

# **$\mu$ PD789026 Subseries**

## **8-Bit Single-Chip Microcontrollers**

---

**$\mu$ PD789022**

**$\mu$ PD789024**

**$\mu$ PD789025**

**$\mu$ PD789026**

**$\mu$ PD78F9026A**

[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.

EEPROM and FIP are trademarks of NEC Electronics Corporation.

Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

PC/AT is a trademark of International Business Machines Corporation.

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

SPARCstation is a trademark of SPARC International, Inc.

Solaris and SunOS are trademarks of Sun Microsystems, Inc.

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

- **The information in this document is current as of September, 2002. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".  
The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.
  - "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  - "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).
  - "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

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

(Note)

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

# Regional Information

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

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

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

## **NEC Electronics America, Inc. (U.S.)**

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

## **NEC Electronics (Europe) GmbH**

Duesseldorf, Germany  
Tel: 0211-65 03 01  
Fax: 0211-65 03 327

### **• Sucursal en España**

Madrid, Spain  
Tel: 091-504 27 87  
Fax: 091-504 28 60

### **• Succursale Française**

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

### **• Filiale Italiana**

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

### **• Branch The Netherlands**

Eindhoven, The Netherlands  
Tel: 040-244 58 45  
Fax: 040-244 45 80

### **• Tyskland Filial**

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

### **• United Kingdom Branch**

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

## **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 Shanghai, Ltd.**

Shanghai, P.R. China  
Tel: 021-6841-1138  
Fax: 021-6841-1137

## **NEC Electronics Taiwan Ltd.**

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

## **NEC Electronics Singapore Pte. Ltd.**

Novena Square, Singapore  
Tel: 6253-8311  
Fax: 6250-3583

## Major Revisions in This Edition

Page	Description
Throughout	Deletion of the following packages <ul style="list-style-type: none"> <li>• 42-pin plastic shrink DIP (CU type)</li> <li>• 44-pin plastic QFP (GB-3BS-MTX type)</li> </ul>
p.33	Modification of pin handling of $V_{PP}$ pin
pp.92, 94	Modification of description in <b>6.4.1 Operation as timer interrupt</b> and <b>6.4.2 Operation as timer output</b>
p.92	Modification of <b>Caution</b> on rewriting CR20 in <b>6.4.1 Operation as timer interrupt</b>
pp.97, 98	Addition of <b>6.5 Notes on Using 16-Bit Timer 20</b>
p.122	Modification of description of PE00 flag in <b>Figure 9-5 Format of Asynchronous Serial Interface Status Register 00</b>
p.138	Addition of description on reading receive data of UART
p.148	Addition of <b>Caution</b> in <b>Figure 10-2 Format of Interrupt Request Flag Register</b>
pp.172 to 181	Overall revision of contents related to flash memory programming as <b>13.1 Flash Memory Characteristics</b>
pp.192 to 203	Addition of <b>CHAPTER 15 ELECTRICAL SPECIFICATIONS</b>
p.204	Addition of <b>CHAPTER 16 CHARACTERISTICS CURVES (MASK ROM VERSION)</b>
p.205	Addition of <b>CHAPTER 17 PACKAGE DRAWING</b>
p.206	Addition of <b>CHAPTER 18 RECOMMENDED SOLDERING CONDITIONS</b>
pp.207 to 213	Overall revision of <b>APPENDIX A DEVELOPMENT TOOLS</b> Deletion of embedded software
pp.214, 215	Addition of <b>APPENDIX B NOTES ON TARGET SYSTEM DESIGN</b>

The mark ★ shows the major revised points.

## INTRODUCTION

### Readers

This manual is intended for user engineers who wish to understand the functions of the  $\mu$ PD789026 Subseries to design and develop its application systems and programs.

The target subseries is the  $\mu$ PD789026 Subseries, which consists of the  $\mu$ PD789022,  $\mu$ PD789024,  $\mu$ PD789025,  $\mu$ PD789026, and  $\mu$ PD78F9026A.

### Purpose

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

### Organization

Two manuals are available for the  $\mu$ PD789026 Subseries: this manual and Instruction Manual (common to the 78K/0S Series).

$\mu$ PD789026 Subseries User's Manual	78K/0S Series User's Manual Instructions
<ul style="list-style-type: none"><li>• Pin functions</li><li>• Internal block functions</li><li>• Interrupts</li><li>• Other internal peripheral functions</li><li>• Electrical specifications</li></ul>	<ul style="list-style-type: none"><li>• CPU function</li><li>• Instruction set</li><li>• Instruction description</li></ul>

### How to Read This Manual

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

- ◇ To understand the overall functions of the  $\mu$ PD789026 Subseries
  - Read this manual in the order of the **CONTENTS**.
- ◇ How to read register formats
  - The name of a bit whose number is enclosed in <> 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
  - See **APPENDIX C REGISTER INDEX**.
- ◇ To learn details of the instruction functions of the 78K/0S Series
  - Refer to **78K/0S Series Instructions User's Manual (U11047E)** separately available.
- ◇ To learn the electrical specifications of the  $\mu$ PD789026 Subseries
  - Refer to **CHAPTER 15 ELECTRICAL SPECIFICATIONS**.

### 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)
<b>Note:</b>	Footnote for item marked with <b>Note</b> in the text
<b>Caution:</b>	Information requiring particular attention
<b>Remark:</b>	Supplementary information
Numerical representation:	Binary ... xxxx or xxxxB Decimal ... xxxx Hexadecimal ... xxxxH

**Related Documents**

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

**Documents Related to Devices**

Document Name	Document No.
μPD789026 Subseries User's Manual	This manual
78K/0S Series Instructions User's Manual	U11047E

**Documents Related to Development Software Tools (User's Manuals)**

Document Name	Document No.
RA78K0S Assembler Package	Operation
	Language
	Structured Assembly Language
CC78K0S C Compiler	Operation
	Language
SM78K Series System Simulator Ver. 2.30 or Later	Operation (Windows™ Based)
	External Part User Open Interface Specifications
ID78K Series Integrated Debugger Ver. 2.30 or Later	Operation (Windows Based)
Project Manager Ver. 3.12 or Later (Windows Based)	U14610E

**Documents Related to Development Hardware Tools (User's Manuals)**

Document Name	Document No.
IE-78K0S-NS In-Circuit Emulator	U13549E
IE-78K0S-NS-A In-Circuit Emulator	U15207E
IE-789026-NS-EM1 Emulation Board	U14362E

**Documents Related to Flash Memory Writing**

Document Name	Document No.
PG-FP3 Flash Memory Programmer User's Manual	U13502E
PG-FP4 Flash Memory Programmer User's Manual	U15260E

**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.
SEMICONDUCTOR SELECTION GUIDE - Products and Packages -	X13769X
Semiconductor Device Mounting Technology Manual	C10535E
Quality Grades on NEC Semiconductor Devices	C11531E
NEC Semiconductor Device Reliability/Quality Control System	C10983E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E

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

## CONTENTS

<b>CHAPTER 1 GENERAL</b>	<b>20</b>
1.1 Features	20
1.2 Applications	20
1.3 Ordering Information	21
1.4 Pin Configuration (Top View)	22
1.5 78K/0S Series Lineup	23
1.6 Block Diagram	26
1.7 Overview of Functions	27
<b>CHAPTER 2 PIN FUNCTIONS</b>	<b>28</b>
2.1 List of Pin Functions	28
2.2 Description of Pin Functions	30
2.2.1 P00 to P07 (Port 0)	30
2.2.2 P10 to P17 (Port 1)	30
2.2.3 P20 to P22 (Port 2)	30
2.2.4 P30 to P32 (Port 3)	31
2.2.5 P40 to P47 (Port 4)	31
2.2.6 P50 to P53 (Port 5)	32
2.2.7 $\overline{\text{RESET}}$	32
2.2.8 X1, X2	32
2.2.9 NC	32
2.2.10 $V_{DD}$	32
2.2.11 $V_{SS}$	32
2.2.12 $V_{PP}$ ( $\mu\text{PD78F9026A}$ only)	33
2.2.13 IC (mask ROM version only)	33
2.3 Pin I/O Circuits and Recommended Connection of Unused Pins	34
<b>CHAPTER 3 CPU ARCHITECTURE</b>	<b>36</b>
3.1 Memory Space	36
3.1.1 Internal program memory space	41
3.1.2 Internal data memory (internal high-speed RAM) space	42
3.1.3 Special function register (SFR) area	42
3.1.4 Data memory addressing	43
3.2 Processor Registers	48
3.2.1 Control registers	48
3.2.2 General-purpose registers	51
3.2.3 Special function registers (SFR)	52
3.3 Instruction Address Addressing	55
3.3.1 Relative addressing	55
3.3.2 Immediate addressing	56
3.3.3 Table indirect addressing	57
3.3.4 Register addressing	57

<b>3.4</b>	<b>Operand Address Addressing .....</b>	<b>58</b>
3.4.1	Direct addressing .....	58
3.4.2	Short direct addressing .....	59
3.4.3	Special function register (SFR) addressing.....	60
3.4.4	Register addressing .....	61
3.4.5	Register indirect addressing.....	62
3.4.6	Based addressing.....	63
3.4.7	Stack addressing.....	63
<b>CHAPTER 4</b>	<b>PORT FUNCTIONS .....</b>	<b>64</b>
<b>4.1</b>	<b>Functions of Ports.....</b>	<b>64</b>
<b>4.2</b>	<b>Port Configuration.....</b>	<b>66</b>
4.2.1	Port 0 .....	66
4.2.2	Port 1 .....	67
4.2.3	Port 2 .....	68
4.2.4	Port 3.....	71
4.2.5	Port 4.....	72
4.2.6	Port 5.....	73
<b>4.3</b>	<b>Registers Controlling Port Function .....</b>	<b>76</b>
<b>4.4</b>	<b>Operation of Port Functions.....</b>	<b>78</b>
4.4.1	Writing to I/O port .....	78
4.4.2	Reading from I/O port.....	78
4.4.3	Arithmetic operation of I/O port .....	78
<b>CHAPTER 5</b>	<b>CLOCK GENERATOR.....</b>	<b>79</b>
<b>5.1</b>	<b>Function of Clock Generator.....</b>	<b>79</b>
<b>5.2</b>	<b>Configuration of Clock Generator .....</b>	<b>79</b>
<b>5.3</b>	<b>Register Controlling Clock Generator.....</b>	<b>80</b>
<b>5.4</b>	<b>System Clock Oscillator .....</b>	<b>81</b>
5.4.1	System clock oscillator .....	81
5.4.2	Examples of incorrect resonator connection .....	82
5.4.3	Divider circuit.....	83
<b>5.5</b>	<b>Operation of Clock Generator .....</b>	<b>84</b>
<b>5.6</b>	<b>Changing Setting of System Clock and CPU Clock.....</b>	<b>85</b>
5.6.1	Time required for switching between system clock and CPU clock .....	85
5.6.2	Switching CPU clock .....	85
<b>CHAPTER 6</b>	<b>16-BIT TIMER 20 .....</b>	<b>86</b>
<b>6.1</b>	<b>Functions of 16-Bit Timer 20 .....</b>	<b>86</b>
<b>6.2</b>	<b>Configuration of 16-Bit Timer 20.....</b>	<b>87</b>
<b>6.3</b>	<b>Registers Controlling 16-Bit Timer 20.....</b>	<b>89</b>
<b>6.4</b>	<b>Operation of 16-Bit Timer 20 .....</b>	<b>92</b>
6.4.1	Operation as timer interrupt.....	92
6.4.2	Operation as timer output.....	94
6.4.3	Capture operation.....	95

6.4.4	16-bit timer counter 20 readout .....	96
★ 6.5	<b>Notes on Using 16-Bit Timer 20 .....</b>	<b>97</b>
6.5.1	Restrictions when rewriting 16-bit compare register 20 .....	97
<b>CHAPTER 7 8-BIT TIMER/EVENT COUNTER 00 .....</b>		<b>99</b>
7.1	<b>Functions of 8-Bit Timer/Event Counter 00 .....</b>	<b>99</b>
7.2	<b>Configuration of 8-Bit Timer/Event Counter 00.....</b>	<b>100</b>
7.3	<b>Registers Controlling 8-Bit Timer/Event Counter 00 .....</b>	<b>101</b>
7.4	<b>Operation of 8-Bit Timer/Event Counter 00 .....</b>	<b>103</b>
7.4.1	Operation as interval timer .....	103
7.4.2	Operation as external event counter .....	105
7.4.3	Operation as square wave output .....	106
7.5	<b>Notes on Using 8-Bit Timer/Event Counter 00 .....</b>	<b>108</b>
<b>CHAPTER 8 WATCHDOG TIMER.....</b>		<b>109</b>
8.1	<b>Functions of Watchdog Timer .....</b>	<b>109</b>
8.2	<b>Configuration of Watchdog Timer .....</b>	<b>110</b>
8.3	<b>Registers Controlling Watchdog Timer .....</b>	<b>111</b>
8.4	<b>Operation of Watchdog Timer.....</b>	<b>113</b>
8.4.1	Operation as watchdog timer .....	113
8.4.2	Operation as interval timer .....	114
<b>CHAPTER 9 SERIAL INTERFACE 00.....</b>		<b>115</b>
9.1	<b>Functions of Serial Interface 00.....</b>	<b>115</b>
9.2	<b>Configuration of Serial Interface 00 .....</b>	<b>115</b>
9.3	<b>Registers Controlling Serial Interface 00.....</b>	<b>119</b>
9.4	<b>Operation of Serial Interface 00 .....</b>	<b>126</b>
9.4.1	Operation stop mode.....	126
9.4.2	Asynchronous serial interface (UART) mode .....	128
9.4.3	3-wire serial I/O mode .....	140
<b>CHAPTER 10 INTERRUPT FUNCTIONS.....</b>		<b>144</b>
10.1	<b>Interrupt Function Types.....</b>	<b>144</b>
10.2	<b>Interrupt Sources and Configuration .....</b>	<b>144</b>
10.3	<b>Registers Controlling Interrupt Function.....</b>	<b>147</b>
10.4	<b>Interrupt Servicing Operation .....</b>	<b>153</b>
10.4.1	Non-maskable interrupt request acknowledgement operation .....	153
10.4.2	Maskable interrupt request acknowledgement operation.....	155
10.4.3	Nesting processing.....	157
10.4.4	Interrupt request hold .....	159
<b>CHAPTER 11 STANDBY FUNCTION .....</b>		<b>160</b>
11.1	<b>Standby Function and Configuration.....</b>	<b>160</b>

	11.1.1	Standby function.....	160
	11.1.2	Standby function control register.....	161
	<b>11.2</b>	<b>Operation of Standby Function .....</b>	<b>162</b>
	11.2.1	HALT mode .....	162
	11.2.2	STOP mode.....	165
	<b>CHAPTER 12</b>	<b>RESET FUNCTION.....</b>	<b>168</b>
	<b>CHAPTER 13</b>	<b>μPD78F9026A.....</b>	<b>171</b>
★	<b>13.1</b>	<b>Flash Memory Characteristics .....</b>	<b>172</b>
	13.1.1	Programming environment .....	172
	13.1.2	Communication mode.....	173
	13.1.3	On-board pin processing .....	176
	13.1.4	Connection of adapter for flash writing.....	179
	<b>CHAPTER 14</b>	<b>INSTRUCTION SET.....</b>	<b>182</b>
	<b>14.1</b>	<b>Operation .....</b>	<b>182</b>
	14.1.1	Operand identifiers and writing methods.....	182
	14.1.2	Description of “Operation” column.....	183
	14.1.3	Description of “Flag” column .....	183
	<b>14.2</b>	<b>Operation List .....</b>	<b>184</b>
	<b>14.3</b>	<b>Instructions Listed by Addressing Type.....</b>	<b>189</b>
★	<b>CHAPTER 15</b>	<b>ELECTRICAL SPECIFICATIONS .....</b>	<b>192</b>
★	<b>CHAPTER 16</b>	<b>CHARACTERISTICS CURVES (MASK ROM VERSION) .....</b>	<b>204</b>
★	<b>CHAPTER 17</b>	<b>PACKAGE DRAWING.....</b>	<b>205</b>
★	<b>CHAPTER 18</b>	<b>RECOMMENDED SOLDERING CONDITIONS .....</b>	<b>206</b>
	<b>APPENDIX A</b>	<b>DEVELOPMENT TOOLS .....</b>	<b>207</b>
★	<b>A.1</b>	<b>Software Package.....</b>	<b>209</b>
	<b>A.2</b>	<b>Language Processing Software.....</b>	<b>209</b>
★	<b>A.3</b>	<b>Control Software .....</b>	<b>210</b>
	<b>A.4</b>	<b>Flash Memory Writing Tools .....</b>	<b>210</b>
	<b>A.5</b>	<b>Debugging Tools (Hardware) .....</b>	<b>211</b>
	<b>A.6</b>	<b>Debugging Tools (Software) .....</b>	<b>212</b>
	<b>A.7</b>	<b>Conversion Adapter (TGB-044SAP) Drawing .....</b>	<b>213</b>
★	<b>APPENDIX B</b>	<b>NOTES ON TARGET SYSTEM DESIGN.....</b>	<b>214</b>

**APPENDIX C REGISTER INDEX .....216**

**C.1 Register Name Index.....216**

**C.2 Register Symbol Index.....218**

**APPENDIX D REVISION HISTORY.....220**

## LIST OF FIGURES (1/3)

Figure No.	Title	Page
2-1	Pin I/O Circuits .....	35
3-1	Memory Map ( $\mu$ PD789022) .....	36
3-2	Memory Map ( $\mu$ PD789024) .....	37
3-3	Memory Map ( $\mu$ PD789025) .....	38
3-4	Memory Map ( $\mu$ PD789026) .....	39
3-5	Memory Map ( $\mu$ PD78F9026A) .....	40
3-6	Data Memory Addressing ( $\mu$ PD789022) .....	43
3-7	Data Memory Addressing ( $\mu$ PD789024) .....	44
3-8	Data Memory Addressing ( $\mu$ PD789025) .....	45
3-9	Data Memory Addressing ( $\mu$ PD789026) .....	46
3-10	Data Memory Addressing ( $\mu$ PD78F9026A) .....	47
3-11	Program Counter Configuration .....	48
3-12	Program Status Word Configuration .....	48
3-13	Stack Pointer Configuration .....	50
3-14	Data to Be Saved to Stack Memory .....	50
3-15	Data to Be Restored from Stack Memory .....	50
3-16	General-Purpose Register Configuration .....	51
4-1	Port Types .....	64
4-2	Block Diagram of P00 to P07 .....	66
4-3	Block Diagram of P10 to P17 .....	67
4-4	Block Diagram of P20 .....	68
4-5	Block Diagram of P21 .....	69
4-6	Block Diagram of P22 .....	70
4-7	Block Diagram of P30 to P32 .....	71
4-8	Block Diagram of P40 to P47 .....	72
4-9	Block Diagram of P50 .....	73
4-10	Block Diagram of P51 .....	74
4-11	Block Diagram of P52 and P53 .....	75
4-12	Format of Port Mode Register .....	77
4-13	Format of Pull-Up Resistor Option Register .....	77
5-1	Block Diagram of Clock Generator .....	79
5-2	Format of Processor Clock Control Register .....	80
5-3	External Circuit of System Clock Oscillator .....	81
5-4	Examples of Incorrect Resonator Connection .....	82
5-5	Switching CPU Clock .....	85
6-1	Block Diagram of 16-Bit Timer 20 .....	87

## LIST OF FIGURES (2/3)

Figure No.	Title	Page
6-2	Format of 16-Bit Timer Mode Control Register 20 .....	90
6-3	Format of Port Mode Register 5.....	91
6-4	Settings of 16-Bit Timer Mode Control Register 20 for Timer Interrupt Operation .....	92
6-5	Timer Interrupt Operation Timing .....	93
6-6	Settings of 16-Bit Timer Mode Control Register 20 for Timer Output Operation.....	94
6-7	Timer Output Timing .....	94
6-8	Settings of 16-Bit Timer Mode Control Register 20 for Capture Operation.....	95
6-9	Capture Operation Timing (Both Edges of CPT2 Pin Are Specified) .....	95
6-10	Readout Timing of 16-Bit Timer Counter 20 .....	96
7-1	Block Diagram of 8-Bit Timer/Event Counter 00 .....	100
7-2	Format of 8-Bit Timer Mode Control Register 00 .....	101
7-3	Format of Port Mode Register 5.....	102
7-4	Interval Timer Operation Timing .....	104
7-5	External Event Counter Operation Timing (with Rising Edge Specified) .....	105
7-6	Square Wave Output Timing.....	107
7-7	Start Timing of 8-Bit Timer Counter 00 .....	108
7-8	One Pulse Count Operation Timing .....	108
8-1	Block Diagram of Watchdog Timer .....	110
8-2	Format of Timer Clock Select Register 2 .....	111
8-3	Format of Watchdog Timer Mode Register .....	112
9-1	Block Diagram of Serial Interface 00 .....	116
9-2	Block Diagram of Baud Rate Generator .....	117
9-3	Format of Serial Operation Mode Register 00 .....	119
9-4	Format of Asynchronous Serial Interface Mode Register 00 .....	120
9-5	Format of Asynchronous Serial Interface Status Register 00 .....	122
9-6	Format of Baud Rate Generator Control Register 00.....	123
9-7	Format of Asynchronous Serial Interface Transmit/Receive Data .....	133
9-8	Asynchronous Serial Interface Transmission Completion Interrupt Timing .....	135
9-9	Asynchronous Serial Interface Reception Completion Interrupt Timing .....	136
9-10	Receive Error Timing .....	137
9-11	3-Wire Serial I/O Mode Timing.....	143
10-1	Basic Configuration of Interrupt Function.....	146
10-2	Format of Interrupt Request Flag Register .....	148
10-3	Format of Interrupt Mask Flag Register .....	149
10-4	Format of External Interrupt Mode Register 0.....	150
10-5	Configuration of Program Status Word .....	151



