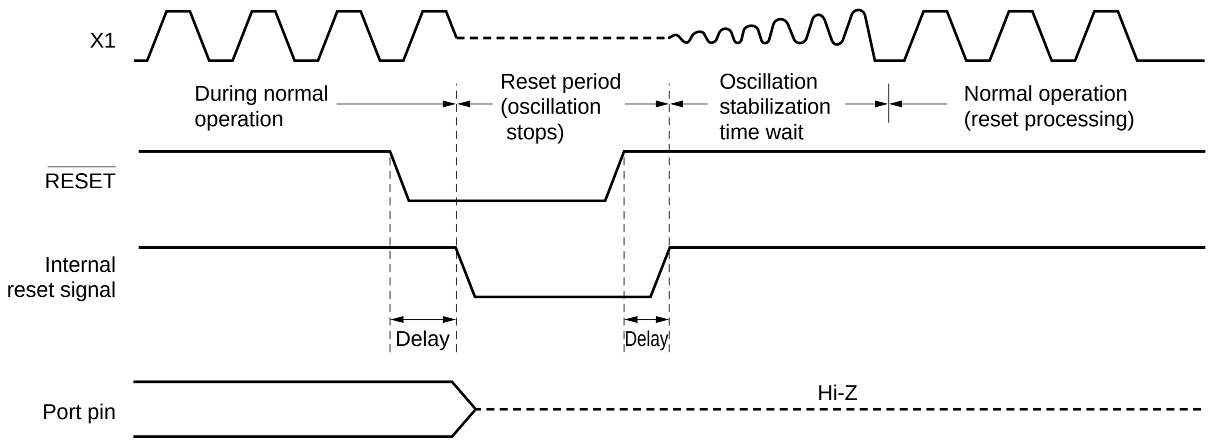

X1
During normal operation
Reset period (oscillation stops)
Oscillation stabilization time wait
Normal operation (reset processing)
RESET
Internal reset signal
Delay
Delay
Port pin
Hi-Z

Figure 11-3. Reset Timing by Overflow in Watchdog Timer

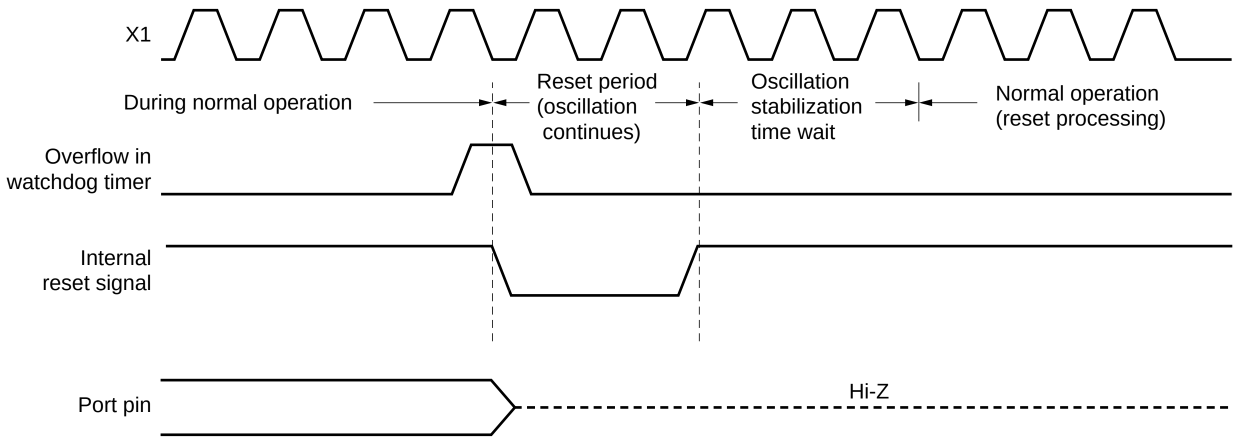

X1
During normal operation
Reset period (oscillation continues)
Oscillation stabilization time wait
Normal operation (reset processing)
Overflow in watchdog timer
Internal reset signal
Port pin
Hi-Z

Figure 11-4. Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode

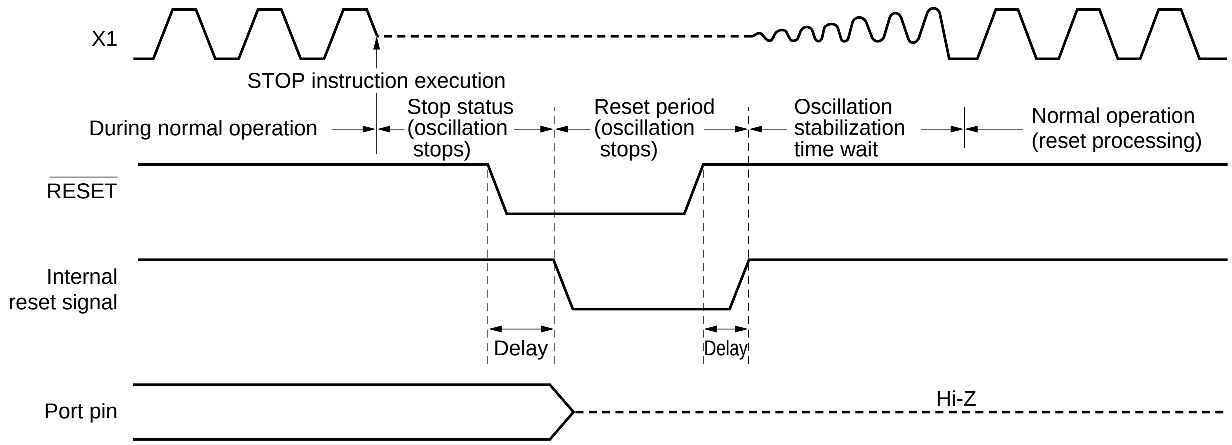

X1
STOP instruction execution
During normal operation
Stop status (oscillation stops)
Reset period (oscillation stops)
Oscillation stabilization time wait
Normal operation (reset processing)
RESET
Internal reset signal
Delay
Delay
Port pin
Hi-Z