## LIST OF FIGURES (3/3)

Figure No.	Title	Page
10-6	Format of Key Return Mode Register 00 .....	152
10-7	Block Diagram of Falling Edge Detector .....	152
10-8	Flowchart from Non-Maskable Interrupt Request Generation to Acknowledgement .....	154
10-9	Non-Maskable Interrupt Request Acknowledgement Timing .....	154
10-10	Acknowledgement of Non-Maskable Interrupt Request .....	154
10-11	Interrupt Request Acknowledgement Processing Algorithm .....	156
10-12	Interrupt Request Acknowledgement Timing (Example of MOV A, r) .....	157
10-13	Interrupt Request Acknowledgement Timing (When Interrupt Request Flag Is Set at Last Clock During Instruction Execution) .....	157
10-14	Example of Nesting .....	158
11-1	Format of Oscillation Stabilization Time Select Register .....	161
11-2	Releasing HALT Mode by Interrupt .....	163
11-3	Releasing HALT Mode by $\overline{\text{RESET}}$ Input .....	164
11-4	Releasing STOP Mode by Interrupt .....	166
11-5	Releasing STOP Mode by $\overline{\text{RESET}}$ Input .....	167
12-1	Block Diagram of Reset Function .....	168
12-2	Reset Timing by $\overline{\text{RESET}}$ Input .....	169
12-3	Reset Timing by Overflow in Watchdog Timer .....	169
12-4	Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode .....	169
13-1	Environment for Writing Program to Flash Memory .....	172
13-2	Communication Mode Selection Format .....	173
13-3	Example of Connection with Dedicated Flash Programmer .....	174
13-4	V <sub>PP</sub> Pin Connection Example .....	176
13-5	Signal Conflict (Input Pin of Serial Interface) .....	177
13-6	Abnormal Operation of Other Device .....	177
13-7	Signal Conflict ( $\overline{\text{RESET}}$ Pin) .....	178
13-8	Wiring Example for Flash Writing Adapter with 3-Wire Serial I/O .....	179
13-9	Wiring Example for Flash Writing Adapter with UART .....	180
13-10	Wiring Example for Flash Writing Adapter with Pseudo 3-Wire (When P0 Is Used) .....	181
A-1	Development Tools .....	208
A-2	TGB-044SAP Package Drawing (Reference) .....	213
B-1	Distance Between In-Circuit Emulator and Conversion Adapter .....	214
B-2	Connection Condition of Target System (NP-H44GB-TQ) .....	215

## LIST OF TABLES (1/2)

Table No.	Title	Page
2-1	Types of Pin I/O Circuits and Recommended Connection of Unused Pins .....	34
3-1	Internal ROM Capacity .....	41
3-2	Vector Table .....	41
3-3	Internal High-Speed RAM Capacity .....	42
3-4	Special Function Registers .....	53
4-1	Port Functions .....	65
4-2	Port Configuration .....	66
4-3	Port Mode Register and Output Latch Settings When Using Alternate Functions .....	76
5-1	Configuration of Clock Generator .....	79
5-2	Maximum Time Required for Switching CPU Clock .....	85
6-1	Configuration of 16-Bit Timer 20 .....	87
6-2	Interval Time of 16-Bit Timer 20 .....	92
6-3	Settings of Capture Edge .....	95
7-1	Interval Time of 8-Bit Timer/Event Counter 00 .....	99
7-2	Square Wave Output Range of 8-Bit Timer/Event Counter 00 .....	99
7-3	Configuration of 8-Bit Timer/Event Counter 00 .....	100
7-4	Interval Time of 8-Bit Timer/Event Counter 00 .....	103
7-5	Square Wave Output Range of 8-Bit Timer/Event Counter 00 .....	106
8-1	Inadvertent Loop Detection Time of Watchdog Timer .....	109
8-2	Interval Time .....	109
8-3	Configuration of Watchdog Timer .....	110
8-4	Inadvertent Loop Detection Time of Watchdog Timer .....	113
8-5	Interval Time of Interval Timer .....	114
9-1	Configuration of Serial Interface 00 .....	115
9-2	Operating Mode Settings of Serial Interface 00 .....	121
9-3	Example of Relationship Between System Clock and Baud Rate .....	124
9-4	Relationship Between ASCK Pin Input Frequency and Baud Rate (When BRGC00 Is Set to 80H) .....	125
9-5	Example of Relationship Between System Clock and Baud Rate .....	132
9-6	Relationship Between ASCK Pin Input Frequency and Baud Rate (When BRGC00 Is Set to 80H) .....	132
9-7	Receive Error Causes .....	137
10-1	Interrupt Source List .....	145
10-2	Flags Corresponding to Interrupt Request Signal Names .....	147

## LIST OF TABLES (2/2)

Table No.	Title	Page
10-3	Time from Generation of Maskable Interrupt Request to Servicing .....	155
11-1	HALT Mode Operating Status .....	162
11-2	Operation After Release of HALT Mode .....	164
11-3	STOP Mode Operating Status .....	165
11-4	Operation After Release of STOP Mode .....	167
12-1	Hardware Status After Reset .....	170
13-1	Differences Between $\mu$ PD78F9026A and Mask ROM Versions .....	171
13-2	Communication Mode List.....	173
13-3	Pin Connection List .....	175
14-1	Operand Identifiers and Writing Methods .....	182
18-1	Surface Mounting Type Soldering Conditions.....	206

## CHAPTER 1 GENERAL

### 1.1 Features

- ROM and RAM capacity

Part Number \ Item	Program Memory		Data Memory
$\mu$ PD789022	ROM	4 KB	256 bytes
$\mu$ PD789024		8 KB	
$\mu$ PD789025		12 KB	512 bytes
$\mu$ PD789026		16 KB	
$\mu$ PD78F9026A	Flash memory	16 KB	

- Minimum instruction execution time can be changed from high-speed (0.4  $\mu$ s) to low-speed (1.6  $\mu$ s) at 5.0 MHz operation with system clock
- I/O ports: 34 lines
- Serial interface: 1 channel  
3-wire serial I/O mode/UART mode selectable
- Timer: 3 channels
  - 16-bit timer: 1 channel
  - 8-bit timer/event counter: 1 channel
  - Watchdog timer: 1 channel
- Vectored interrupts: 10
- Supply voltage:  $V_{DD} = 1.8$  to 5.5 V
- Operating ambient temperature:  $T_A = -40$  to  $+85^{\circ}\text{C}$

### 1.2 Applications

Home appliances, car accessories, air conditioners, game machines, etc.

★ **1.3 Ordering Information**

Part Number	Package	Internal ROM
$\mu$ PD789022GB-xxx-8ES	44-pin plastic LQFP (10 × 10)	Mask ROM
$\mu$ PD789024GB-xxx-8ES	44-pin plastic LQFP (10 × 10)	Mask ROM
$\mu$ PD789025GB-xxx-8ES	44-pin plastic LQFP (10 × 10)	Mask ROM
$\mu$ PD789026GB-xxx-8ES	44-pin plastic LQFP (10 × 10)	Mask ROM
$\mu$ PD78F9026AGB-8ES	44-pin plastic LQFP (10 × 10)	Flash memory

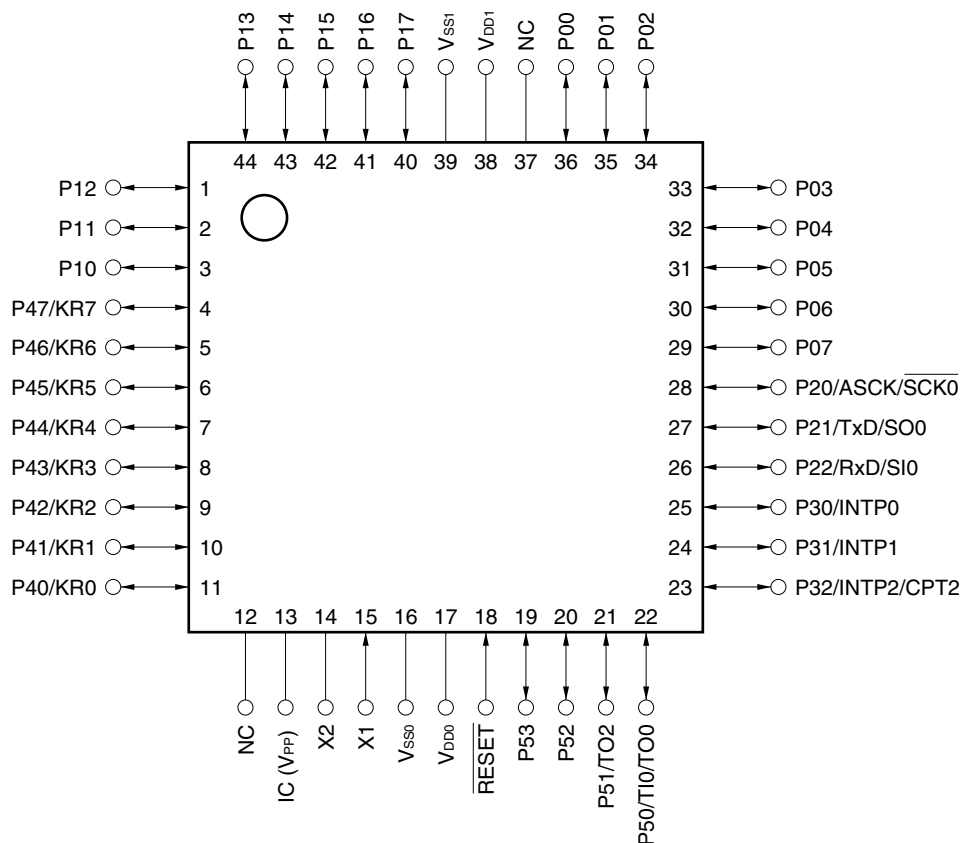
**Remark** xxx indicates ROM code suffix.

## 1.4 Pin Configuration (Top View)

### • 44-pin plastic LQFP (10 × 10)

μPD789022GB-xxx-8ES   μPD789024GB-xxx-8ES   μPD789025GB-xxx-8ES

μPD789026GB-xxx-8ES   μPD78F9026AGB-8ES



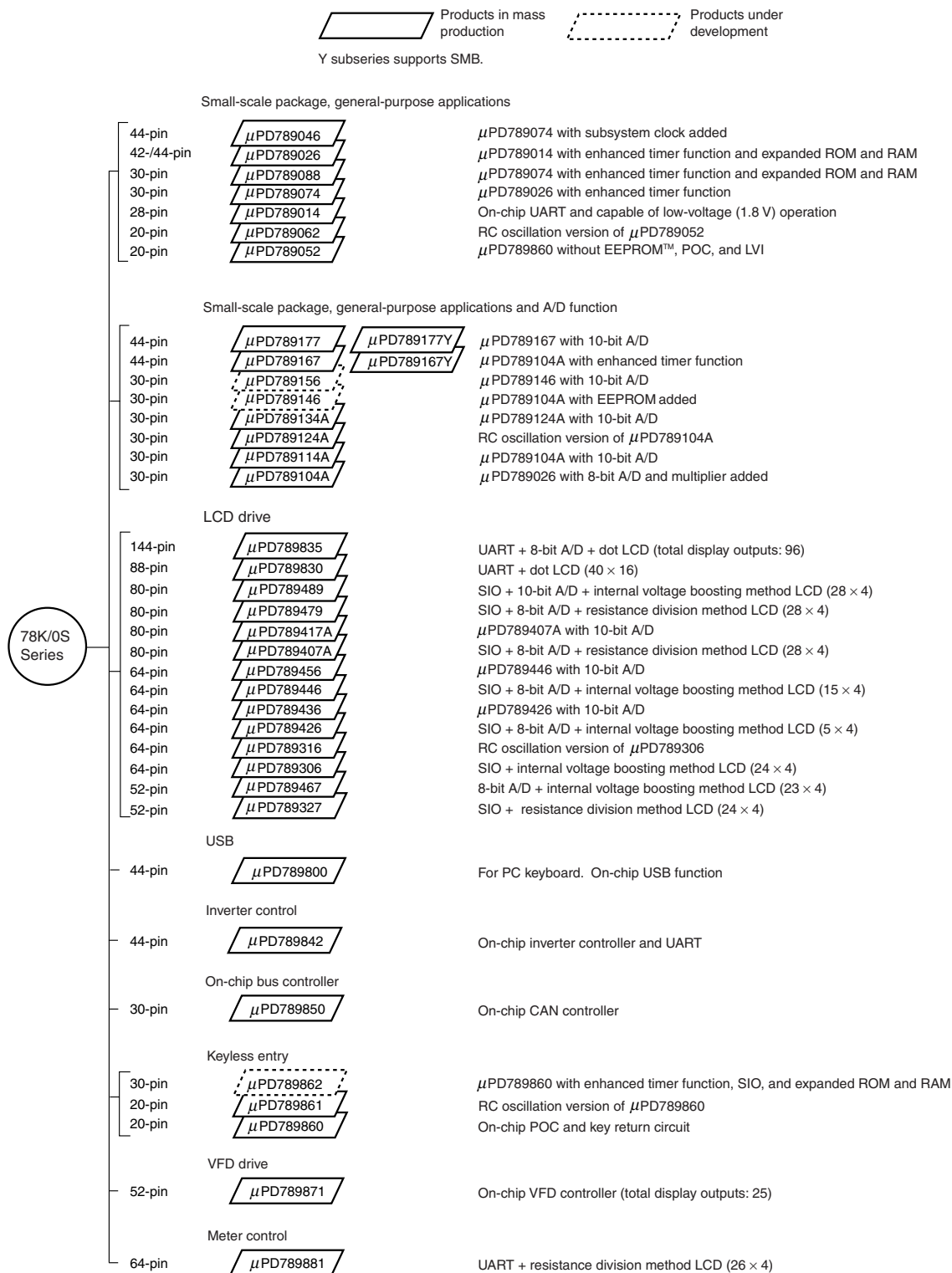
**Caution** Connect the IC pin directly to V<sub>SS0</sub> or V<sub>SS1</sub>.

**Remark** The item in parentheses applies to the μPD78F9026A only.

ASCK:	Asynchronous serial clock	RESET:	Reset
CPT2:	Capture trigger input	RxD:	Receive data
IC:	Internally connected	SCK0:	Serial clock
INTP0 to INTP2:	Interrupt from peripherals	SI0:	Serial input
KR0 to KR7:	Key return	SO0:	Serial output
NC:	Non-connection	TI0:	Timer input
P00 to P07:	Port 0	TO0, TO2:	Timer output
P10 to P17:	Port 1	TxD:	Transmit data
P20 to P22:	Port 2	V <sub>DD0</sub> , V <sub>DD1</sub> :	Power supply
P30 to P32:	Port 3	V <sub>PP</sub> :	Programming power supply
P40 to P47:	Port 4	V <sub>SS0</sub> , V <sub>SS1</sub> :	Ground
P50 to P53:	Port 5	X1, X2:	Crystal

## ★ 1.5 78K/0S Series Lineup

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



**Remark** VFD (Vacuum Fluorescent Display) is referred to as FIP™ (Fluorescent Indicator Panel) in some documents, but the functions of the two are the same.

The major functional differences between the subseries are listed below.

**Series for general-purpose applications and LCD drive**

Function Subseries		ROM Capacity (Bytes)	Timer				8-Bit A/D	10-Bit A/D	Serial Interface	I/O	V <sub>DD</sub>	Remarks
			8-Bit	16-Bit	Watch	WDT					MIN. Value	
Small-scale package, general-purpose applications	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			–							
	μPD789088	16 K to 32 K	3 ch							24		
	μPD789074	2 K to 8 K	1 ch									
	μPD789014	2 K to 4 K	2 ch	–						22		
	μPD789062	4 K							–	14		RC-oscillation version
	μPD789052											–
Small-scale package, general-purpose applications + A/D converter	μPD789177	16 K to 24 K	3 ch	1 ch	1 ch	1 ch	–	8 ch	1 ch (UART: 1 ch)	31	1.8 V	–
	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-oscillation version
	μPD789124A						4 ch	–				
	μPD789114A						–	4 ch				–
	μPD789104A						4 ch	–				
LCD drive	μPD789835	24 K to 60 K	6 ch	–	1 ch	1 ch	3 ch	–	1 ch (UART: 1 ch)	37	1.8 V <sup>Note</sup>	Dot LCD supported
	μPD789830	24 K	1 ch	1 ch			–			30	2.7 V	
	μPD789489	32 K to 48 K	3 ch					8 ch	2 ch (UART: 1 ch)	45	1.8 V	–
	μPD789479	24 K to 48 K					8 ch	–				
	μPD789417A	12 K to 24 K					–	7 ch	1 ch (UART: 1 ch)	43		
	μPD789407A						7 ch	–				
	μPD789456	12 K to 16 K	2 ch				–	6 ch		30		
	μPD789446						6 ch	–				
	μPD789436						–	6 ch		40		
	μPD789426						6 ch	–				
	μPD789316	8 K to 16 K					–		2 ch (UART: 1 ch)	23		RC-oscillation version
	μPD789306											–
	μPD789467	4 K to 24 K		–			1 ch		–	18		
	μPD789327						–		1 ch	21		

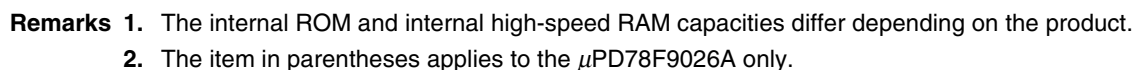
**Note** Flash memory version: 3.0 V



## Series for ASSP

Subseries	Function	ROM Capacity (Bytes)	Timer				8-Bit A/D	10-Bit A/D	Serial Interface	I/O	V <sub>DD</sub>	Remarks
			8-Bit	16-Bit	Watch	WDT					MIN. Value	
USB	μPD789800	8 K	2 ch	–	–	1 ch	–	–	2 ch (USB: 1 ch)	31	4.0 V	–
Inverter control	μPD789842	8 K to 16 K	3 ch	<b>Note 1</b>	1 ch	1 ch	8 ch	–	1 ch (UART: 1 ch)	30	4.0 V	–
On-chip bus controller	μPD789850	16 K	1 ch	1 ch	–	1 ch	4 ch	–	2 ch (UART: 1 ch)	18	4.0 V	–
Keyless entry	μPD789861	4 K	2 ch	–	–	1 ch	–	–	–	14	1.8 V	RC-oscillation version, on-chip EEPROM
	μPD789860											On-chip EEPROM
	μPD789862	16 K	1 ch	2 ch					1 ch (UART: 1 ch)	22		
VFD drive	μPD789871	4 K to 8 K	3 ch	–	1 ch	1 ch	–	–	1 ch	33	2.7 V	–
Meter control	μPD789881	16 K	2 ch	1 ch	–	1 ch	–	–	1 ch (UART: 1 ch)	28	2.7 V <sup>Note 2</sup>	–

**Notes** 1. 10-bit timer: 1 channel  
2. Flash memory version: 3.0 V



## 1.7 Overview of Functions

Part Number		μPD789022	μPD789024	μPD789025	μPD789026	μPD78F9026A
Item						
Internal memory	ROM	Mask ROM				Flash memory
		4 KB	8 KB	12 KB	16 KB	16 KB
	High-speed RAM	256 bytes		512 bytes		
Minimum instruction execution time		0.4/1.6 μs (when operated at 5.0 MHz with system clock)				
Instruction set		<ul style="list-style-type: none"><li>16-bit operations</li><li>Bit manipulation (set, reset, test), etc.</li></ul>				
I/O ports		<div>Total: 34</div> <ul style="list-style-type: none"><li>CMOS I/O: 34</li></ul>				
Serial interface		<ul style="list-style-type: none"><li>3-wire serial I/O mode/UART mode selectable: 1 channel</li></ul>				
Timer		<ul style="list-style-type: none"><li>16-bit timer: 1 channel</li><li>8-bit timer/event counter: 1 channel</li><li>Watchdog timer: 1 channel</li></ul>				
Timer outputs		2				
Vectored interrupt sources	Maskable	Internal: 5, External: 4				
	Non-maskable	Internal: 1				
Power supply voltage		V <sub>DD</sub> = 1.8 to 5.5 V				
Operating ambient temperature		T <sub>A</sub> = −40 to +85°C				
Package		44-pin plastic LQFP (10 × 10)				

The outline of the timer is as follows.

		16-Bit Timer 20	8-Bit Timer/Event Counter 00	Watchdog Timer
Operating mode	Interval timer	–	1 channel	1 channel <sup>Note</sup>
	External event counter	–	1 channel	–
Function	Timer output	1 output	1 output	–
	Capture	1 input	–	–
	Interrupt source	1	1	2

**Note** The watchdog timer has watchdog timer and interval timer functions. Select one of them.

## CHAPTER 2 PIN FUNCTIONS

### 2.1 List of Pin Functions

#### (1) Port pins

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P07	I/O	Port 0 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	—
P10 to P17	I/O	Port 1 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	—
P20	I/O	Port 2 3-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	SCK0/ASCK
P21				SO0/TxD
P22				SI0/RxD
P30	I/O	Port 3 3-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	INTP0
P31				INTP1
P32				INTP2/CPT2
P40 to P47	I/O	Port 4 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	KR0 to KR7
P50	I/O	Port 5 4-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	T10/TO0
P51				TO2
P52, P53				—

## (2) Non-port pins

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0	Input	External interrupt request input for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified	Input	P30
INTP1				P31
INTP2				P32/CPT2
KR0 to KR7	Input	Key return signal detection	Input	P40 to P47
SI0	Input	3-wire serial interface serial data input	Input	P22/RxD
SO0	Output	3-wire serial interface serial data output	Input	P21/TxD
SCK0	I/O	3-wire serial interface serial clock input/output	Input	P20/ASCK
ASCK	Input	Asynchronous serial interface serial clock input	Input	P20/SCK0
RxD	Input	Asynchronous serial interface serial data input	Input	P22/SI0
TxD	Output	Asynchronous serial interface serial data output	Input	P21/SO0
TO2	Output	16-bit timer 20 output	Input	P51
CPT2	Input	16-bit timer capture edge input	Input	P32/INTP2
TI0	Input	External count clock input to 8-bit timer/event counter 00	Input	P50/TO0
TO0	Output	8-bit timer/event counter 00 output	Input	P50/TI0
X1	Input	Connection of crystal for system clock oscillation	—	—
X2	—		—	—
RESET	Input	System reset input	Input	—
NC	—	Not connected internally. Connect this pin to the V <sub>SS0</sub> or V <sub>SS1</sub> pin (it can also be left open).	—	—
V <sub>DD0</sub>	—	Positive power supply for ports	—	—
V <sub>DD1</sub>	—	Positive power supply (except for ports)	—	—
V <sub>SS0</sub>	—	Ground potential for ports	—	—
V <sub>SS1</sub>	—	Ground potential (except for ports)	—	—
IC	—	Internally connected. Connect this pin directly to the V <sub>SS0</sub> or V <sub>SS1</sub> pin.	—	—
V <sub>PP</sub>	—	Flash memory programming mode setting. Apply high voltage during program write/verify.	—	—

## 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 to 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 by setting the pull-up resistor option register (PUO).

This port can drive LEDs directly.

### 2.2.2 P10 to P17 (Port 1)

These pins constitute an 8-bit I/O port and can be set to 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 by setting the pull-up resistor option register (PUO).

This port can drive LEDs directly.

### 2.2.3 P20 to P22 (Port 2)

These pins constitute a 3-bit I/O port. In addition, they also function as the data and clock I/O of the serial interface.

This port can drive LEDs 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. Port 2 can be set to input or output port mode in 1-bit units by using port mode register 2 (PM2). When the port is used as an input port, an on-chip pull-up resistor can be used by setting the pull-up resistor option register (PUO).

#### (2) Control mode

In this mode, port 2 functions as the data I/O and the clock I/O of the serial interface.

##### (a) SIO, 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

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

##### (d) ASCK

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

**Caution** When using port 2 as serial interface pins, the I/O mode and output latch must be set according to the function to be used. For details of the setting, see Table 9-2 Operating Mode Settings of Serial Interface 00.

### 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 and capture edge inputs.

This port can drive LEDs 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. Port 3 can be set to input or output port mode in 1-bit units by using port mode register 3 (PM3). When the port is used as an input port, an on-chip pull-up resistor can be used by setting the pull-up resistor option register (PUO).

#### (2) Control mode

In this mode, port 3 functions as the external interrupt input pins.

##### (a) INTP0 to INTP2

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

##### (b) CPT2

This is the capture edge input pin.

### 2.2.5 P40 to P47 (Port 4)

These pins constitute an 8-bit I/O port. In addition, they also function as key return signal detection pins.

This port can drive LEDs directly.

Port 4 can be specified in the following operation modes in 1-bit units.

#### (1) Port mode

In this mode, port 4 functions as an 8-bit I/O port. Port 4 can be set to input or output port mode in 1-bit units by using port mode register 4 (PM4). When the port is used as an input port, an on-chip pull-up resistor can be used by setting the pull-up resistor option register (PUO).

#### (2) Control mode

In this mode, port 4 functions as the key return signal detection pins (KR0 to KR7).

### 2.2.6 P50 to P53 (Port 5)

These pins constitute a 4-bit I/O port. In addition, they also function as timer I/O pins.

This port can drive LEDs directly.

Port 5 can be specified in the following operation modes in 1-bit units.

#### (1) Port mode

In this mode, port 5 functions as a 4-bit I/O port. Port 5 can be set to input or output port mode in 1-bit units by using port mode register 5 (PM5). When the port is used as an input port, an on-chip pull-up resistor can be used by setting the pull-up resistor option register (PUO).

#### (2) Control mode

In this mode, port 5 functions as the timer I/O.

##### (a) T10

This is the external clock input pin for 8-bit timer/event counter 00.

##### (b) T00

This is the 8-bit timer/event counter 00 output pin.

##### (c) T02

This is the 16-bit timer 20 output pin.

### 2.2.7 $\overline{\text{RESET}}$

This pin inputs an active-low system reset signal.

### 2.2.8 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.

### 2.2.9 NC

The NC (non-connection) pin is not connected internally. Connect this pin to the  $V_{SS0}$  or  $V_{SS1}$  pin (it can also be left open).

### 2.2.10 $V_{DD}$

This is the positive power supply pin.

### 2.2.11 $V_{SS}$

This is the ground potential pin.



**2.2.12 V<sub>PP</sub> ( $\mu$ PD78F9026A only)**

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

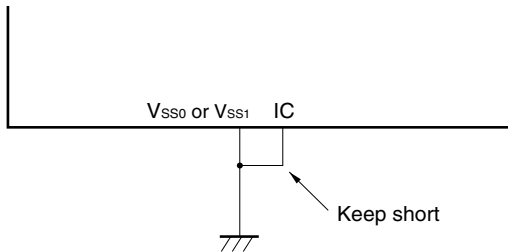
- ★ Handle the pins in either of the following ways.
- Independently connect a 10 k $\Omega$  pull-down resistor.
  - Switch this pin to be directly connected to the dedicated flash programmer in programming mode or to V<sub>SS0</sub> or V<sub>SS1</sub> in normal operation mode using a jumper on the board.

**2.2.13 IC (mask ROM version only)**

The IC (internally connected) pin is used to set the  $\mu$ PD789026 Subseries in the test mode for testing before shipment. In the normal operating mode, directly connect the IC pin to the V<sub>SS0</sub> or V<sub>SS1</sub> pin with as short a wire as possible.

If a potential difference is generated between the IC pin and V<sub>SS0</sub> or V<sub>SS1</sub> pin due to a long wiring length between these pins, or external noise is superimposed on the IC pin, the user program may not run correctly.

- Connect the IC pin directly to the V<sub>SS0</sub> or V<sub>SS1</sub> pin.



### 2.3 Pin I/O Circuits and Recommended Connection of Unused Pins

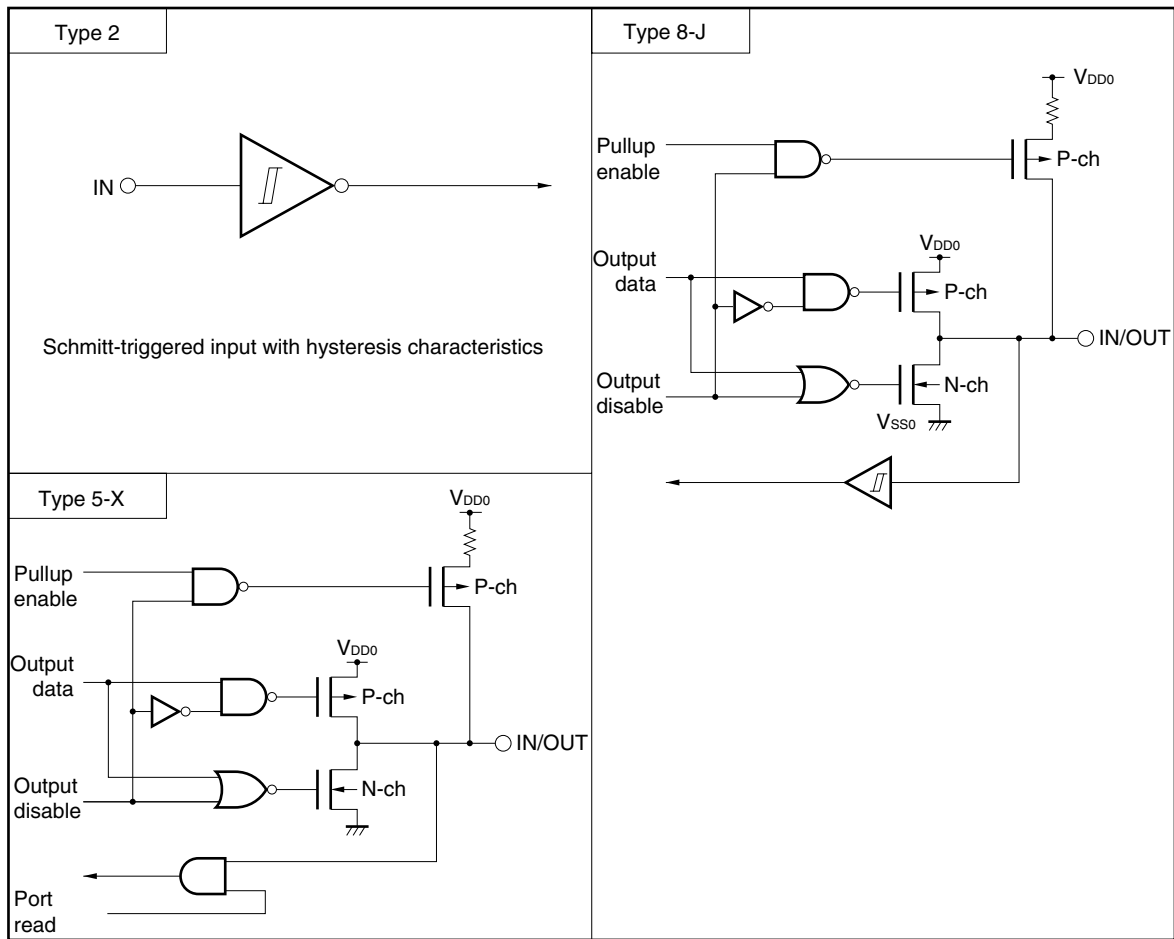
The I/O circuit type of each pin and recommended connection of unused pins are shown in Table 2-1.

For the I/O circuit configuration of each type, see Figure 2-1.

**Table 2-1. Types of Pin I/O Circuits and Recommended Connection of Unused Pins**

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P00 to P07	5-X	I/O	Input: Independently connect to $V_{DD0}$ , $V_{DD1}$ , $V_{SS0}$ , or $V_{SS1}$ via a resistor. Output: Leave open.
P10 to P17			
P20/ASCK/ $\overline{SCK0}$	8-J		
P21/TxD/SO0	5-X		
P22/RxD/SI0	8-J		
P30/INTP0			
P31/INTP1			
P32/INTP2/CPT2			
P40/KR0 to P47/KR7			
P50/TI0/TO0			
P51/TO2	5-X		
P52, P53			
$\overline{\text{RESET}}$	2	Input	—
NC	—	—	Connect directly to $V_{SS0}$ or $V_{SS1}$ (can also be left open).
IC (mask ROM version)			Connect directly to $V_{SS0}$ or $V_{SS1}$ .
$V_{PP}$ ( $\mu\text{PD78F9026A}$ )			Independently connect to a 10 k $\Omega$ pull-down resistor or directly to $V_{SS0}$ or $V_{SS1}$ .

Figure 2-1. Pin I/O Circuits



# CHAPTER 3 CPU ARCHITECTURE

## 3.1 Memory Space

The  $\mu$ PD789026 Subseries can access 64 KB of memory space. Figures 3-1 through 3-5 show the memory maps.

Figure 3-1. Memory Map ( $\mu$ PD789022)

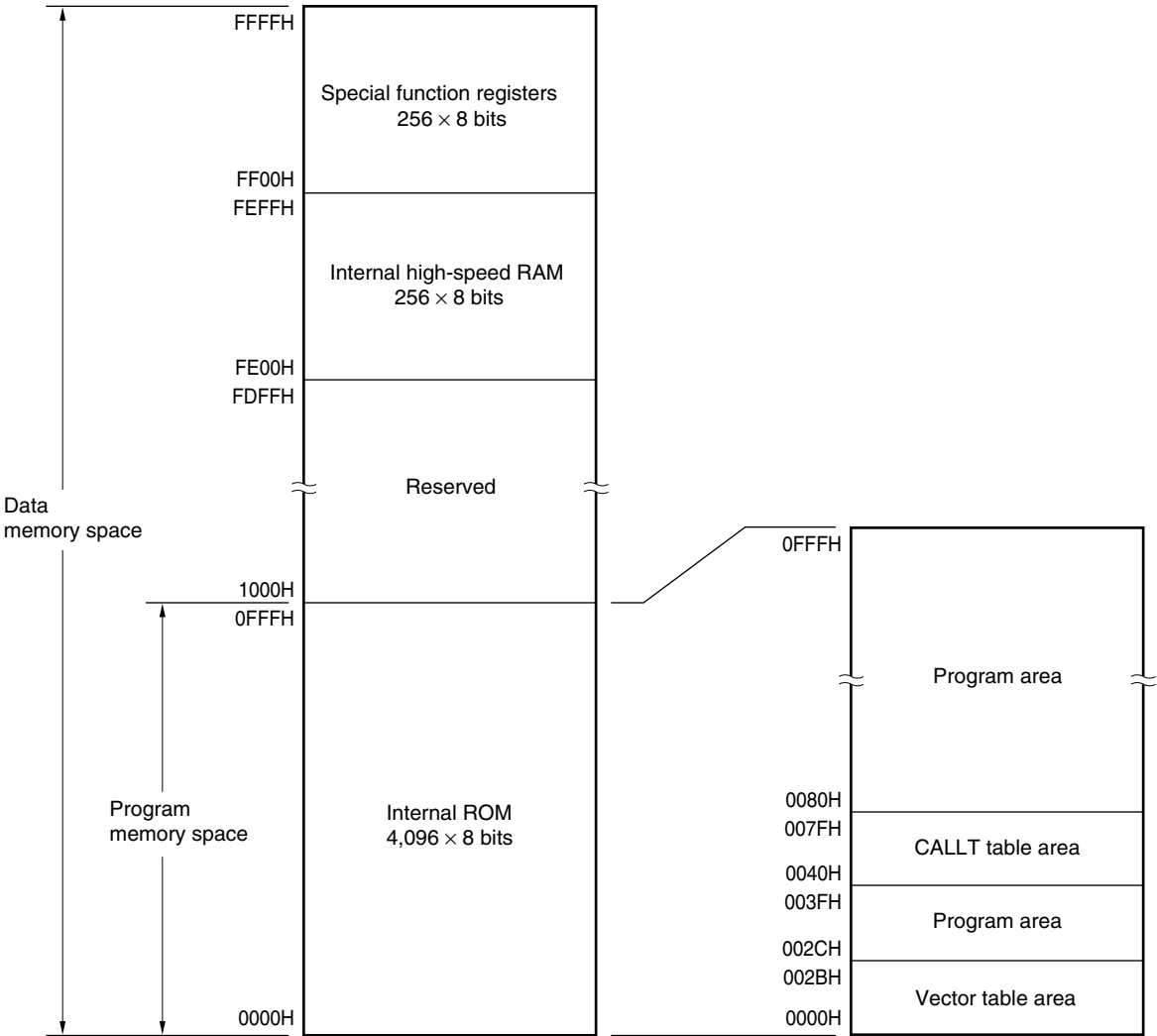


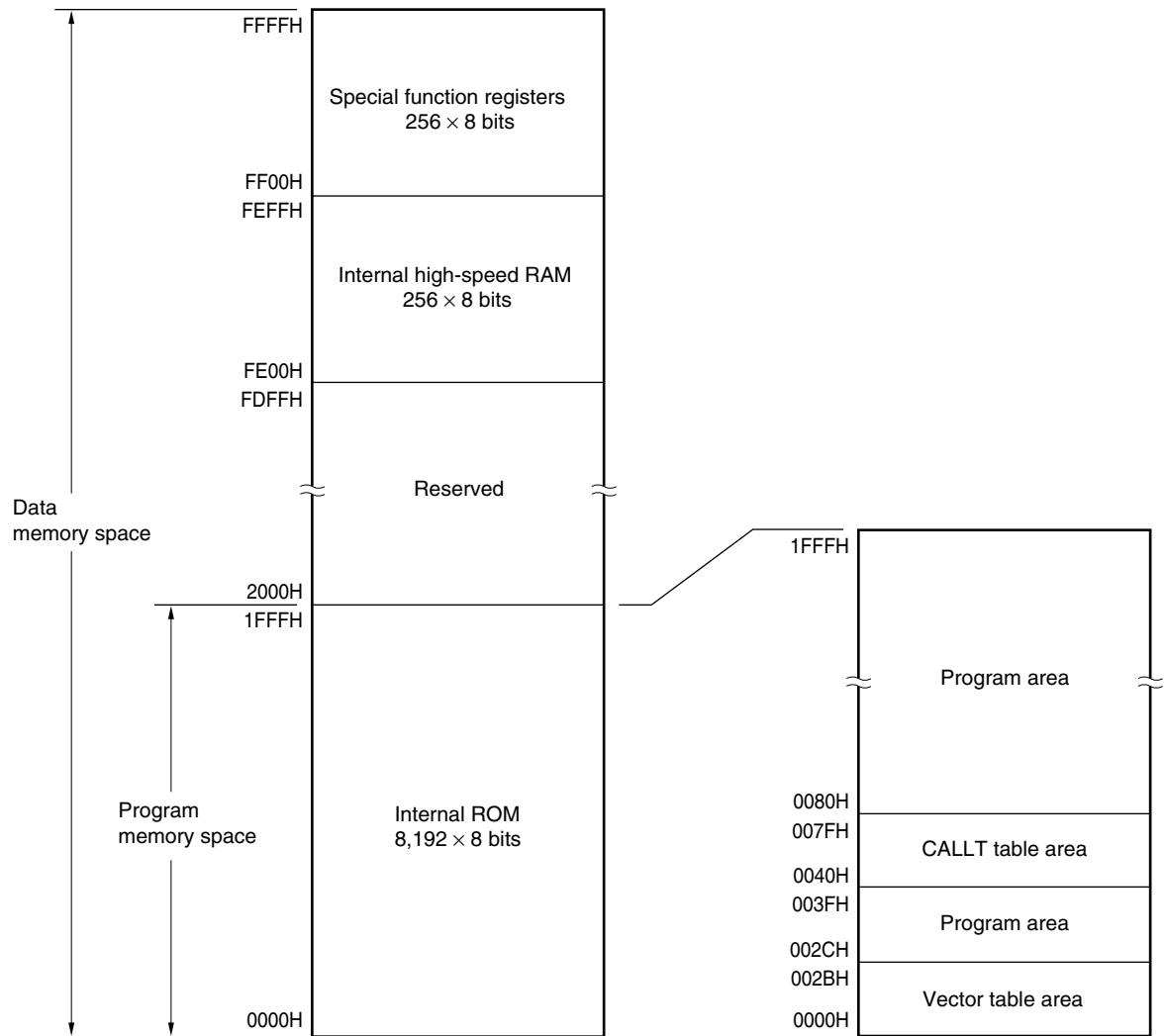
Figure 3-2. Memory Map ( $\mu$ PD789024)

Figure 3-3. Memory Map ( $\mu$ PD789025)