Table 11-1. Hardware Status after Reset

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 (P0 to P3) (Output latch)		00H
Port mode register (PM0 to PM3)		FFH
Pull-up resistor option register (PUO)		00H
Processor clock control register (PCC)		02H
Oscillation stabilization time select register (OSTS)		04H
8-bit timer/event counter	Timer register (TM00, TM01)	00H
	Compare register (CR00, CR01)	Undefined
	Mode control register (TMC00, TMC01)	00H
Watchdog timer	Timer clock select register (TCL2)	00H
	Mode register (WDTM)	00H
Serial interface	Mode register (CSIM00)	00H
	Asynchronous serial interface mode register (ASIM00)	00H
	Asynchronous serial interface status register (ASIS00)	00H
	Baud rate generator control register (BRGC00)	00H
	Transmit shift register (TXS00)	FFH
	Receive buffer register (RXB00)	Undefined
Interrupt	Request flag register (IF0)	00H
	Mask flag register (MK0)	FFH
	External interrupt mode register (INTM0)	00H

Notes

1. During reset input and oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined.
   All other hardware remains unchanged after reset.
2. The post-reset values are retained in the standby mode.

[MEMO]

CHAPTER 12 μPD78P9014

The μPD78P9014 is a version with an internal ROM of the μPD789011 and 789012 expanded and replaced with a one-time PROM. The differences between the μPD78P9014 and the mask ROM versions are shown in Table 12-1.

Table 12-1. Differences between μPD78P9014 and Mask ROM Versions

Item		One-Time PROM Version	Mask ROM Version	
		μPD78P9014	μPD789011	μPD789012
Internal memory	ROM	8 Kbytes	2 Kbytes	4 Kbytes
	High-speed RAM	256 bytes	128 bytes	
IC pin		Not provided	Provided	
V_{PP} pin		Provided	Not provided	
Electrical specifications		Refer to each data sheet.		

★ **Caution There are differences in noise immunity and noise radiation between the PROM and mask ROM versions. When pre-producing an application set with the PROM version and then mass-producing it with the mask ROM version, be sure to conduct sufficient evaluations for the commercial samples (not engineering samples) of the mask ROM version.**

12.1 PROM Programming

An on-chip program memory in the μPD78P9014 is an 8-Kbyte one-time PROM that can be written electrically. Write/verify of this one-time PROM uses the pins shown in Table 12-2. For the connection of unused pins, refer to **1.4 Pin Configuration (Top View) (2) PROM programming mode**.

Addresses are updated through clock input from the X1 pin instead of address input.

Table 12-2. Pins in PROM Programming Mode

Pin	Function
V_{PP}	PROM programming mode setting, and high voltage applied during program write/verify (normally V_{DD} potential).
MD0 to MD3	Pin used for selecting operating mode during program write/verify.
D0 to D7	Data bus
X1	Address update clock input during program write/verify.
V_{DD}	PROM programming mode setting and power supply voltage application pin. Normally, 1.8 to 5.5 V and +5.5 V are applied in normal operating mode and PROM programming mode, respectively.

12.1.1 Operating modes

When +5.5 V is applied to the V_{DD} pin, and +12.5 V is applied to the V_{PP} pin, the PROM programming mode is set. This mode will become the operating mode as shown in Table 12-3 when the MD0 to MD3 pins are set as shown.

Table 12-3. Operating Modes of PROM Programming

Pin Operating Mode	V_{PP}	V_{DD}	MD0	MD1	MD2	MD3
Program memory address 0 clear	+12.5	+5.5	H	L	H	L
Write mode			L	H	H	H
Verify mode			L	L	H	H
Program inhibit mode			H	×	H	H

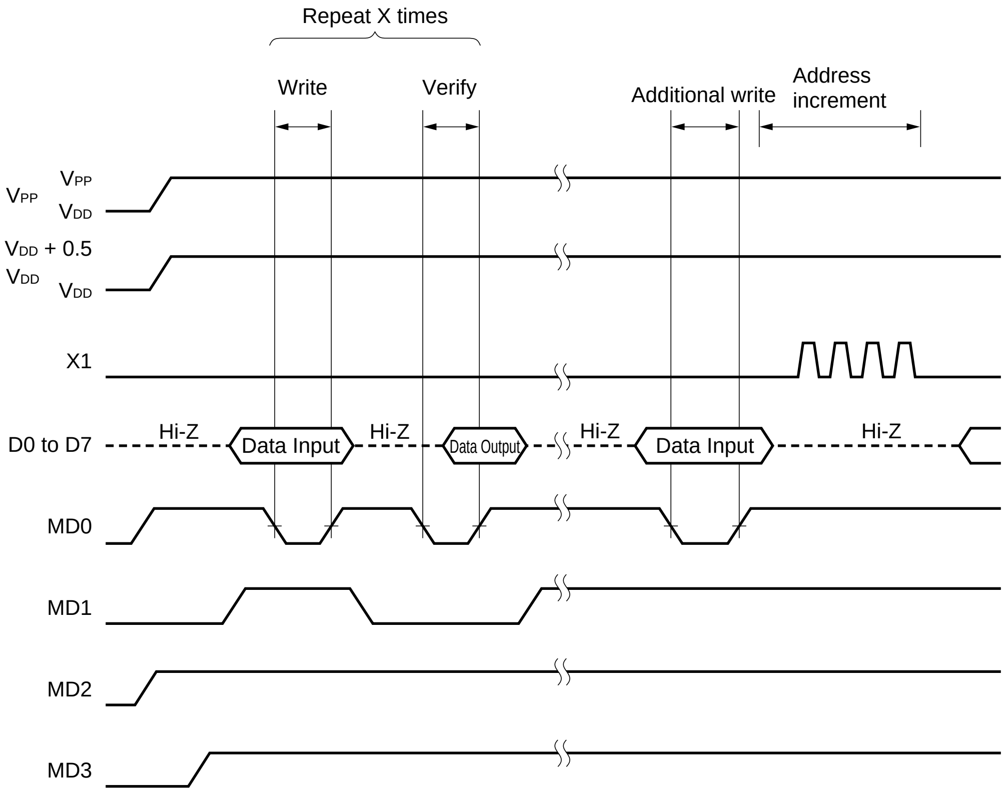
×: L or H

12.1.2 Program memory write procedure

Program memory is written by the following procedure. High-speed writing is possible.