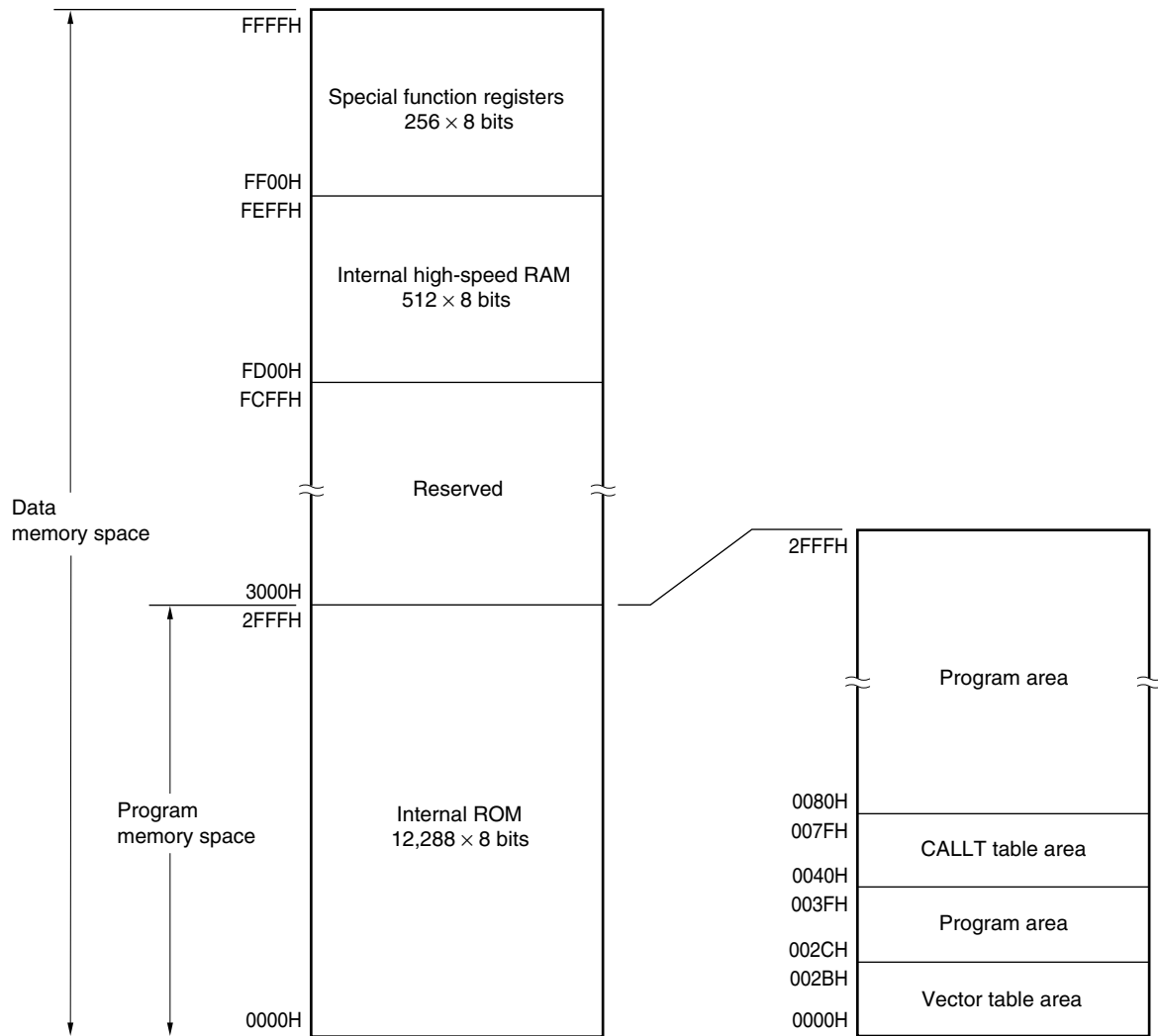


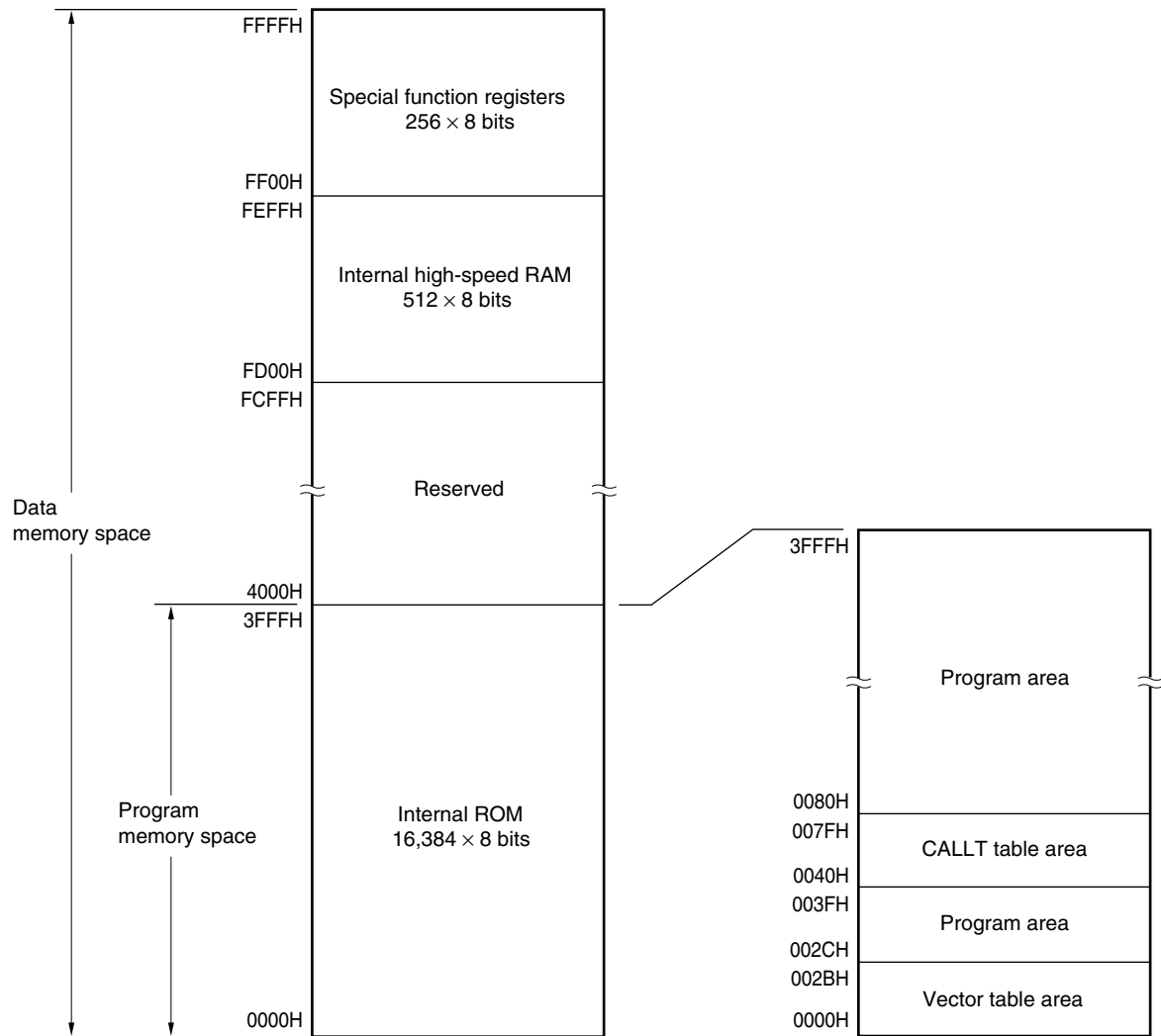
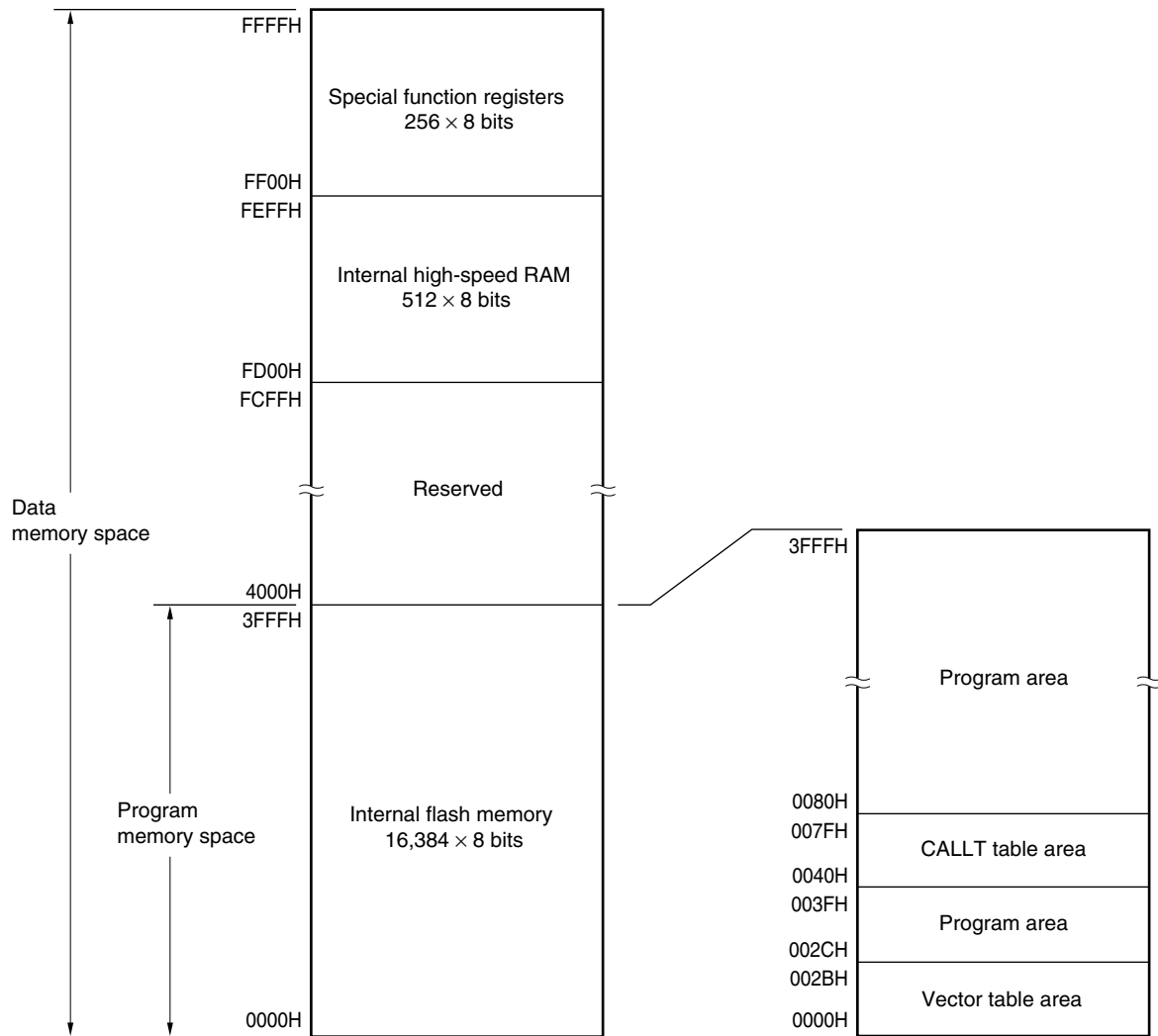
Figure 3-4. Memory Map ( $\mu$ PD789026)

Figure 3-5. Memory Map ( $\mu$ PD78F9026A)



### 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  $\mu$ PD789026 Subseries provides internal ROM (or flash memory) with the following capacities for each product.

**Table 3-1. Internal ROM Capacity**

Part Number	Internal ROM	
	Structure	Capacity
$\mu$ PD789022	Mask ROM	$4,096 \times 8$ bits
$\mu$ PD789024		$8,192 \times 8$ bits
$\mu$ PD789025		$12,288 \times 8$ bits
$\mu$ PD789026		$16,384 \times 8$ bits
$\mu$ PD78F9026A	Flash memory	$16,384 \times 8$ bits

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

#### (1) Vector table area

A 44-byte area of addresses 0000H to 002BH is reserved as a vector table area. This area stores program start addresses to be used when branching by  $\overline{\text{RESET}}$  input or 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/INTCSIO
0004H	INTWDT	000EH	INTST
0006H	INTP0	0010H	INTTM0
0008H	INTP1	0014H	INTTM2
000AH	INTP2	002AH	INTKR

#### (2) CALLT instruction table area

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

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

The  $\mu$ PD789026 Subseries provides internal high-speed RAM with the following capacities for each product.

The internal high-speed RAM cannot be used as a program area for writing and executing instructions.

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

**Table 3-3. Internal High-Speed RAM Capacity**

Part Number	Capacity
$\mu$ PD789022	256 $\times$ 8 bits
$\mu$ PD789024	
$\mu$ PD789025	512 $\times$ 8 bits
$\mu$ PD789026	
$\mu$ PD78F9026A	

### 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 (see **Table 3-4**).

### 3.1.4 Data memory addressing

The  $\mu$ PD789026 Subseries provides a variety of addressing modes which take account of memory manipulability, etc. Especially at addresses corresponding to data memory area (FE00H to FFFFH<sup>Note 1</sup>, FD00H to FFFFH<sup>Note 2</sup>), particular addressing modes can be realized to meet the functions of the special function registers (SFR) and other registers. Figures 3-6 through 3-10 show the data memory addressing modes.

- Notes**
1. With  $\mu$ PD789022 or  $\mu$ PD789024
  2. With  $\mu$ PD789025,  $\mu$ PD789026, or  $\mu$ PD78F9026A

**Figure 3-6. Data Memory Addressing ( $\mu$ PD789022)**

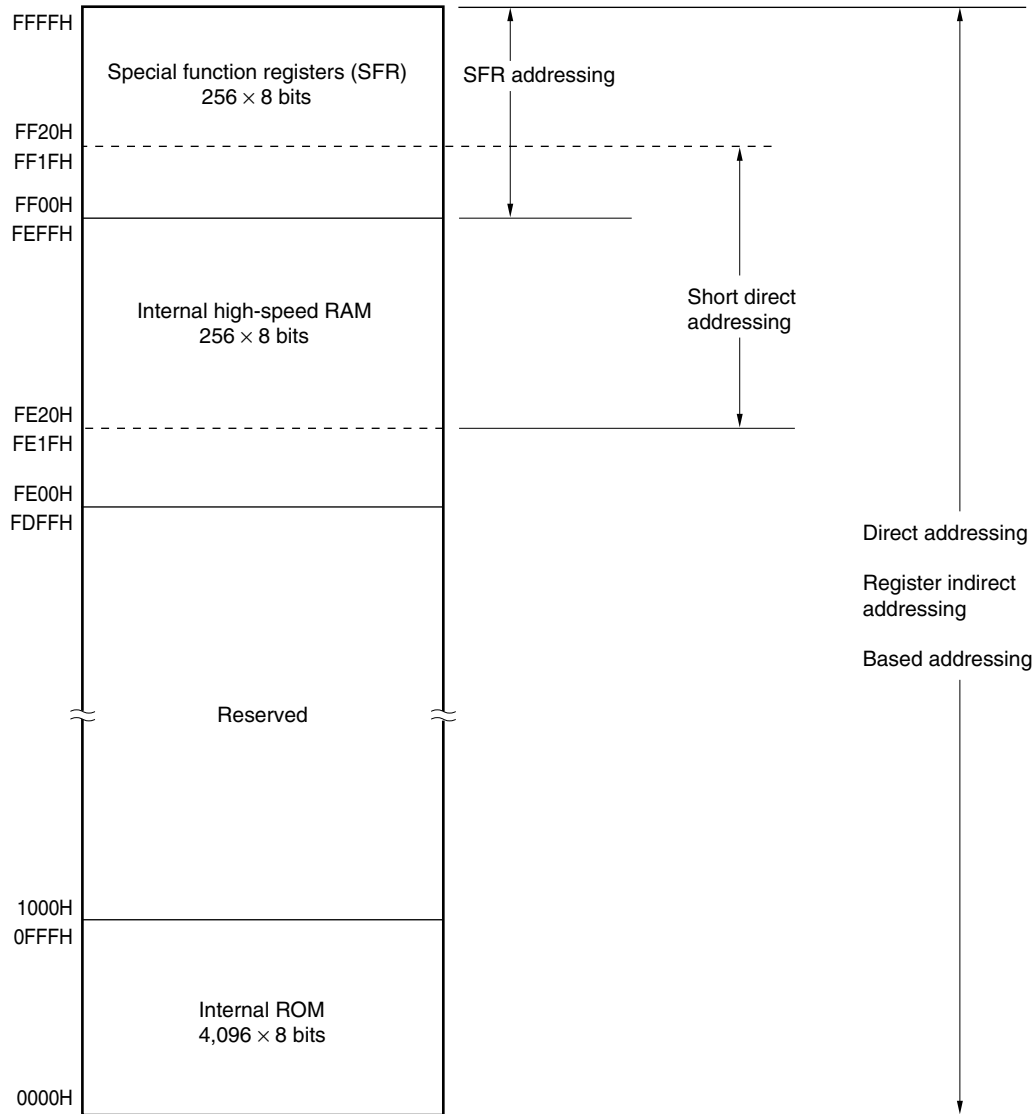


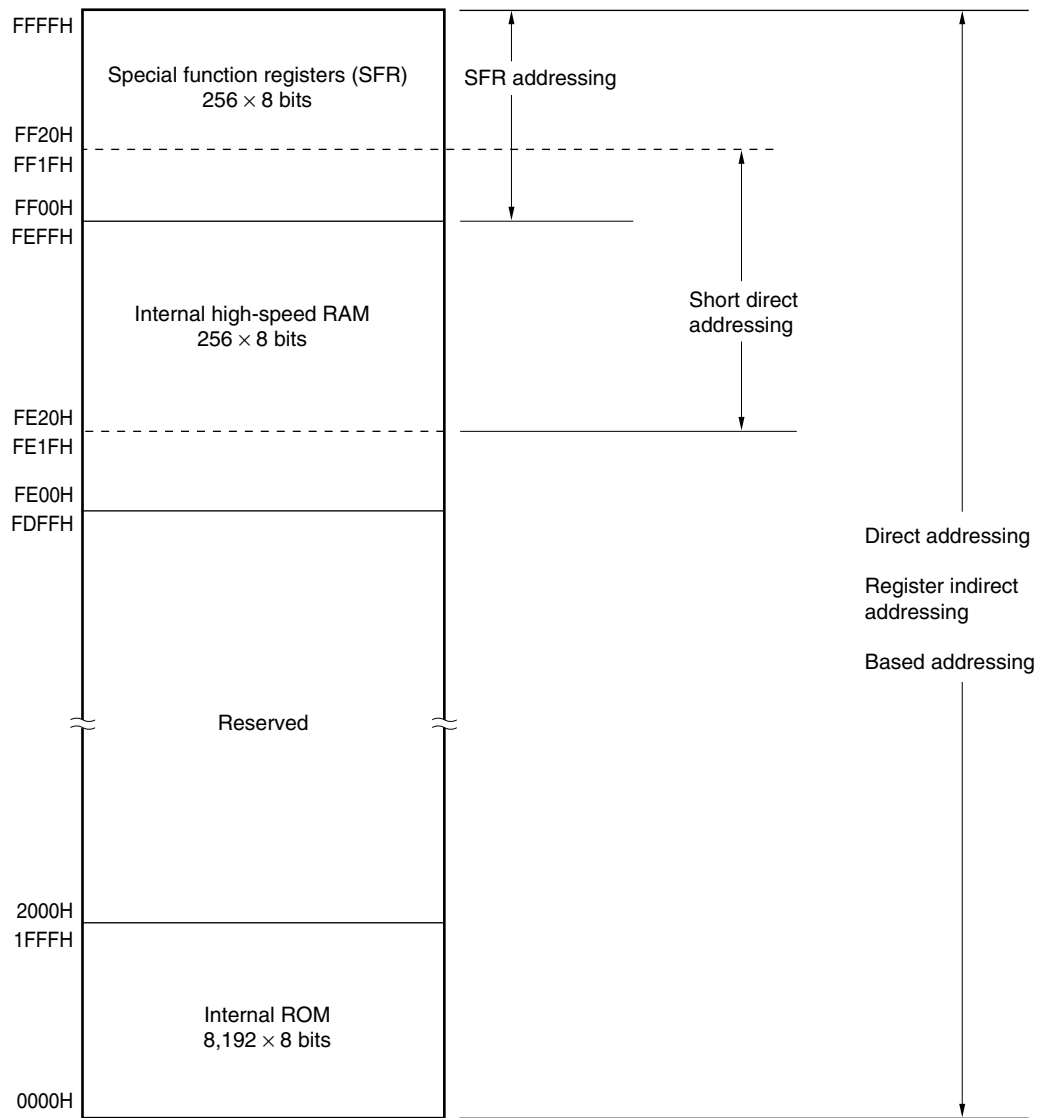
Figure 3-7. Data Memory Addressing ( $\mu$ PD789024)