(1) Pull down unused pins to V_{SS} via a resistor. Fix the X1 pin at low level.
(2) Supply 5 V to the V_{DD} and V_{PP} pins.
(3) 10 μs wait.
(4) Program memory address 0 clear mode.
(5) Supply +5.5 V to V_{DD} and +12.5 V to V_{PP}.
(6) Write data with 1-ms write mode.
(7) Verify mode. Proceed to step (8) if write was performed, or repeat steps (6), (7) if write was not performed.
(8) (X: Number of writes in steps (6), (7)) × 1-ms additional write.
(9) Program memory address incremented (+1) through input of 4 pulses to X1 pin.
(10) Repeat steps (6) to (9) until last address.
(11) Program memory address 0 clear mode.
(12) Change V_{DD} and V_{PP} pins voltage to 5 V.
(13) Power off.

Steps (2) to (9) of the above procedure are shown in the figure below.

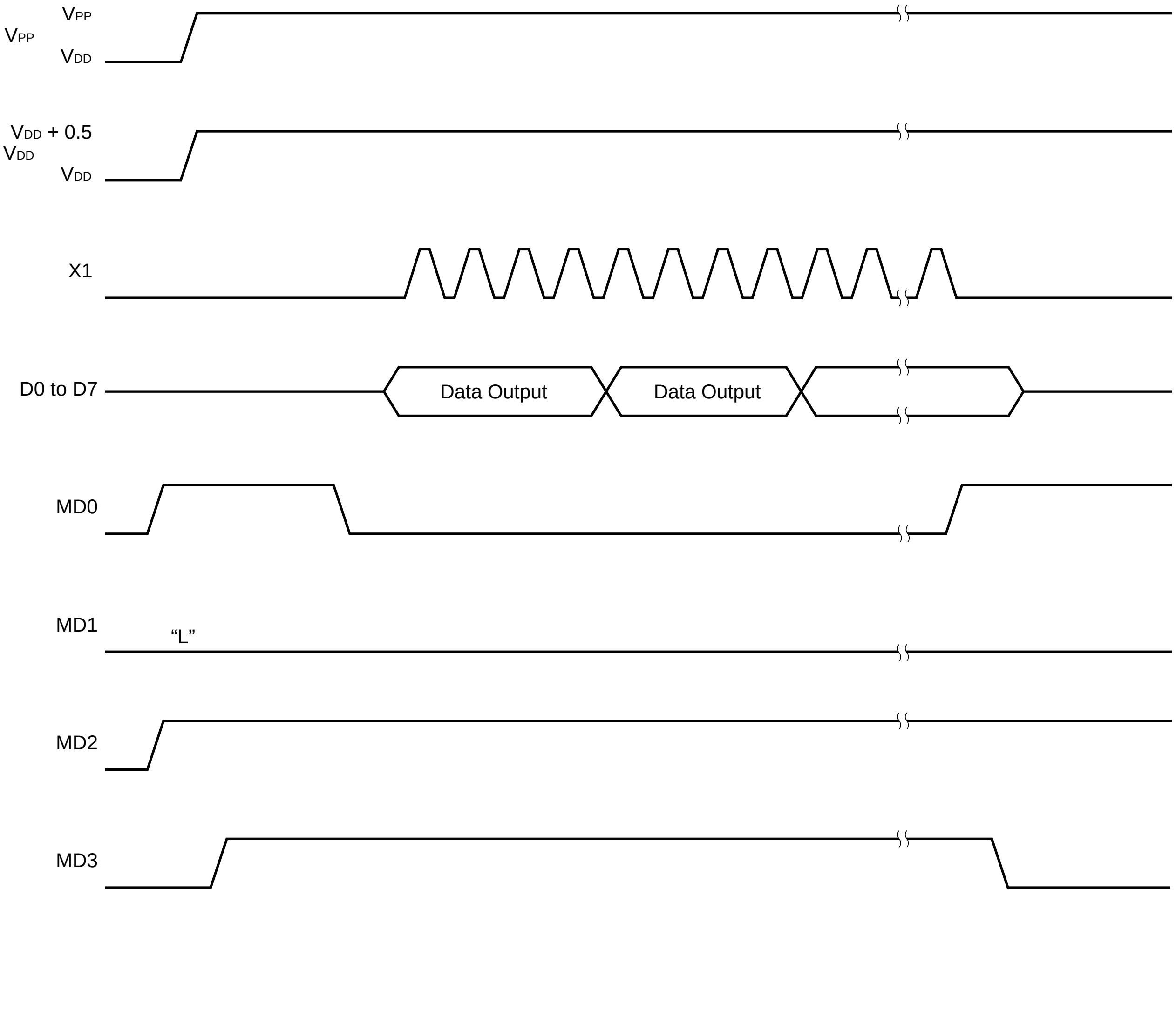


12.1.3 Program memory read procedure

Program memory is read by the following procedure.

(1) Pull down unused pins to V_{SS} via a resistor. Fix the X1 pin at low level.
(2) Supply +5 V to the V_{DD} and V_{PP} pins.
(3) 10 μs wait.
(4) Program memory address 0 clear mode.
(5) Supply +5.5 V to V_{DD} and +12.5 V to V_{PP}.
(6) Verify mode. Data are output as one address sequentially every fourth pulse cycles when clock pulse is fed to the X1 pin.
(7) Program memory address 0 clear mode.
(8) Change V_{DD} and V_{PP} pins voltage to +5 V.
(9) Power off.

Steps (2) to (7) of the above procedure are shown in the figure below.



12.1.4 One-time PROM screening

The one-time PROM version, due to its structure, cannot be fully tested by NEC before shipping. It is recommended to perform screening to verify PROM after writing necessary data and performing high-temperature storage under the conditions indicated below.

Storage Temperature	Storage Time
125°C	24 hours

NEC offers the services with charged to write through mark, screen, and verify a one-time PROM version as we call the “QTOP^{TM} Microcontroller”. For details, contact an NEC sales representative.

Phase-out/Discontinued

[MEMO]

CHAPTER 13 INSTRUCTION SET

This chapter lists the instruction set of the μPD789014 Subseries. For the details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series User's Manual — Instruction (U11047E)**.

13.1 Operation

13.1.1 Operand identifiers and description methods

Operands are described in "Operand" column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for detail). When there are two or more description methods, select one of them. Alphabetic letters in capitals and symbols, #, !, $, and [ ] are key words and are described as they are. Each symbol has the following meaning.

- # : Immediate data specification
- ! : Absolute address specification
- $: Relative address specification
- [ ] : Indirect address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, $ and [ ] symbols.

For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parenthesis in the table below, R0, R1, R2, etc.) can be used for description.

Table 13-1. Operand Identifiers and Description Methods

Identifier	Description Method
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special-function register symbol
saddr	FE80H to FF1FH Immediate data or labels
saddrp	FE80H to FF1FH Immediate data or labels (even addresses only)
addr16	0000H to FFFFH Immediate data or labels (only even addresses for 16-bit data transfer instructions)
addr5	0040H to 007FH Immediate data or labels (even addresses only)
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label

Remark Refer to **Table 3-4 Special Function Register List** for symbols of special function registers.

13.1.2 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
IE	: Interrupt request enable flag
NMIS	: Flag indicating non-maskable interrupt servicing in progress
( )	: Memory contents indicated by address or register contents in parenthesis
$\times_H$, $\times_L$	: Higher 8 bits and lower 8 bits of 16-bit register
∧	: Logical product (AND)
∨	: Logical sum (OR)
⊻	: Exclusive logical sum (exclusive OR)
—	: Inverted data
addr16	: 16-bit immediate data or label
jdisp8	: Signed 8-bit data (displacement value)

13.1.3 Description of “flag operation” column

(Blank)	: Unchanged
0	: Cleared to 0
1	: Set to 1
×	: Set/cleared according to the result
R	: Previously saved value is restored

13.2 Operation List

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
MOV	r, #byte	3	6	r ← byte			
	saddr, #byte	3	6	(saddr) ← byte			
	sfr, #byte	3	6	sfr ← byte			
	A, r[Note 1]	2	4	A ← r			
	r, A[Note 1]	2	4	r ← A			
	A, saddr	2	4	A ← (saddr)			
	saddr, A	2	4	(saddr) ← A			
	A, sfr	2	4	A ← sfr			
	sfr, A	2	4	sfr ← A			
	A, !addr16	3	8	A ← (addr16)			
	!addr16, A	3	8	(addr16) ← A			
	PSW, #byte	3	6	PSW ← byte	×	×	×
	A, PSW	2	4	A ← PSW			
	PSW, A	2	4	PSW ← A	×	×	×
	A, [DE]	1	6	A ← (DE)			
	[DE], A	1	6	(DE) ← A			
	A, [HL]	1	6	A ← (HL)			
	[HL], A	1	6	(HL) ← A			
	A, [HL+byte]	2	6	A ← (HL+byte)			
	[HL+byte], A	2	6	(HL+byte) ← A			
XCH	A, X	1	4	A ↔ X			
	A, r[Note 2]	2	6	A ↔ r			
	A, saddr	2	6	A ↔ (saddr)			
	A, sfr	2	6	A ↔ sfr			
	A, [DE]	1	8	A ↔ (DE)			
	A, [HL]	1	8	A ↔ (HL)			
	A, [HL+byte]	2	8	A ↔ (HL+byte)			

Notes **1.** Except r = A.

2. Except r = A, X.

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
MOVW	rp, #word	3	6	rp ← word			
	AX, saddrp	2	6	AX ← (saddrp)			
	saddrp, AX	2	8	(saddrp) ← AX			
	AX, rp[Note]	1	4	AX ← rp			
	rp, AX[Note]	1	4	rp ← AX			
XCHW	AX, rp[Note]	1	8	AX ↔ rp			
ADD	A, #byte	2	4	A, CY ← A + byte	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte	×	×	×
	A, r	2	4	A, CY ← A + r	×	×	×
	A, saddr	2	4	A, CY ← A + (saddr)	×	×	×
	A, !addr16	3	8	A, CY ← A + (addr16)	×	×	×
	A, [HL]	1	6	A, CY ← A + (HL)	×	×	×
	A, [HL+byte]	2	6	A, CY ← A + (HL+byte)	×	×	×
ADDC	A, #byte	2	4	A, CY ← A + byte + CY	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte + CY	×	×	×
	A, r	2	4	A, CY ← A + r + CY	×	×	×
	A, saddr	2	4	A, CY ← A + (saddr) + CY	×	×	×
	A, !addr16	3	8	A, CY ← A + (addr16) + CY	×	×	×
	A, [HL]	1	6	A, CY ← A + (HL) + CY	×	×	×
	A, [HL+byte]	2	6	A, CY ← A + (HL+byte) + CY	×	×	×
SUB	A, #byte	2	4	A, CY ← A – byte	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte	×	×	×
	A, r	2	4	A, CY ← A – r	×	×	×
	A, saddr	2	4	A, CY ← A – (saddr)	×	×	×
	A, !addr16	3	8	A, CY ← A – (addr16)	×	×	×
	A, [HL]	1	6	A, CY ← A – (HL)	×	×	×
	A, [HL+byte]	2	6	A, CY ← A – (HL+byte)	×	×	×

Note Only when rp = BC, DE, or HL.