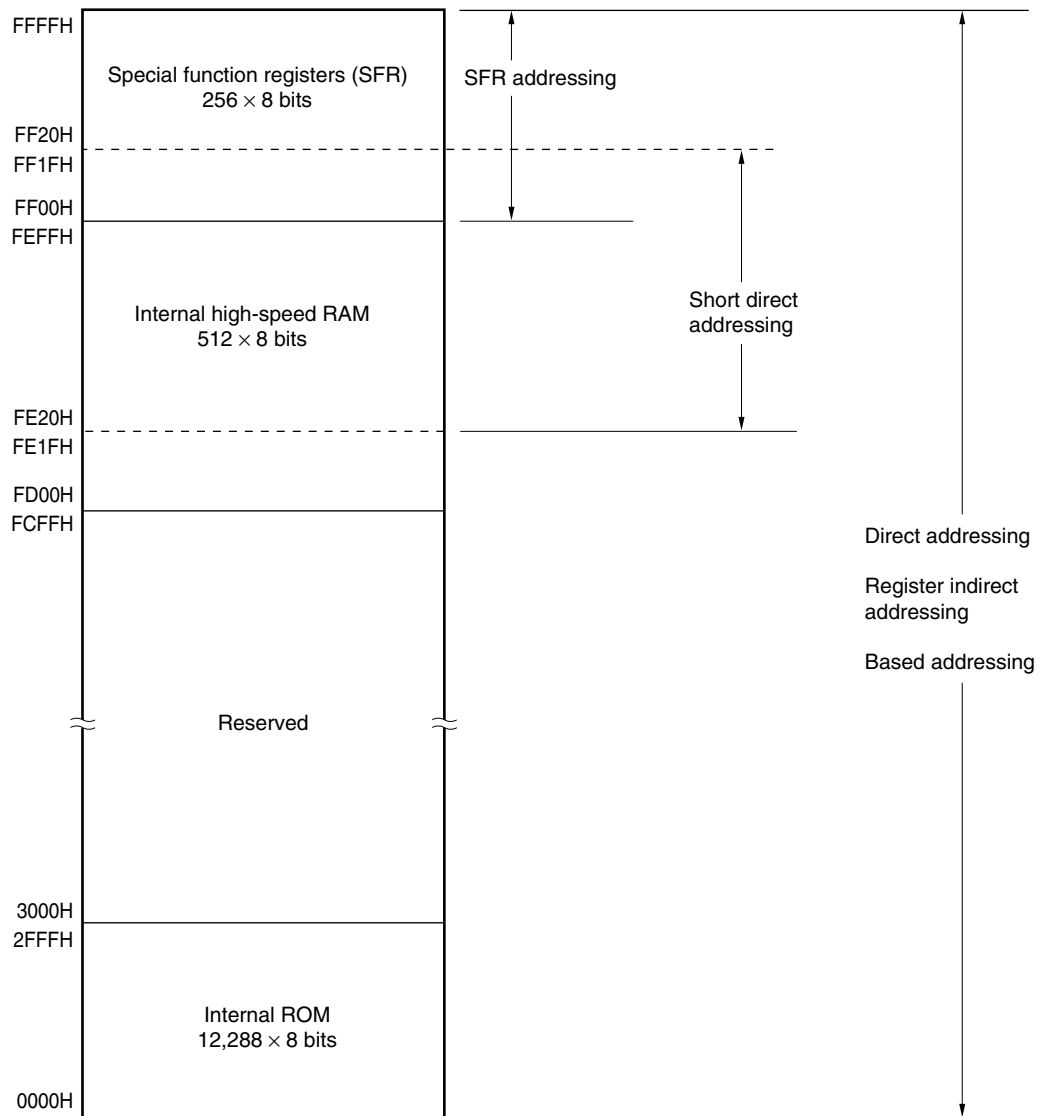
Figure 3-8. Data Memory Addressing ( $\mu$ PD789025)

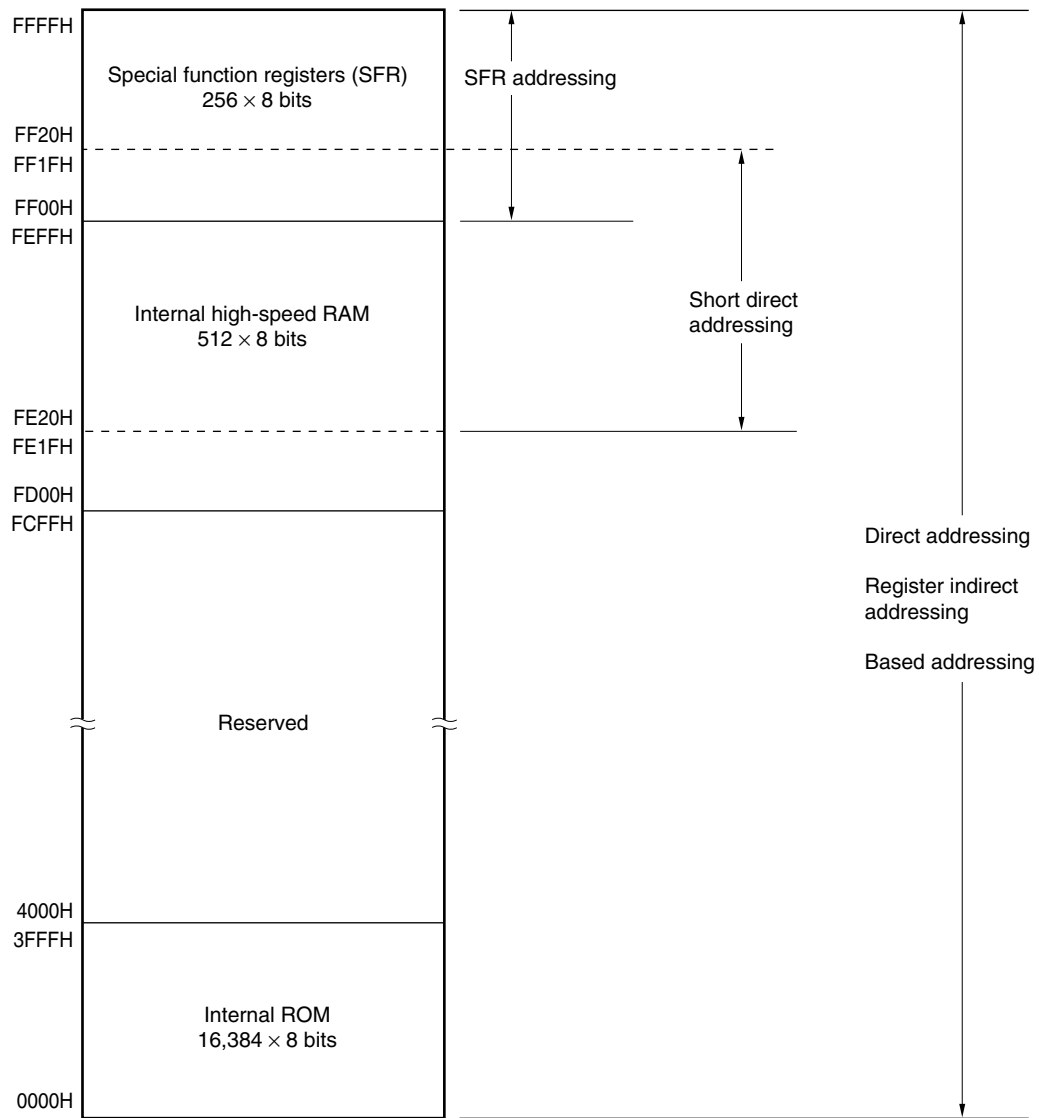
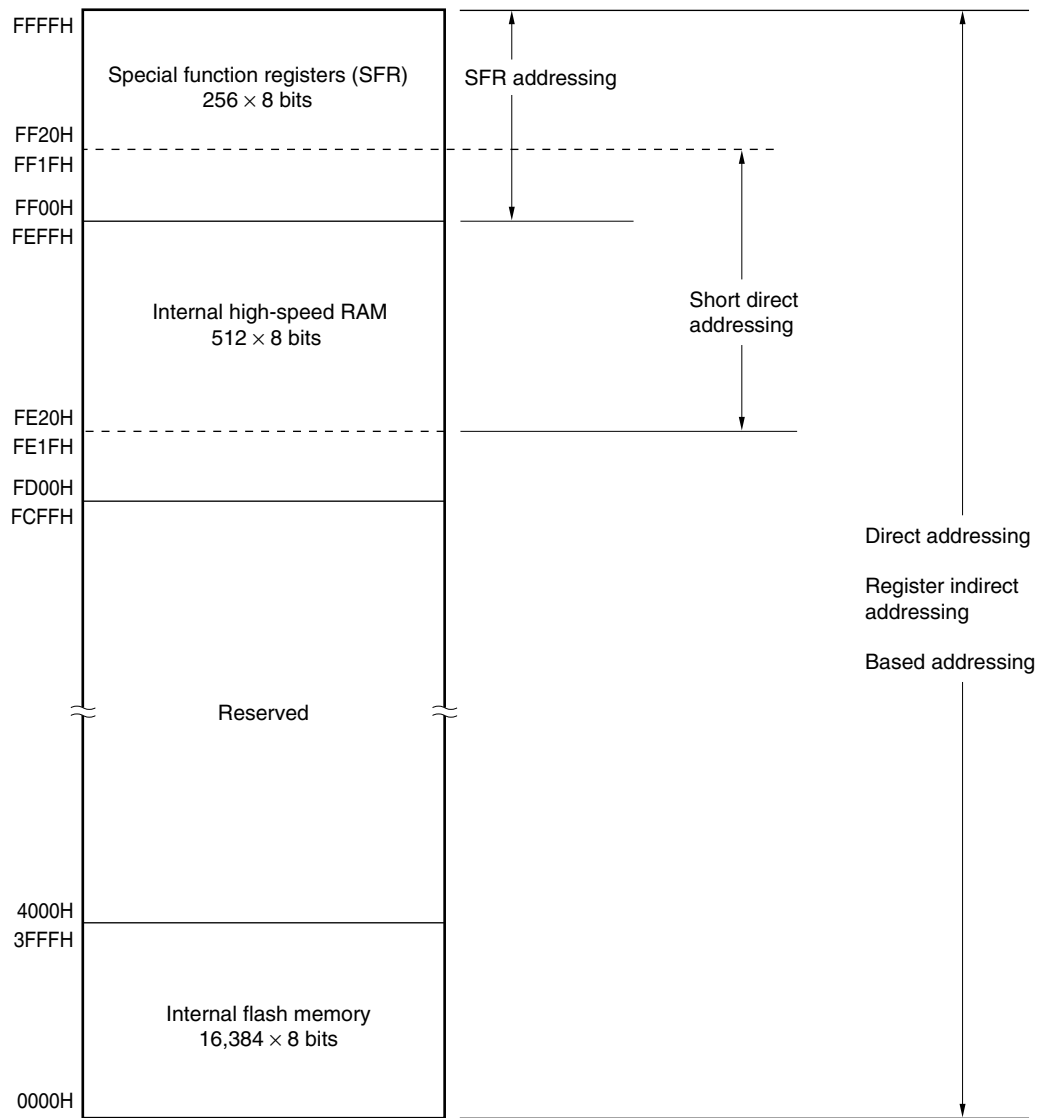
Figure 3-9. Data Memory Addressing ( $\mu$ PD789026)

Figure 3-10. Data Memory Addressing ( $\mu$ PD78F9026A)



## 3.2 Processor Registers

The  $\mu$ PD789026 Subseries provides the following on-chip processor registers.

### 3.2.1 Control registers

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

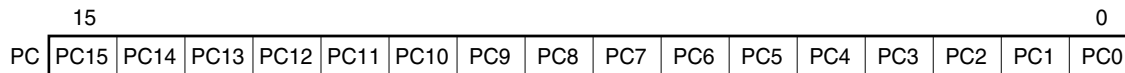
#### (1) Program counter (PC)

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 are set.

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

**Figure 3-11. Program Counter Configuration**



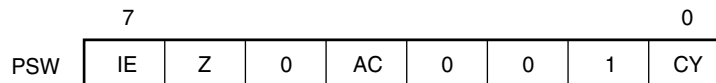
#### (2) Program status word (PSW)

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

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

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

**Figure 3-12. Program Status Word Configuration**





**(a) Interrupt enable flag (IE)**

This flag controls interrupt request acknowledgment operations of the CPU.

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

When IE = 1, the interrupt enabled (EI) status is set and interrupt request acknowledgment is controlled with an interrupt mask flag for each interrupt source.

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.

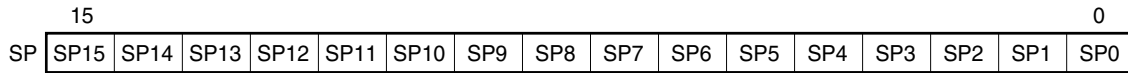
**(d) Carry flag (CY)**

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 operation instruction execution.

### (3) Stack pointer (SP)

This is a 16-bit register used 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 3-13. Stack Pointer Configuration**

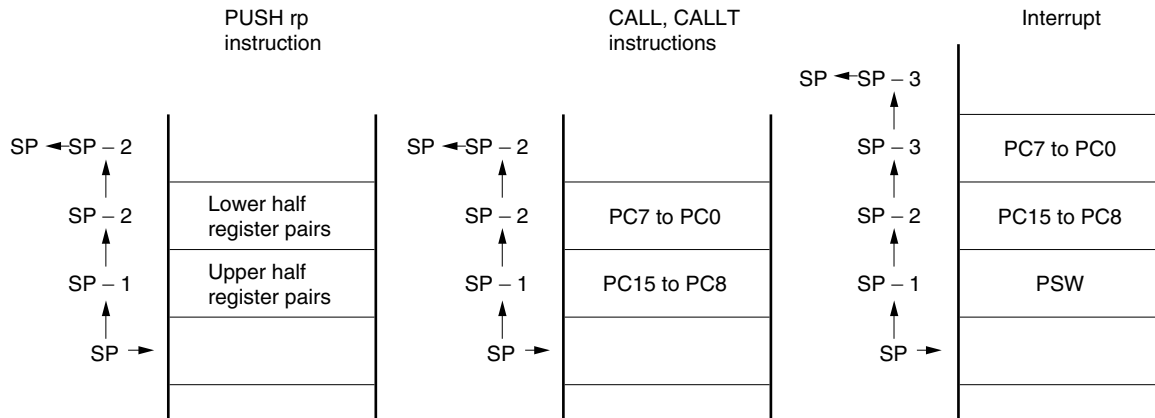


The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.

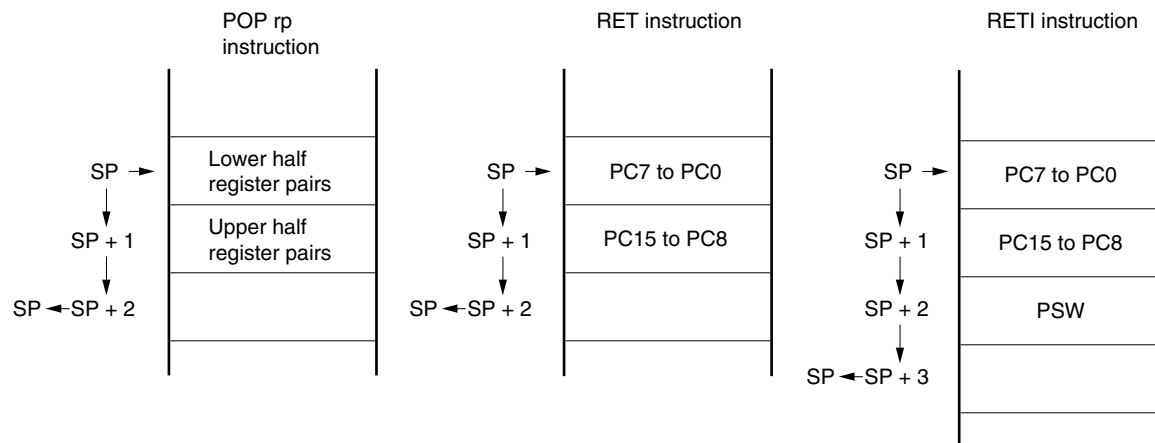
Each stack operation saves/restores data as shown in Figures 3-14 and 3-15.

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

**Figure 3-14. Data to Be Saved to Stack Memory**



**Figure 3-15. Data to Be Restored from Stack Memory**



### 3.2.2 General-purpose registers

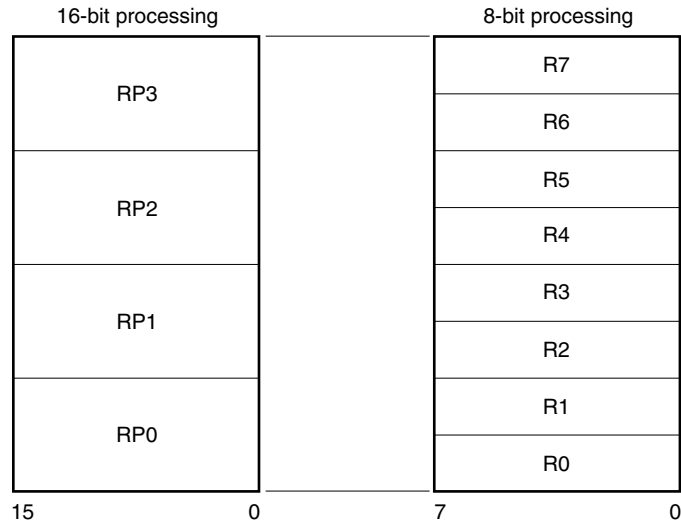
The general-purpose registers consist of eight 8-bit registers (X, A, C, B, E, D, L, and H).

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

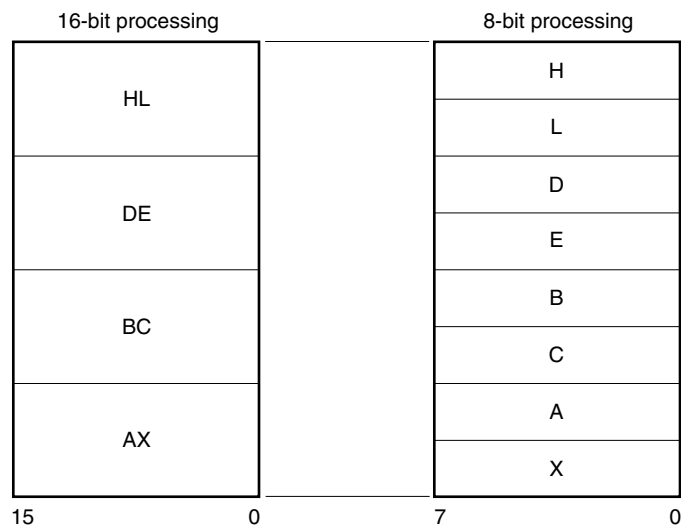
These registers can be written in terms of functional names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

**Figure 3-16. General-Purpose Register Configuration**

**(a) Absolute names**



**(b) Functional names**



### 3.2.3 Special function registers (SFR)

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

Special function registers are allocated in the 256-byte area FF00H to FFFFH.

Special function registers can be manipulated, like general-purpose registers, with operation, transfer, and bit manipulation instructions. The 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  
Writes a symbol reserved by the assembler for the 1-bit manipulation instruction operand (sfr.bit). This manipulation can also be specified using an address.
- 8-bit manipulation  
Writes a symbol reserved by assembler for the 8-bit manipulation instruction operand (sfr). This manipulation can also be specified using an address.
- 16-bit manipulation  
Writes a symbol reserved by assembler for the 16-bit manipulation instruction operand. When specifying an address, write an even address.

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

- Symbol  
Indicates the addresses of the incorporated 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 "sfrbit.h" in the C compiler. Therefore, these symbols can be used as instruction operands if an assembler or integrated debugger is used.
- R/W  
Indicates whether the special function register can be read or written.  
R/W: Read/write  
R: Read only  
W: Write only
- Bit units for manipulation  
Indicates the bit units (1, 8, and 16) in which the special function register 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 Registers (1/2)

Address	Special Function Register (SFR) Name	Symbol		R/W	Bit Units for Manipulation			After Reset
					1 Bit	8 Bits	16 Bits	
FF00H	Port 0	P0		R/W	√	√	—	00H
FF01H	Port 1	P1			√	√	—	
FF02H	Port 2	P2			√	√	—	
FF03H	Port 3	P3			√	√	—	
FF04H	Port 4	P4			√	√	—	
FF05H	Port 5	P5			√	√	—	
FF10H	Transmit shift register 00	TXS00	SIO00	W	—	√	—	FFH
	Receive buffer register 00	RXB00		R	—	√	—	Undefined
FF16H	16-bit compare register 20	CR20		W	—	√ <sup>Note 1</sup>	√ <sup>Note 2</sup>	FFFFH
FF17H								
FF18H	16-bit timer counter 20	TM20		R	—	√ <sup>Note 1</sup>	√ <sup>Note 2</sup>	0000H
FF19H								
FF1AH	16-bit capture register 20	TCP20			—	√ <sup>Note 1</sup>	√ <sup>Note 2</sup>	Undefined
FF1BH								
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			√	√	—	
FF24H	Port mode register 4	PM4			√	√	—	
FF25H	Port mode register 5	PM5			√	√	—	
FF42H	Timer clock select register 2	TCL2			—	√	—	00H
FF50H	8-bit compare register 00	CR00		W	—	√	—	Undefined
FF51H	8-bit timer counter 00	TM00		R	—	√	—	00H
FF53H	8-bit timer mode control register 00	TMC00	R/W	√	√	—		
FF5BH	16-bit timer mode control register 20	TMC20		√	√	—		
FF70H	Asynchronous serial interface mode register 00	ASIM00		√	√	—		
FF71H	Asynchronous serial interface status register 00	ASIS00		R	√	√	—	
FF72H	Serial operation mode register 00	CSIM00		R/W	√	√	—	
FF73H	Baud rate generator control register 00	BRGC00			—	√	—	

- Notes** 1. CR20, TM20, and TCP20 are designed for 16-bit access, but they can also be accessed in 8-bit mode. In 8-bit access mode, use direct addressing.
2. 16-bit access is allowed only with short direct addressing.