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
SUBC	A, #byte	2	4	A, CY ← A – byte – CY	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte – CY	×	×	×
	A, r	2	4	A, CY ← A – r – CY	×	×	×
	A, saddr	2	4	A, CY ← A – (saddr) – CY	×	×	×
	A, !addr16	3	8	A, CY ← A – (addr16) – CY	×	×	×
	A, [HL]	1	6	A, CY ← A – (HL) – CY	×	×	×
	A, [HL+byte]	2	6	A, CY ← A – (HL+byte) – CY	×	×	×
AND	A, #byte	2	4	A ← A ∧ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∧ byte	×		
	A, r	2	4	A ← A ∧ r	×		
	A, saddr	2	4	A ← A ∧ (saddr)	×		
	A, !addr16	3	8	A ← A ∧ (addr16)	×		
	A, [HL]	1	6	A ← A ∧ (HL)	×		
	A, [HL+byte]	2	6	A ← A ∧ (HL+byte)	×		
OR	A, #byte	2	4	A ← A ∨ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∨ byte	×		
	A, r	2	4	A ← A ∨ r	×		
	A, saddr	2	4	A ← A ∨ (saddr)	×		
	A, !addr16	3	8	A ← A ∨ (addr16)	×		
	A, [HL]	1	6	A ← A ∨ (HL)	×		
	A, [HL+byte]	2	6	A ← A ∨ (HL+byte)	×		
XOR	A, #byte	2	4	A ← A ∀ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∀ byte	×		
	A, r	2	4	A ← A ∀ r	×		
	A, saddr	2	4	A ← A ∀ (saddr)	×		
	A, !addr16	3	8	A ← A ∀ (addr16)	×		
	A, [HL]	1	6	A ← A ∀ (HL)	×		
	A, [HL+byte]	2	6	A ← A ∀ (HL+byte)	×		

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
CMP	A, #byte	2	4	A – byte	×	×	×
	saddr, #byte	3	6	(saddr) – byte	×	×	×
	A, r	2	4	A – r	×	×	×
	A, saddr	2	4	A – (saddr)	×	×	×
	A, !addr16	3	8	A – (addr16)	×	×	×
	A, [HL]	1	6	A – (HL)	×	×	×
	A, [HL+byte]	2	6	A – (HL+byte)	×	×	×
ADDW	AX, #word	3	6	AX, CY ← AX + word	×	×	×
SUBW	AX, #word	3	6	AX, CY ← AX – word	×	×	×
CMPW	AX, #word	3	6	AX – word	×	×	×
INC	r	2	4	r ← r + 1	×	×	
	saddr	2	4	(saddr) ← (saddr) + 1	×	×	
DEC	r	2	4	r ← r – 1	×	×	
	saddr	2	4	(saddr) ← (saddr) – 1	×	×	
INCW	rp	1	4	rp ← rp + 1			
DECW	rp	1	4	rp ← rp – 1			
ROR	A, 1	1	2	(CY, A_7 ← A_0, A_{m-1} ← A_m) × 1			×
ROL	A, 1	1	2	(CY, A_0 ← A_7, A_{m+1} ← A_m) × 1			×
RORC	A, 1	1	2	(CY ← A_0, A_7 ← CY, A_{m-1} ← A_m) × 1			×
ROLC	A, 1	1	2	(CY ← A_7, A_0 ← CY, A_{m+1} ← A_m) × 1			×
SET1	saddr.bit	3	6	(saddr.bit) ← 1			
	sfr.bit	3	6	sfr.bit ← 1			
	A.bit	2	4	A.bit ← 1			
	PSW.bit	3	6	PSW.bit ← 1	×	×	×
	[HL].bit	2	10	(HL).bit ← 1			
CLR1	saddr.bit	3	6	(saddr.bit) ← 0			
	sfr.bit	3	6	sfr.bit ← 0			
	A.bit	2	4	A.bit ← 0			
	PSW.bit	3	6	PSW.bit ← 0	×	×	×
	[HL].bit	2	10	(HL).bit ← 0			
SET1	CY	1	2	CY ← 1			1
CLR1	CY	1	2	CY ← 0			0
NOT1	CY	1	2	CY ← $\overline{CY}$			×

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag
					Z AC CY
CALL	!addr16	3	6	(SP – 1) ← (PC + 3)$_H$, (SP – 2) ← (PC + 3)$_L$, PC ← addr16, SP ← SP – 2	
CALLT	[addr5]	1	8	(SP – 1) ← (PC + 1)$_H$, (SP – 2) ← (PC + 1)$_L$, PC$_H$ ← (00000000, addr5 + 1), PC$_L$ ← (00000000, addr5), SP ← SP – 2	
RET		1	6	PC$_H$ ← (SP + 1), PC$_L$ ← (SP), SP ← SP + 2	
RETI		1	8	PC$_H$ ← (SP + 1), PC$_L$ ← (SP), PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R R R
PUSH	PSW	1	2	(SP – 1) ← PSW, SP ← SP – 1	
	rp	1	4	(SP – 1) ← rp$_H$, (SP – 2) ← rp$_L$, SP ← SP – 2	
POP	PSW	1	4	PSW ← (SP), SP ← SP + 1	R R R
	rp	1	6	rp$_H$ ← (SP + 1), rp$_L$ ← (SP), SP ← SP + 2	
MOVW	SP, AX	2	8	SP ← AX	
	AX, SP	2	6	AX ← SP	
BR	!addr16	3	6	PC ← addr16	
	$addr16	2	6	PC ← PC + 2 + jdisp8	
	AX	1	6	PC$_H$ ← A, PC$_L$ ← X	
BC	$saddr16	2	6	PC ← PC + 2 + jdisp8 if CY = 1	
BNC	$saddr16	2	6	PC ← PC + 2 + jdisp8 if CY = 0	
BZ	$saddr16	2	6	PC ← PC + 2 + jdisp8 if Z = 1	
BNZ	$saddr16	2	6	PC ← PC + 2 + jdisp8 if Z = 0	
BT	saddr.bit, $addr16	4	10	PC ← PC + 4 + jdisp8 if (saddr.bit) = 1	
	sfr.bit, $addr16	4	10	PC ← PC + 4 + jdisp8 if sfr.bit = 1	
	A.bit, $addr16	3	8	PC ← PC + 3 + jdisp8 if A.bit = 1	
	PSW.bit, $addr16	4	10	PC ← PC + 4 + jdisp8 if PSW.bit = 1	
BF	saddr.bit, $addr16	4	10	PC ← PC + 4 + jdisp8 if (saddr.bit) = 0	
	sfr.bit, $addr16	4	10	PC ← PC + 4 + jdisp8 if sfr.bit = 0	
	A.bit, $addr16	3	8	PC ← PC + 3 + jdisp8 if A.bit = 0	
	PSW.bit, $addr16	4	10	PC ← PC + 4 + jdisp8 if PSW.bit = 0	
DBNZ	B, $addr16	2	6	B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0	
	C, $addr16	2	6	C ← C – 1, then PC ← PC + 2 + jdisp8 if C ≠ 0	
	saddr, $addr16	3	8	(saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if (saddr) ≠ 0	
NOP		1	2	No Operation	
EI		3	6	IE ← 1 (Enable Interrupt)	
DI		3	6	IE ← 0 (Disable Interrupt)	
HALT		1	2	Set HALT Mode	
STOP		1	2	Set STOP Mode	