Table 3-4. Special Function Registers (2/2)

Address	Special Function Register (SFR) Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFE0H	Interrupt request flag register 0	IF0	R/W	√	√	–	00H
FFE1H	Interrupt request flag register 1	IF1		√	√	–	
FFE4H	Interrupt mask flag register 0	MK0		√	√	–	FFH
FFE5H	Interrupt mask flag register 1	MK1		√	√	–	
FFECH	External interrupt mode register 0	INTM0		–	√	–	00H
FFF5H	Key return mode register 00	KRM00		√	√	–	
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 Instructions User's Manual (U11047E)**).

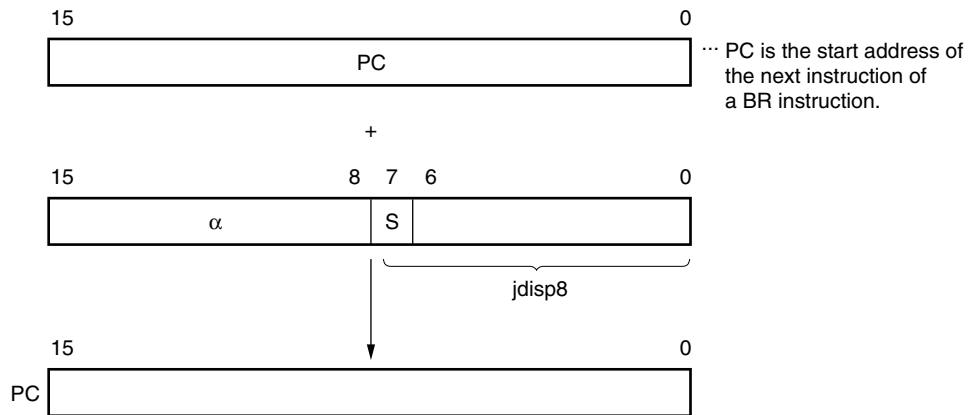
#### 3.3.1 Relative addressing

##### [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]



When S = 0,  $\alpha$  indicates that all bits are "0".  
When S = 1,  $\alpha$  indicates that all bits are "1".

### 3.3.2 Immediate addressing

**[Function]**

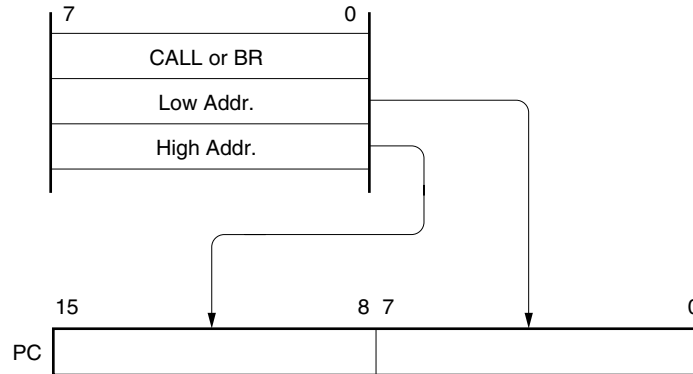
Immediate data in the instruction word is transferred to the program counter (PC) and branched.

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

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

**[Illustration]**

In case of CALL !addr16 or BR !addr16 instruction





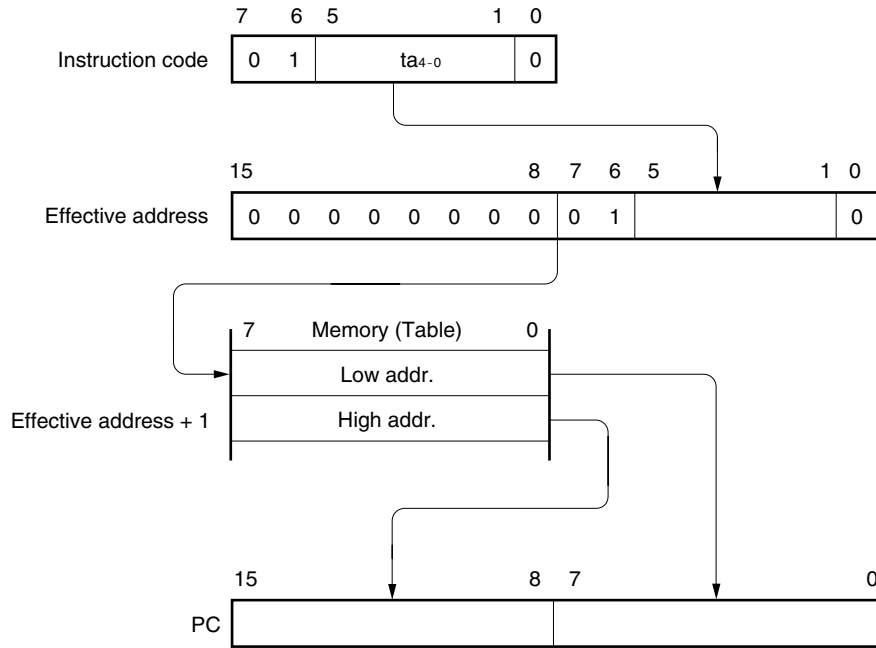
### 3.3.3 Table indirect addressing

#### [Function]

Table contents (branch destination address) of the particular location to be addressed by the 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 reference the address stored in the memory table 40H to 7FH and branch to all the memory spaces.

#### [Illustration]



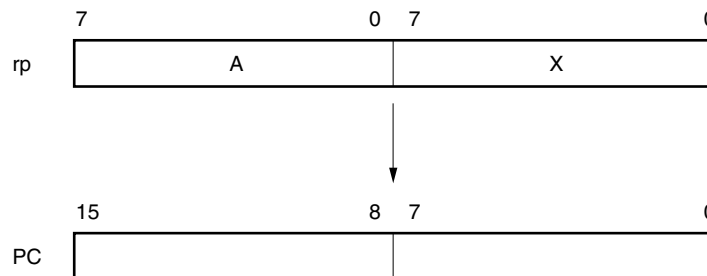
### 3.3.4 Register addressing

#### [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 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]

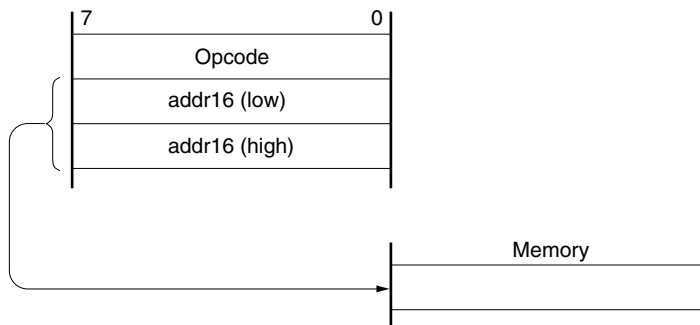
Identifier	Description
addr16	Label or 16-bit immediate data

##### [Example]

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

Instruction code	0	0	1	0	1	0	0	1	Opcode
	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 to which this addressing is applied is the 256-byte space FE20H to FF1FH. An internal high-speed RAM and special function registers (SFR) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

The SFR area (FF00H to FF1FH) where short direct addressing is applied is a part of the total 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.

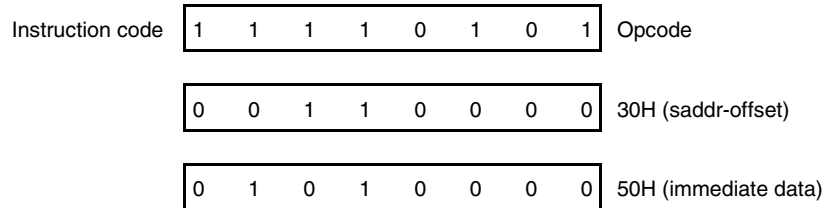
When 8-bit immediate data is at 20H to FFH, bit 8 of an effective address is set to 0. When it is at 00H to 1FH, bit 8 is set to 1. See [Illustration].

#### [Operand format]

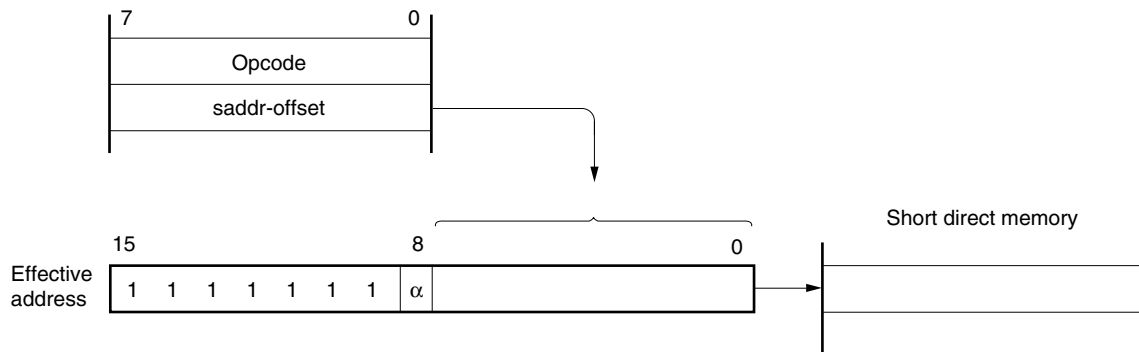
Identifier	Description
saddr	Label or FE20H to FF1FH immediate data
saddrp	Label or FE20H to FF1FH immediate data (even address only)

#### [Example]

MOV FE30H, #50H; When setting saddr to FE30H and the immediate data to 50H



#### [Illustration]



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

### 3.4.3 Special function register (SFR) addressing

#### [Function]

A memory-mapped special function register (SFR) is addressed with 8-bit immediate data in an instruction word.

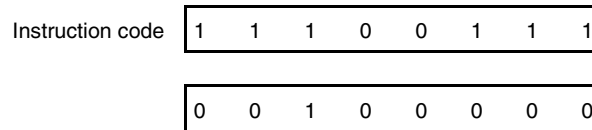
This addressing is applied to the 240-byte spaces FF00H to FF CFH and FFE0H to FFFFH. However, the SFRs mapped at FF00H to FF1FH can also be accessed with short direct addressing.

#### [Operand format]

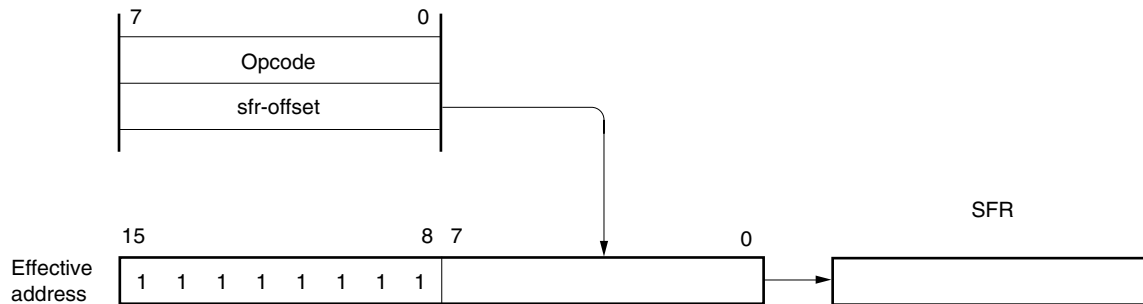
Identifier	Description
sfr	Special function register name

#### [Example]

MOV PM0, A; When selecting PM0 for sfr



#### [Illustration]



### 3.4.4 Register addressing

#### [Function]

A general-purpose register is accessed as an operand. The general-purpose register to be accessed is specified by the register specification 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.

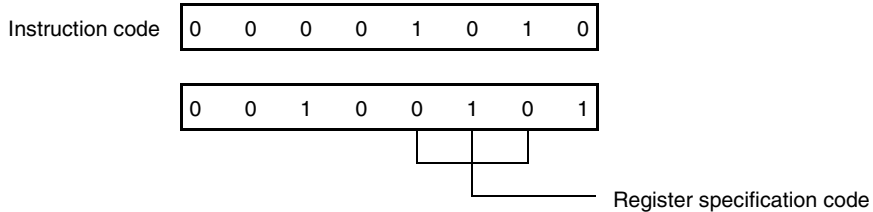
#### [Operand format]

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

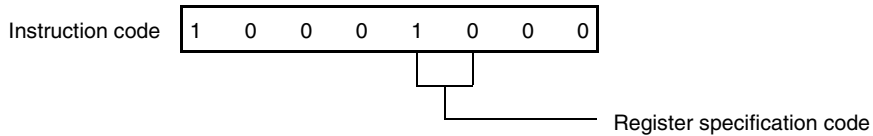
r and rp can be written with absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

#### [Example]

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



INCW DE; When selecting the DE register pair for rp



### 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 specification code in the instruction code. This addressing can be carried out for all the memory spaces.

#### [Operand format]

Identifier	Description
–	[DE], [HL]

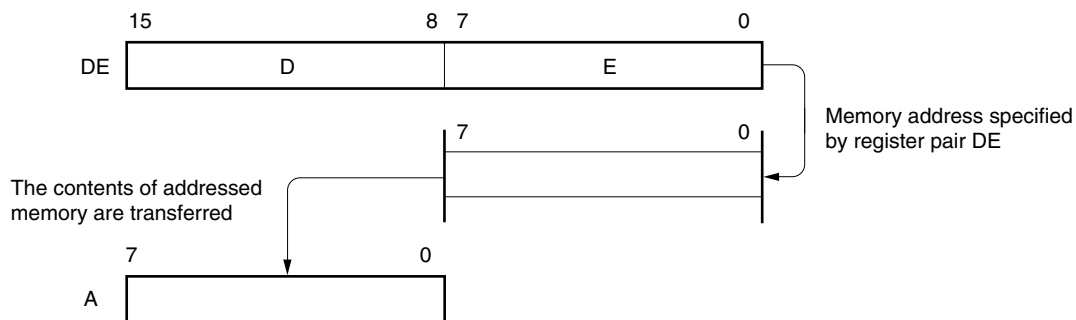
#### [Example]

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

Instruction code 

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

#### [Illustration]



### 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]

#### [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 can be used to access the internal high-speed RAM area only.

#### [Example]

In the case of PUSH DE

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

## CHAPTER 4 PORT FUNCTIONS

### 4.1 Functions of Ports

The  $\mu$ PD789026 Subseries provides the ports shown in Figure 4-1, enabling various methods of control.

Alternate functions are provided in addition to the digital I/O port function. For more information on these alternate functions, see **CHAPTER 2 PIN FUNCTIONS**.

**Figure 4-1. Port Types**

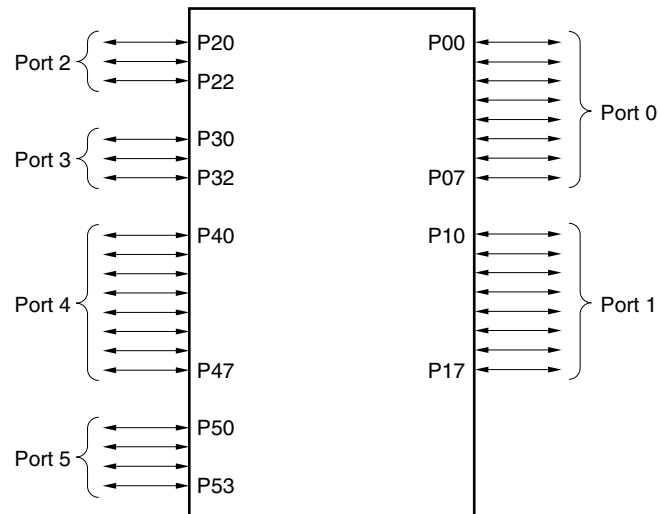




Table 4-1. Port Functions

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P07	I/O	Port 0 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	—
P10 to P17	I/O	Port 1 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	—
P20	I/O	Port 2 3-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	SCK0/ASCK
P21				SO0/TxD
P22				SI0/RxD
P30	I/O	Port 3 3-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	INTP0
P31				INTP1
P32				INTP2/CPT2
P40 to P47	I/O	Port 4 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	KR0 to KR7
P50	I/O	Port 5 4-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of an on-chip pull-up resistor can be specified by setting the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	TI0/TO0
P51				TO2
P52, P53				—

## 4.2 Port Configuration

Ports consist of the following hardware.

**Table 4-2. Port Configuration**

Item	Configuration
Control registers	Port mode register (PMm: m = 0 to 5) Pull-up resistor option register (PUO)
Ports	Total: 34 (I/O: 34)
Pull-up resistors	Total: 34 (on-chip pull-up resistors can be connected by software)

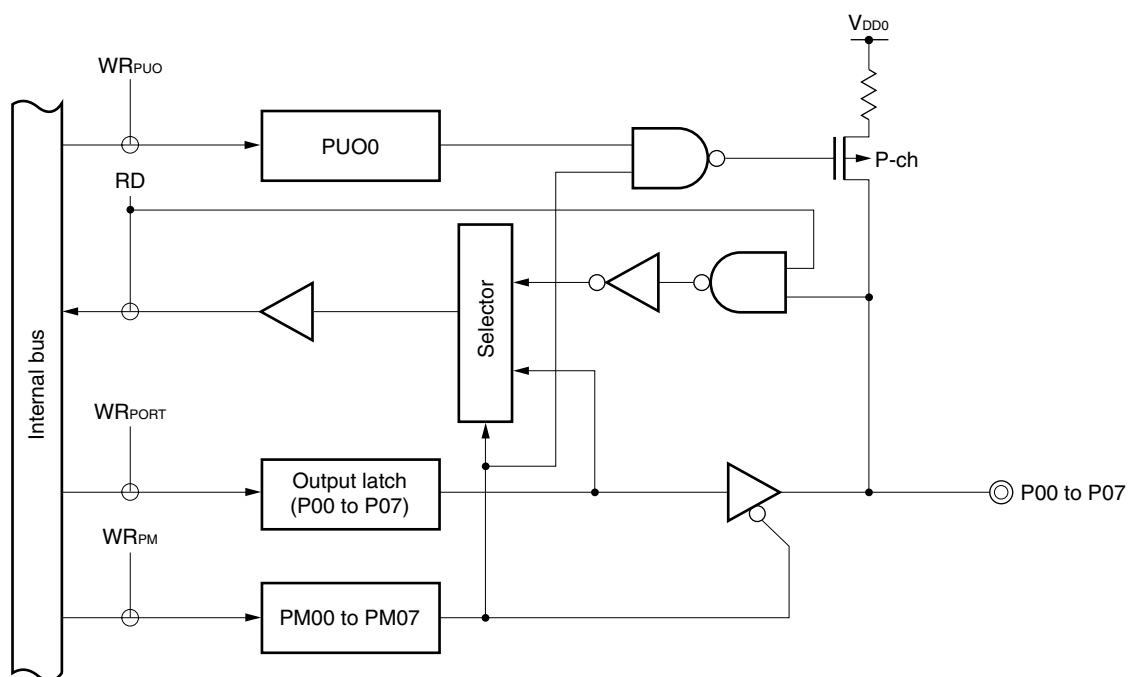
### 4.2.1 Port 0

This is an 8-bit I/O port with an output latch. Port 0 can be specified in the input or output mode in 1-bit units by using port mode register 0 (PM0). When using the P00 to P07 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).

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

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

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



PUO: Pull-up resistor option register

PM: Port mode register

RD: Port 0 read signal

WR: Port 0 write signal

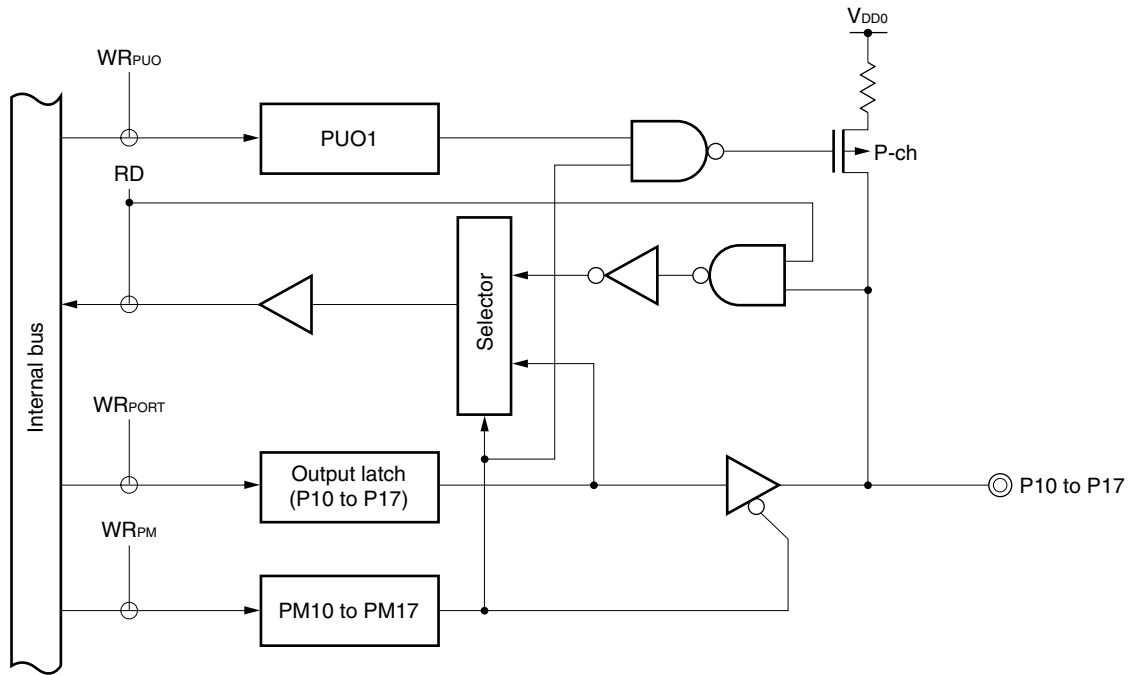
### 4.2.2 Port 1

This is an 8-bit I/O port with an output latch. Port 1 can be specified in the input or output mode in 1-bit units by using port mode register 1 (PM1). When using the 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).

$\overline{\text{RESET}}$  input sets port 1 to input mode.

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

**Figure 4-3. Block Diagram of P10 to P17**



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 an output latch. Port 2 can be specified in the input or output mode in 1-bit units by using port mode register 2 (PM2). When using the P20 to P22 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).

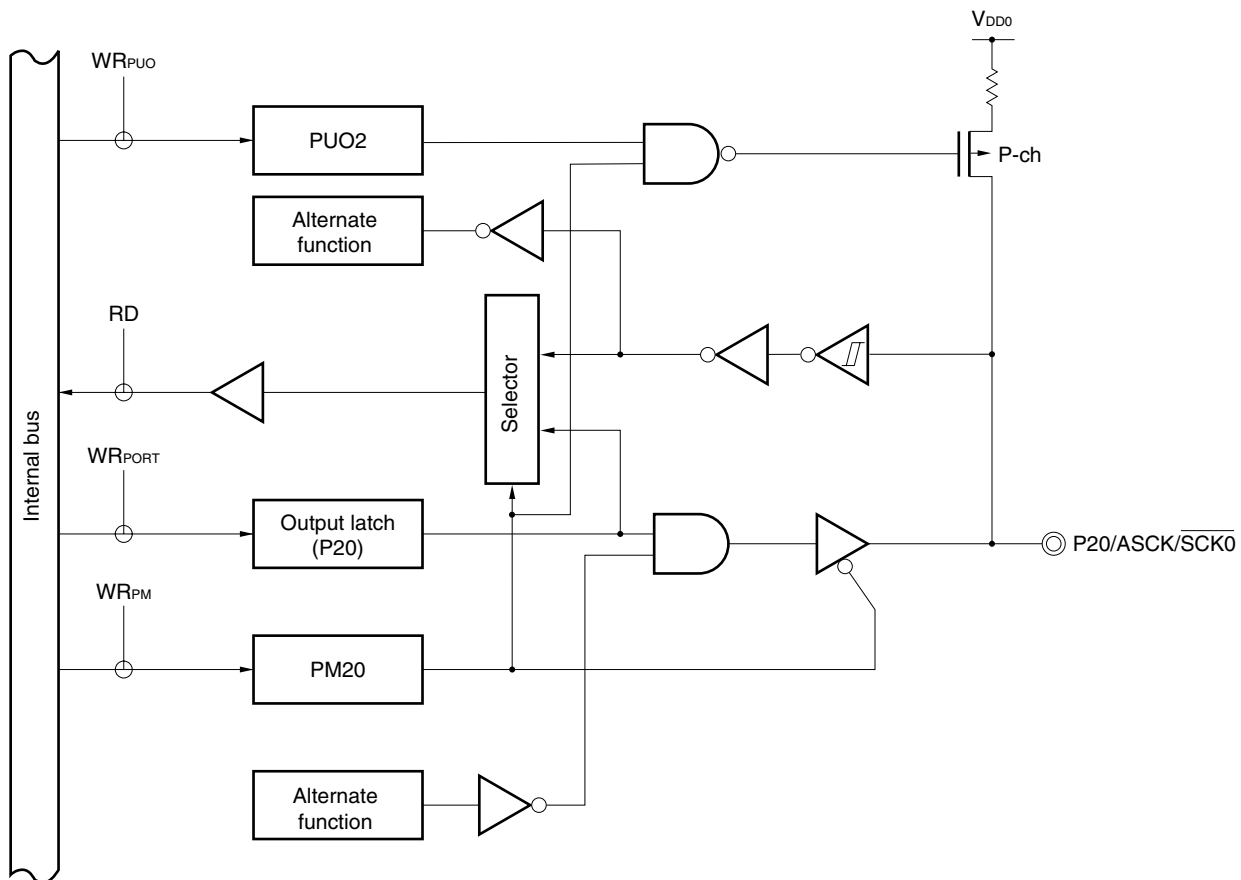
The pins of this port are also used as the data I/O and clock I/O pins of the serial interface.

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

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 mode and output latch must be set according to the function to be used. For details of the settings, see Table 9-2 Operating Mode Settings of Serial Interface 00.

Figure 4-4. Block Diagram of P20



PUO: Pull-up resistor option register

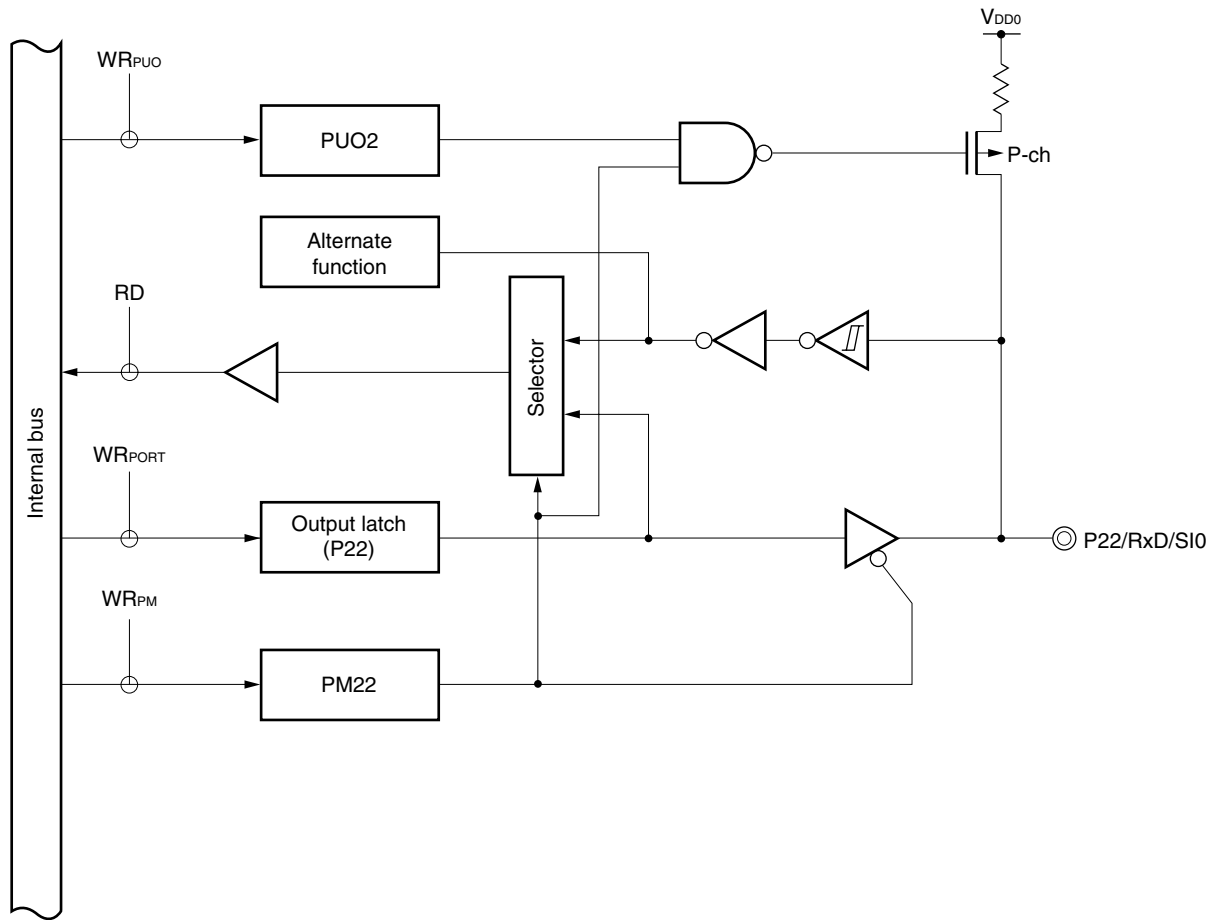
PM: Port mode register

RD: Port 2 read signal

WR: Port 2 write signal

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

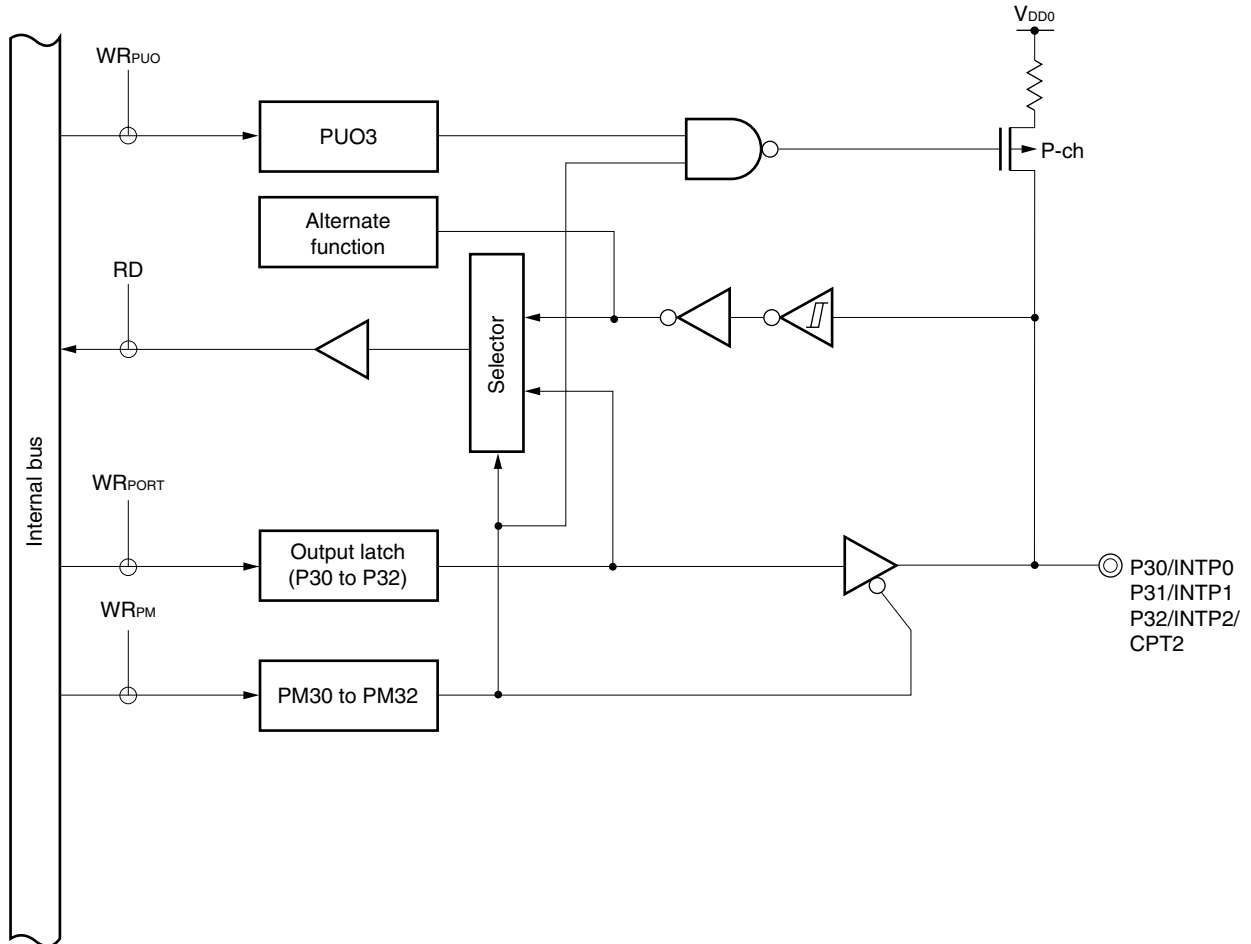
This is a 3-bit I/O port with an output latch. Port 3 can be specified in input or output mode in 1-bit units by using port mode register 3 (PM3). When using the P30 to P32 pins as input port pins, on-chip pull-up resistors can be connected in 3-bit units by using the pull-up resistor option register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

The pins of this port are also used as the external interrupt input and capture edge input.

RESET input sets port 3 to input mode.

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

**Figure 4-7. Block Diagram of P30 to P32**



PUO: Pull-up resistor option register

PM: Port mode register

RD: Port 3 read signal

WR: Port 3 write signal

### 4.2.5 Port 4

This is an 8-bit I/O port with an output latch. Port 4 can be specified in the input or output mode in 1-bit units by using port mode register 4 (PM4). When using the P40 to P47 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).

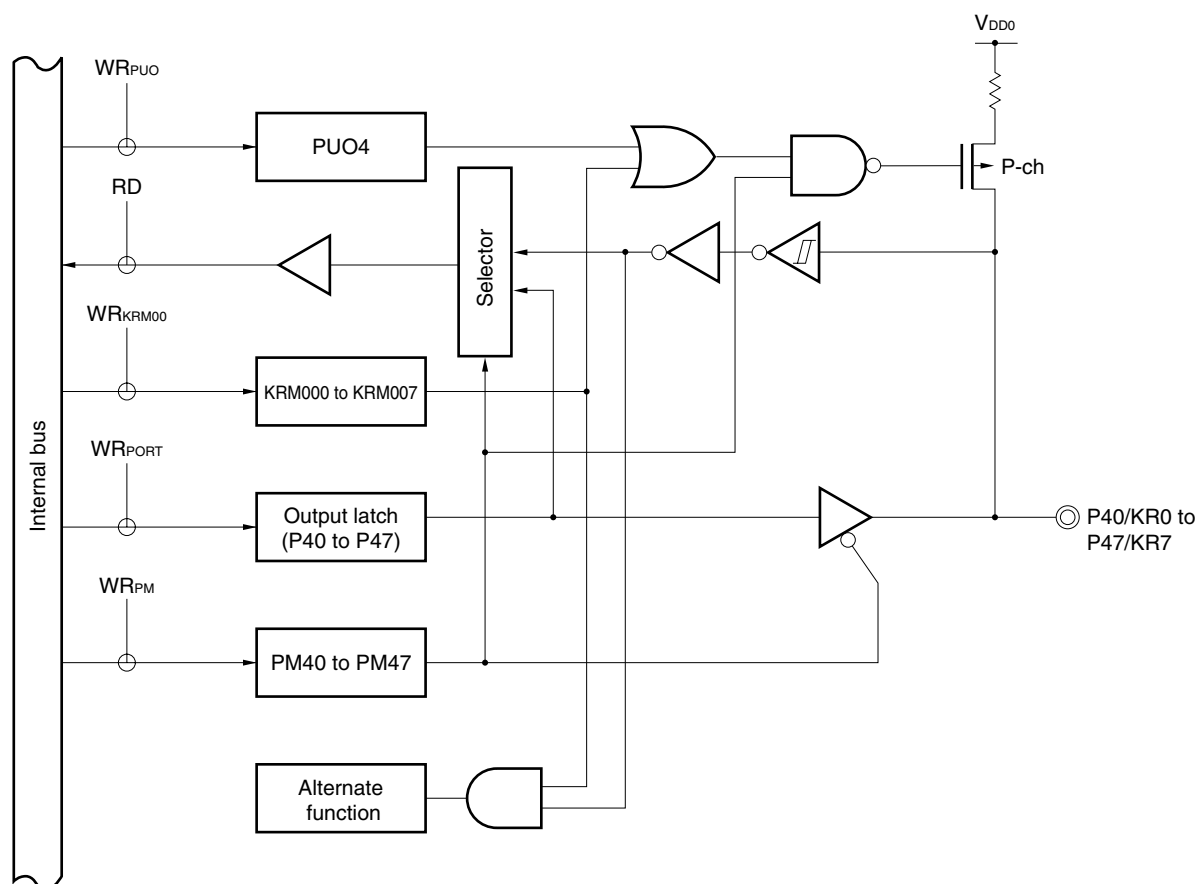
The pins of this port are also used as the key return input.

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

Figure 4-8 shows the block diagram of port 4.

**Caution** When using port 4 for the key return function, it is necessary to set key return mode register 00. For details of the settings, see 10.3 (5) Key return mode register 00 (KRM00).

Figure 4-8. Block Diagram of P40 to P47



KRM00: Key return mode register 00  
 PUO: Pull-up resistor option register  
 PM: Port mode register  
 RD: Port 4 read signal  
 WR: Port 4 write signal



#### 4.2.6 Port 5

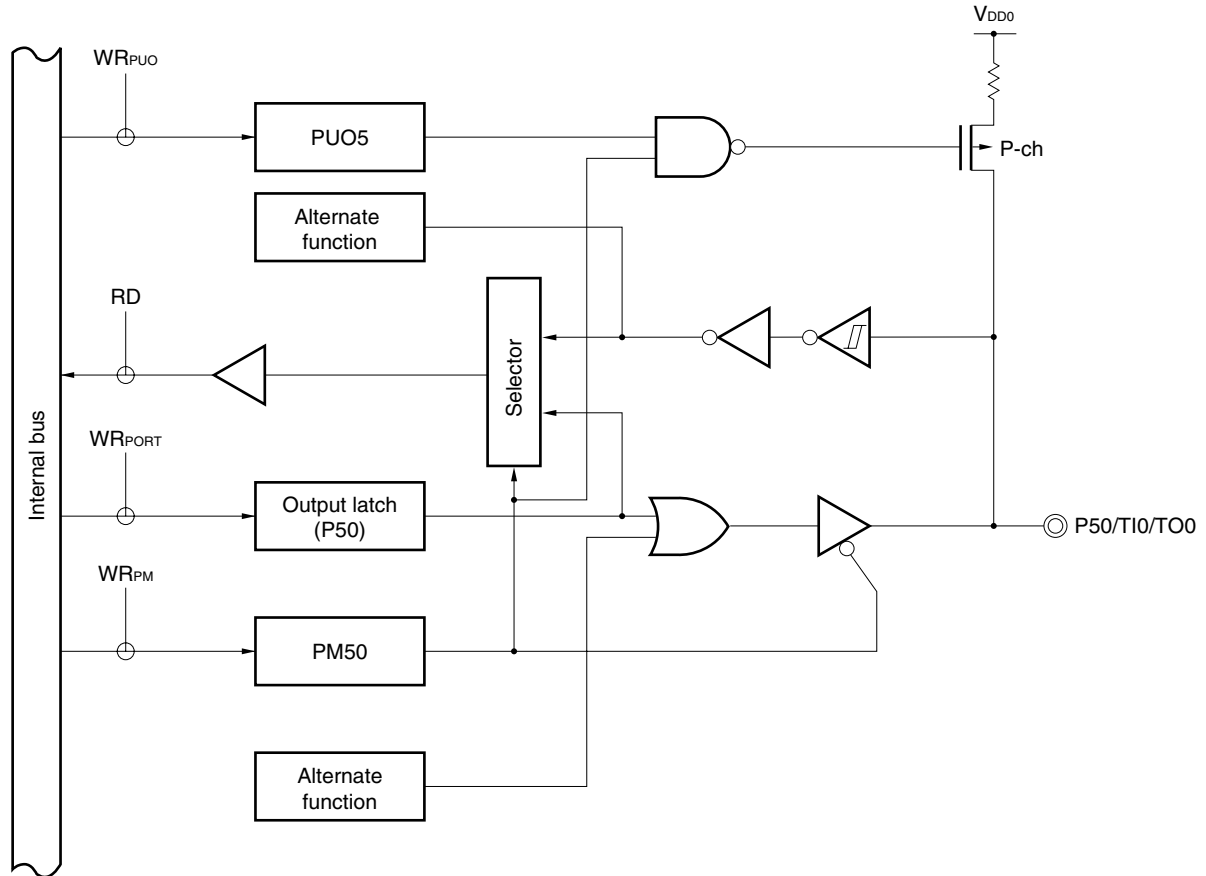
This is a 4-bit I/O port with an output latch. Port 5 can be specified in the input or output mode in 1-bit units by using port mode register 5 (PM5). When using the P50 to P53 pins as input port pins, on-chip pull-up resistors can be connected in 4-bit units by using the pull-up resistor option register (PUO).