Remark One instruction clock cycle is one CPU clock cycle (f$_{CPU}$) selected by processor clock control register (PCC).

13.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, INC, DEC, ROR, ROL, RORC, ROLC, PUSH, POP, DBNZ

2nd Operand / 1st Operand	#byte	A	r	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL+byte]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV[Note] XCH[Note] ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV											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											
[HL+byte]		MOV											

Note Except r = A.

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

2nd Operand 1st Operand	#word	AX	rp[Note]	saddrp	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	
rp	MOVW	MOVW[Note]				INCW DECW PUSH POP
saddrp		MOVW				
SP		MOVW				

Note Only when rp = BC, DE, or HL.

(3) Bit manipulation instructions

SET1, CLR1, NOT1, BT, BF

2nd Operand 1st Operand	$addr16	None
A.bit	BT BF	SET1 CLR1
sfr.bit	BT BF	SET1 CLR1
saddr.bit	BT BF	SET1 CLR1
PSW.bit	BT BF	SET1 CLR1
[HL].bit		SET1 CLR1
CY		SET1 CLR1 NOT1

(4) Call instructions/branch instructions

CALL, CALLT, BR, BC, BNC, BZ, BNZ, DBNZ

2nd Operand 1st Operand	AX	!addr16	[addr5]	$addr16
Basic Instructions	BR	CALL BR	CALLT	BR BC BNC BZ BNZ
Compound Instructions				DBNZ

(5) Other instructions

RET, RETI, NOP, EI, DI, HALT, STOP

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for development of systems using the μPD789014 Subseries. Figure A-1 shows development tools.

★ • Support to PC98-NX Series

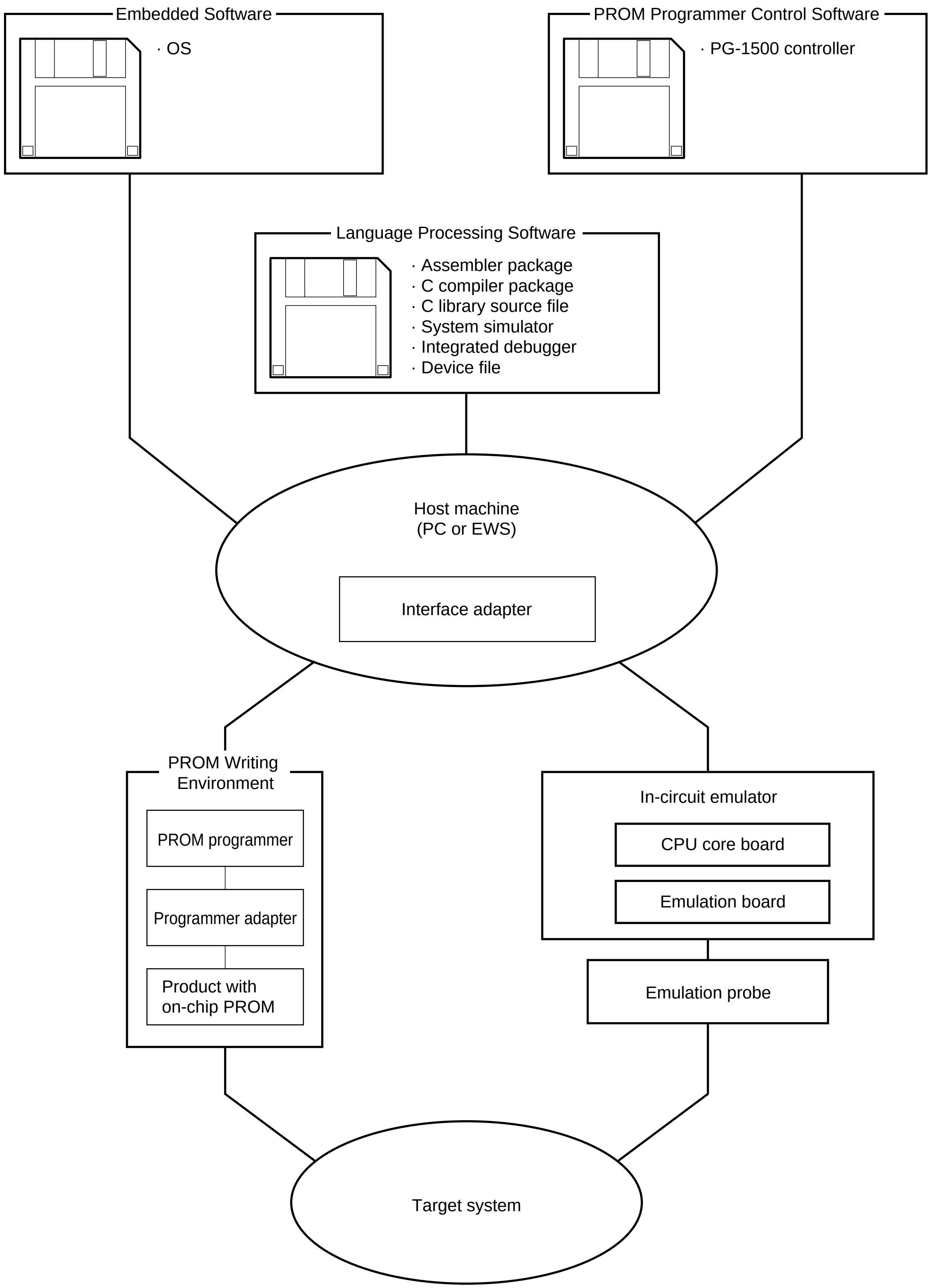
Unless specified otherwise, the products supported by IBM PC/AT™ compatibles can be used in PC98-NX Series. When using the PC98-NX Series, refer to the explanation of IBM PC/AT compatibles.

★ • Windows

Unless specified otherwise, "Windows" indicates the following operating systems.

- Windows 3.1
- Windows 95
- Windows NT™ Ver.4.0

Figure A-1. Development Tools



A.1 Language Processing Software

RA78K0S Assembler package	Program that converts program written in mnemonic into object code that can be executed by microcontroller. In addition, automatic functions to generate symbol table and optimize branch instructions are also provided. Used in combination with optional device file (DF789014). <Caution when used under PC environment> The assembler package is a DOS-based application but may be used under the Windows environment by using Project Manager of Windows (included in the assembler package).
	Part number: µS××××RA78K0S
CC78K0S C compiler package	Program that converts program written in C language into object codes that can be executed by microcontroller. Used in combination with optional assembler package and device file. <Caution when used under PC environment> The C compiler package is a DOS-based application but may be used under the Windows environment by using Project Manager of Windows (included in the assembler package).
	Part number: µS××××CC78K0S
DF789014[Note] Device file	File containing the information inherent to the device. Used in combination with other optional tools (RA78K0S, CC78K0S, SM78K0S). The corresponding operating system and host machine depend on the tools they are combined with.
	Part number: µS××××DF789014

Note DF789014 is a common file that can be used with RA78K0S, CC78K0S, and SM78K0S.

Remark ×××× in the part number differs depending on the host machines and operating systems to be used.

µS××××RA78K0S
µS××××CC78K0S
µS××××DF789014

××××	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows[Note]	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows[Note]	3.5" 2HC FD
BB13		English Windows[Note]	
3P16	HP9000 Series 700^{TM}	HP-UX^{TM} (Rel.10.10)	DAT (DDS)
3K13	SPARCstation^{TM}	SunOS^{TM} (Rel.4.1.4), Solaris (Rel.2.5.1)	3.5" 2HC FD
3K15			1/4" CGMT
3R13	NEWS^{TM} (RISC)	NEWS-OS^{TM} (Rel.6.1)	3.5" 2HC FD

Note Also operates under the DOS environmment.

A.2 PROM Writing Tools

A.2.1 Hardware

PG-1500 PROM programmer	PROM programmer that can program PROM-contained single-chip microcontrollers in stand-alone mode or through manipulation from host machine when connected to board supplied as accessory and optional PROM programmer adapter. Can program representative PROMs from 256K-bit to 4M-bit models.
PA-78P9014GT ★ PA-17K-DZ PROM programmer adapter	PROM programmer adapter for μPD78P9014 and connected to PG-1500.

A.2.2 Software

PG-1500 controller	Connects PG-1500 and host machine with serial and parallel interfaces and controls PG-1500 on host machine. The PG-1500 controller is a DOS-based application. Use this software in the DOS pane when running it on Windows.
	Part number: μS××××PG1500

Remark ×××× in the part number differs depending on the host machines and operating systems to be used.

μS××××PG1500

××××	Host Machine	OS	Supply Media
5A13	PC-9800 Series	MS-DOS^{TM} (Ver. 3.30 to Ver. 6.2[Note])	3.5" 2HD FD
5B13	IBM PC/AT compatibles	IBM DOS^{TM} (J5.02/V[Note])	3.5" 2HC FD

Note Although the MS-DOS or IBM DOS Ver. 5.0 or later provides a task swap function, the above software does not provide the task swap function.

A.3 Debugging Tools

A.3.1 Hardware

IE-78K0-NS In-circuit emulator	In-circuit emulator for debugging hardware and software of application system using 78K/0S Series. Supports integrated debugger (ID-78K0S-NS). Used in combination with AC adapter, emulation probe, and interface adapter for connecting the host machine.
IE-70000-MC-PS-B AC adapter	This is the adapter for supplying power from AC-100 V outlet.
IE-70000-98-IF-C Interface adapter	This adapter is needed when PC-9800 Series (excluding notebook type) is used as a host machine of IE-78K0S-NS (C bus supported).
★ IE-70000-CD-IF-A[Note 1] PC card interface	This PC card and interface cable are needed when a PC-9800 Series notebook-type personal computer is used as a host machine of IE-78K0S-NS (PCMCIA socket supported).
IE-70000-PC-IF-C Interface adapter	This adapter is needed when IBM PC/AT compatibles are used as a host machine of IE-78K0S-NS (ISA bus supported).
★ IE-70000-PCI-IF[Note 1] Interface adapter	This adapter is needed when a personal computer incorporating PCI bus is used as a host machine of IE-78K0S-NS.
IE-789014-NS-EM1 Emulation board	Emulation board for emulating the peripheral hardware inherent to the device. Used in combination with in-circuit emulator.
NP-28CT[Note 2] Emulation probe	Emulation probe for connecting the in-circuit emulator and target system. This is for 28-pin plastic shrink DIP.
NP-28GT[Note 2] Emulation probe	Emulation probe for connecting the in-circuit emulator and target system. This is for 28-pin plastic SOP.

Notes **1.** Under development

2. The NP-28CT and NP-28GT are the products of Naitou Densei Machidaseisakusho Co., Ltd. (+81-44-822-3813). Contact an NEC distributor regarding the purchase of these products.

A.3.2 Software

ID78K0S-NS Integrated debugger (Supports in-circuit emulator IE78K0S-NS)	Control program for debugging 78K/0S Series. This program provides a graphical user interface. It runs on Windows for personal computer users and on OSF/Motif^{TM} for engineering work station users, and has visual designs and operationability that comply with these operating systems. In addition, it has a powerful debug function that supports C language. Therefore, trace results can be displayed at a C language level by the window integration function that links source program, disassembled display, and memory display, to the trace result. This software also allows users to add other function extension modules such as task debugger and system performance analyzer to improve the debug efficiency for programs using a real-time operating system. Used in combination with optional device file.
	Part number: μS××××ID78K0S-NS

Remark ×××× in the part number differs depending on the host machines and operating system to be used.

μS××××ID78K0S-NS

××××	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows	3.5" 2HC FD
BB13		English Windows	

SM78K0S System simulator	Debugs program at C source level or assembler level while simulating operation of target system on host machine. SM78K0S runs on Windows. By using SM78K0S, the logic and performance of an application can be verified independently of hardware development even when the in-circuit emulator is not used. This enhances development efficiency and improves software quality. Used in combination with optional device file (DF789014).
	Part number: μS××××SM78K0S

Remark ×××× in the part number differs depending on the host machines and operating system to be used.

μS××××SM78K0S

××××	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows	3.5" 2HC FD
BB13		English Windows	

APPENDIX B   EMBEDDED SOFTWARE

The following embedded software is available for efficient program development and maintenance of the μPD789014 Subseries.

MX78K0S OS	MX78K0S is a subset OS that is based on the μITRON specification. Supplied with the MX78K0S nucleus. The MX78K0S OS controls tasks, events, and time. In task control, the MX78K0S OS controls task execution order, and then perform the switching process to a task to be executed. <Caution when used under PC environment> The MX78K0S is a DOS-based application. Use this software in the DOS pane when running it on Windows.

[MEMO]

APPENDIX C   REGISTER INDEX

C.1 Register Name Index (Alphabetic Order)

C.2 Register Symbol Index (Alphabetic Order)

[MEMO]

★

APPENDIX D REVISION HISTORY

Major revisions by edition and revised chapters are shown below.

Edition	Major Revisions from Previous Edition	Revised Chapters
2nd	“Under development” changed to “Developed” for the μPD789011, 789012, and 78P9014	Whole manual
	Addition of **Caution** for I/O port	CHAPTER 4 PORT FUNCTIONS
	Modification of maximum time required for switching CPU clock	CHAPTER 5 CLOCK GENERATOR
	Modification and addition of **Caution** to setting of watchdog timer mode register	CHAPTER 7 WATCHDOG TIMER
	Addition of **Caution** to setting of external interrupt mode register 0	CHAPTER 9 INTERRUPT FUNCTIONS
	Addition of description of interrupt request acceptance timing in maskable interrupt	
	Modification of timing chart of reset timing by $\overline{\text{RESET}}$ input	CHAPTER 11 RESET FUNCTION
	Modification of timing chart of reset timing by overflow in watchdog timer	
	Modification of timing chart of reset timing by $\overline{\text{RESET}}$ input in STOP mode	
3rd	Modification of recommended connection of unused pins	CHAPTER 2 PIN FUNCTIONS
	Modification of part of register name and symbol of special function register	CHAPTER 3 CPU ARCHITECTURE
	Addition of **Caution** to 8-bit compare register 0n	CHAPTER 6 8-BIT TIMER/EVENT COUNTER
	Modification of symbol and flag name of 8-bit timer mode control register 00	
	Modification of symbol and flag name of 8-bit timer mode control register 01	
	Modification of description of interval timer operation	
	Modification of description of external event counter operation	
	Modification of description of square wave output operation	
	Modification of symbol and flag name of serial operating mode register 00, and addition of **Caution** regarding transmit/receive operation	CHAPTER 8 SERIAL INTERFACE 00
	Modification of symbol and flag name of asynchronous serial interface mode register 00, and addition of **Caution** regarding transmit operation	
	Modification of description of 1-bit memory manipulation instruction enabled in asynchronous serial interface status register 00	
	Modification of symbol and flag name of asynchronous serial interface status register 00	
	Addition of description regarding transmit operation and description regarding read operation of RXB00 register in asynchronous serial interface (UART) mode	
	Addition of description regarding transmit/receive operation in 3-wire serial I/O mode	
	Modification of flag name of interrupt request flag register 0	CHAPTER 9 INTERRUPT FUNCTIONS
	Modification of flag name of interrupt mask flag register 0	
	Addition of **Caution** for replacing PROM version with mask ROM version	CHAPTER 12 μPD78P9014
	Addition of Solaris to operating system of language processing software	APPENDIX A DEVELOPMENT TOOLS
	Addition of PA-17K-DZ to PROM programmer adapter	
	Modification of PC card interface name, and addition of IE-70000-PCI-IF to interface adapter	

[MEMO]

Phase-out/Discontinued

NEC

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

Europe
NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

South America
NEC do Brasil S.A.
Fax: +55-11-6462-6829

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Korea
NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: 02-528-4411

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-2719-5951

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Japan
NEC Semiconductor Technical Hotline
Fax: 044-435-9608

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	❏	❏	❏	❏
Technical Accuracy	❏	❏	❏	❏
Organization	❏	❏	❏	❏