The pins of this port are also used as the data I/O pins of the timer.

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

Figures 4-9 through 4-11 show the block diagrams of port 5.

**Figure 4-9. Block Diagram of P50**



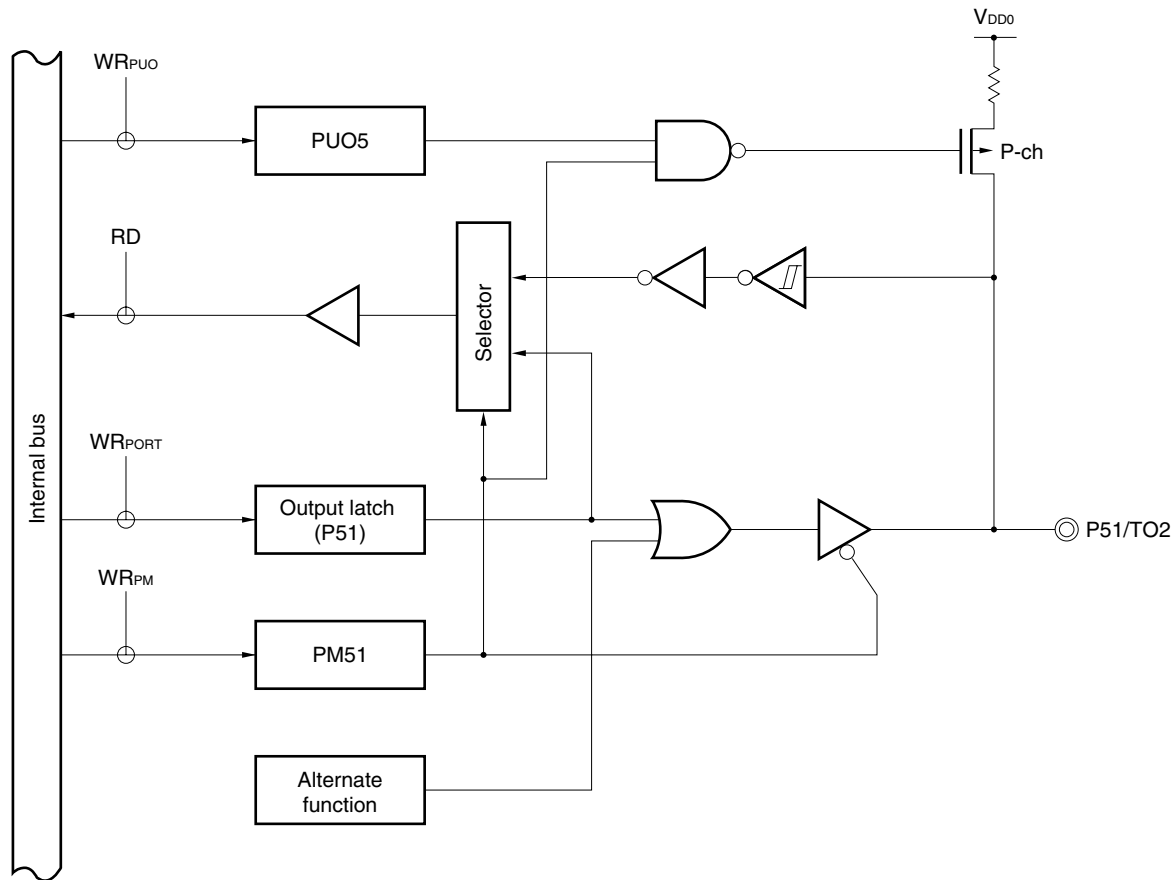
PUO: Pull-up resistor option register

PM: Port mode register

RD: Port 5 read signal

WR: Port 5 write signal

Figure 4-10. Block Diagram of P51



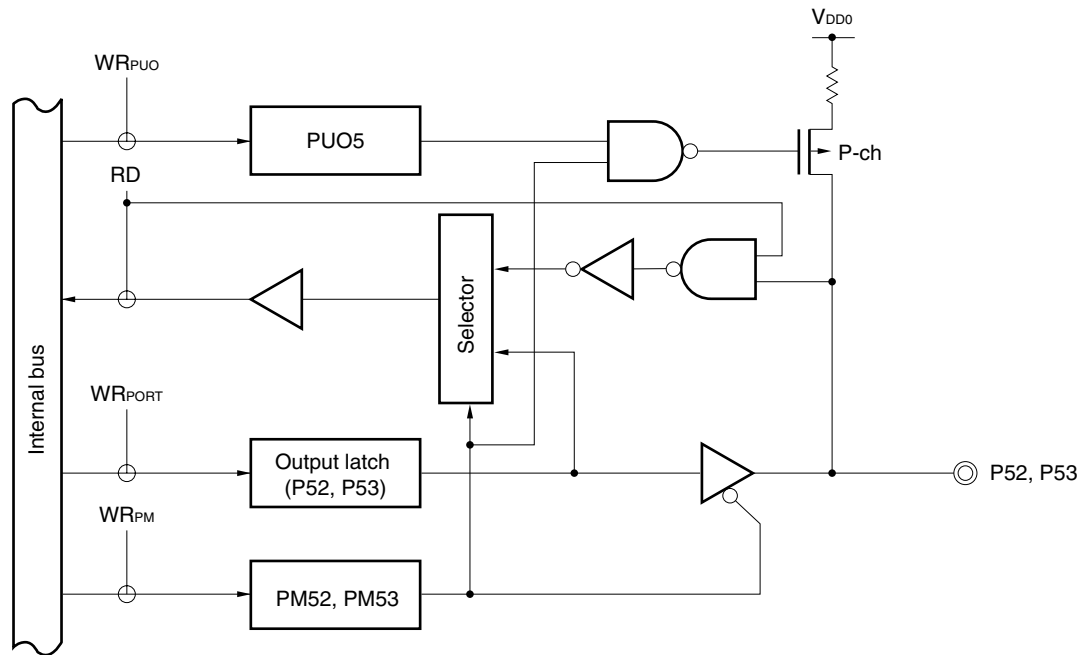
PUO: Pull-up resistor option register

PM: Port mode register

RD: Port 5 read signal

WR: Port 5 write signal

Figure 4-11. Block Diagram of P52 and P53



PUO: Pull-up resistor option register

PM: Port mode register

RD: Port 5 read signal

WR: Port 5 write signal

### 4.3 Registers Controlling Port Function

The following two types of registers control the ports.

- Port mode registers (PM0 to PM5)
- Pull-up resistor option register (PUO)

#### (1) Port mode registers (PM0 to PM5)

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

Port mode registers 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 the 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 <sub>xx</sub>	P <sub>xx</sub>
	Name	I/O		
P30	INTP0	Input	1	×
P31	INTP1	Input	1	×
P32	INTP2	Input	1	×
	CPT2	Input	1	×
P40 to P47 <sup>Note</sup>	KR0 to KR7	Input	1	×
P50	TI0	Input	1	×
	TO0	Output	0	0
P51	TO2	Output	0	0

**Note** When an alternate function is used, set key return mode register 00 (KRM00) to 1 (see **10.3 (5) Key return mode register 00 (KRM00)**).

**Caution** When port 2 is used as serial interface pins, the I/O mode and output latch must be set according to the function to be used. For details of the settings, see Table 9-2 Operating Mode Settings of Serial Interface 00.

**Remark** ×: Don't care  
 PM<sub>xx</sub>: Port mode register  
 P<sub>xx</sub>: Output latch of port

Figure 4-12. Format of Port Mode Register

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
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	FFH	R/W
PM5	1	1	1	1	PM53	PM52	PM51	PM50	FF25H	FFH	R/W

PMmn	Pmn pin I/O mode selection $\left( \begin{array}{l} m = 0, 1, 4: n = 0 \text{ to } 7 \\ m = 2, 3: n = 0 \text{ to } 2 \\ m = 5: n = 0 \text{ to } 3 \end{array} \right)$
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 is used on each port or not.

On the port specified by PUO to use an on-chip pull-up resistor, 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 for the bits set in the output mode regardless of the setting of PUO. On-chip pull-up resistors cannot be used 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 clears PUO to 00H.

Figure 4-13. Format of Pull-Up Resistor Option Register

Symbol	7	6	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
PUO	0	0	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	FFF7H	00H	R/W

PUOm	Port m on-chip pull-up resistor selection (m = 0 to 5)
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.

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

#### (2) In input mode

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

Once data is written to the output latch, it 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 a port consisting both of inputs and outputs, 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 contents of an output latch 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 on the contents of an output latch. The result of the operation is written to the output latch. The contents of the output latch are output from the port pins.

Once data is written to the output latch, it 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 a port consisting both of inputs and outputs, 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 following type of system clock oscillator is used.

- System clock oscillator

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

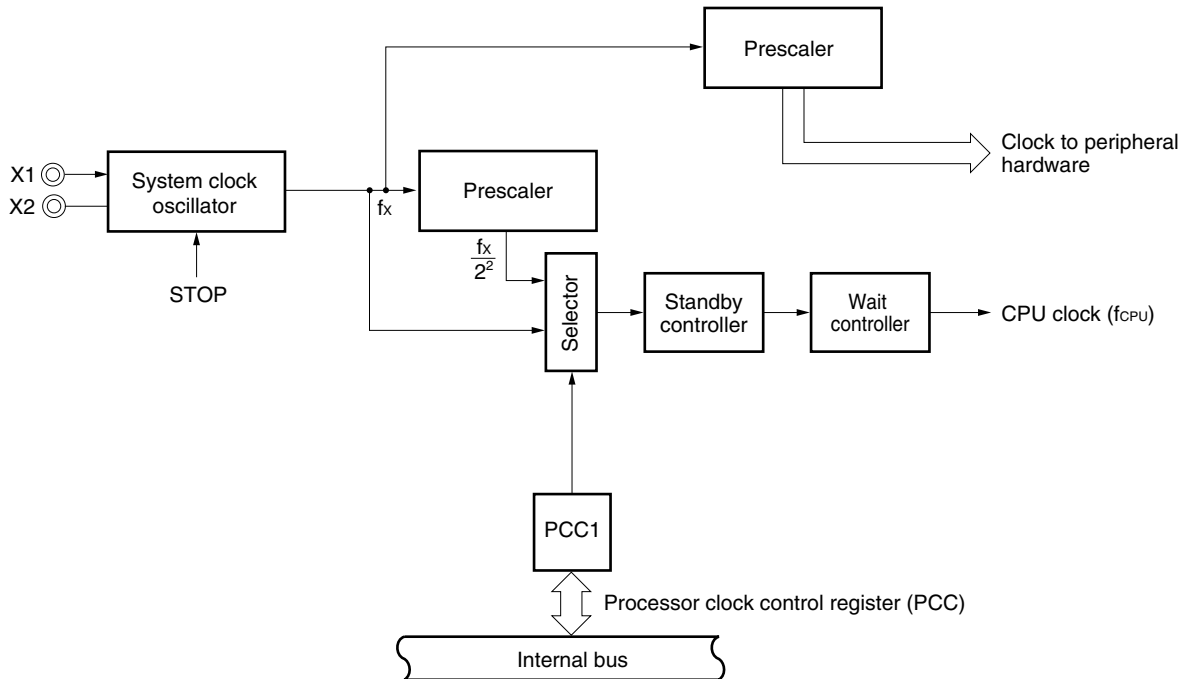
### 5.2 Configuration of Clock Generator

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 Register Controlling Clock Generator

The clock generator is controlled by the following register.

- Processor clock control register (PCC)

#### (1) Processor clock control register (PCC)

PCC sets CPU clock selection and the division ratio.

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

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

**Figure 5-2. Format of Processor Clock Control Register**

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 ( $f_{\text{CPU}}$ ) selection		Minimum instruction execution time: $2/f_{\text{CPU}}$
			$f_x = 5.0 \text{ MHz}$
0	$f_x$		$0.4 \mu\text{s}$
1	$f_x/2^2$		$1.6 \mu\text{s}$

**Caution** Be sure to set bits 0 and 2 to 7 to 0.

**Remark**  $f_x$ : System clock oscillation frequency



## 5.4 System Clock Oscillator

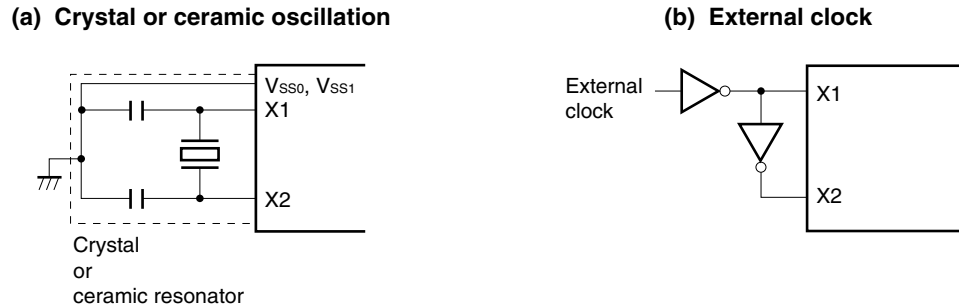
### 5.4.1 System clock oscillator

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 inverted 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**



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

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

### 5.4.2 Examples of incorrect resonator connection

Figure 5-4 shows examples of incorrect resonator connection.

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

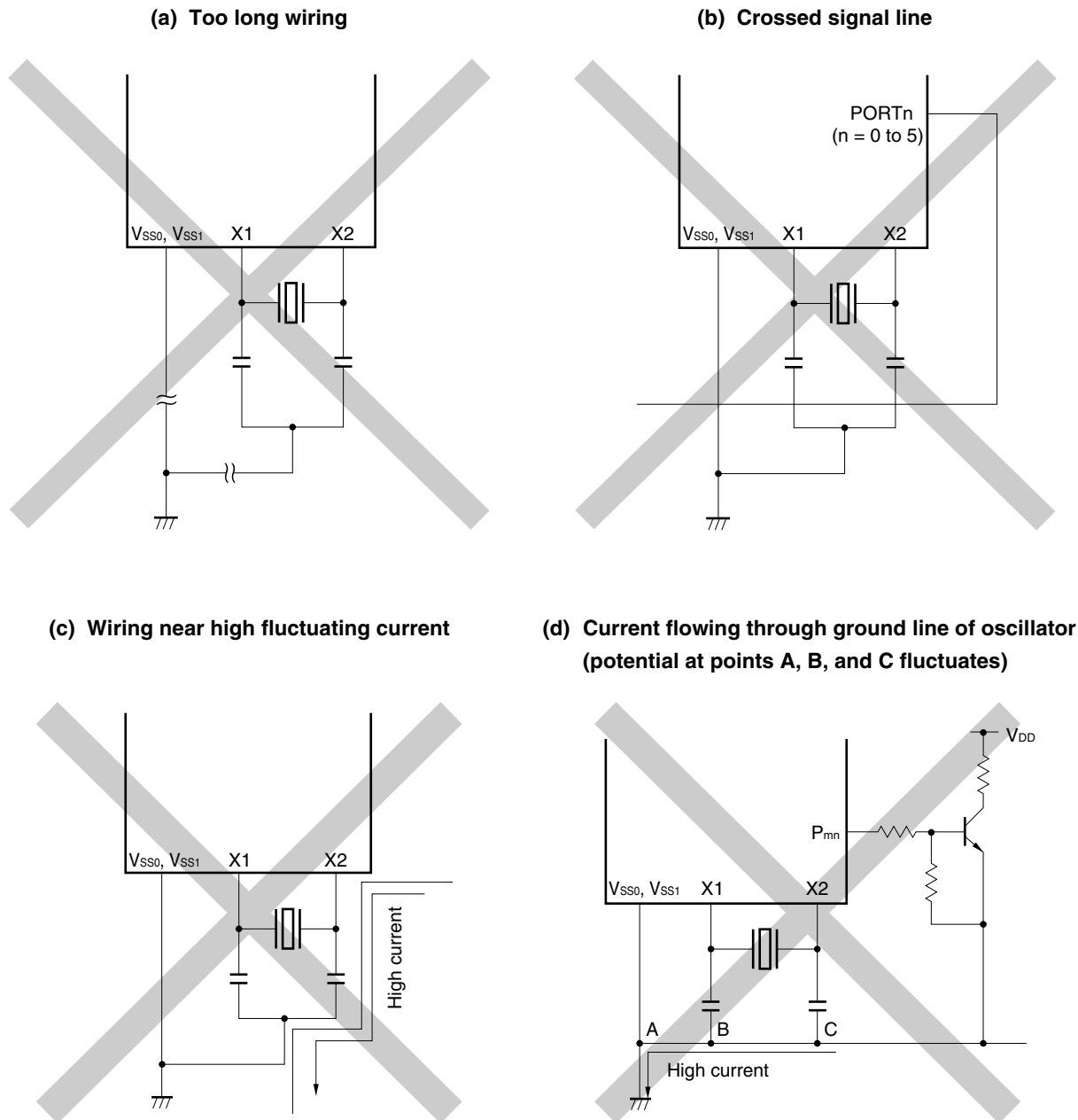
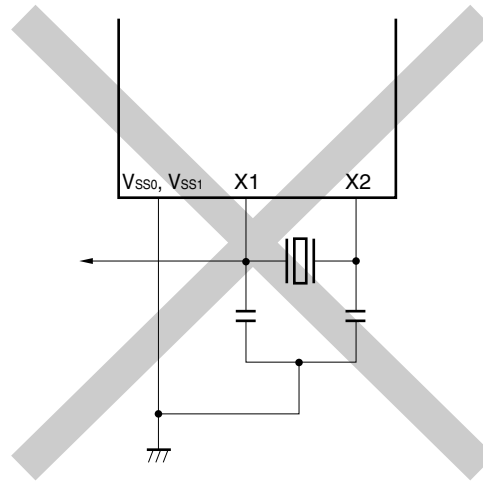


Figure 5-4. Examples of Incorrect Resonator Connection (2/2)

(e) Signal is fetched



#### 5.4.3 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 slow mode (1.6  $\mu s$ : at 5.0 MHz operation) of the system clock is selected when the  $\overline{RESET}$  signal is generated (PCC = 02H). While a low level is being input to the  $\overline{RESET}$  pin, oscillation of the system clock is stopped.
- (b) Two types of minimum instruction execution time (0.4  $\mu s$  and 1.6  $\mu s$ : 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 input clock operation).

## 5.6 Changing Setting of System Clock and CPU Clock

### 5.6.1 Time required for switching between system clock and 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; the old clock is used for the duration of several instructions after that (see **Table 5-2**).

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

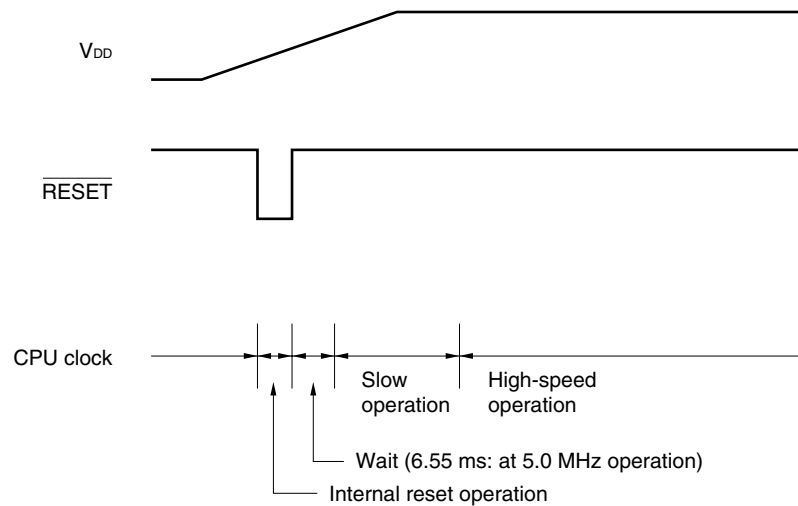
Set Value Before Switching	Set Value After Switching	
PCC1	PCC1	PCC1
	0	1
0		4 clocks
1	2 clocks	

**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  $\overline{RESET}$  pin is made low on power application. The effect of resetting is released when the  $\overline{RESET}$  pin is later made high, and the system clock starts oscillating. At this time, the oscillation stabilization time ( $2^{15}/f_x$ ) is automatically secured.

After that, the CPU starts instruction execution at the slow speed of the system clock (1.6  $\mu$ s: 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 high speed has elapsed, the processor clock control register (PCC) is rewritten so that the high-speed mode can be selected.

### 6.1 Functions of 16-Bit Timer 20

16-bit timer 20 has the following functions.

- Timer interrupt
- Timer output
- Count value capture

#### (1) Timer interrupt

An interrupt is generated when the count value and compare value match.

#### (2) Timer output

Timer output control is possible when the count value and compare value match.

#### (3) Count value capture

The TM20 count value is latched into a capture register in synchronization with the capture trigger and retained.

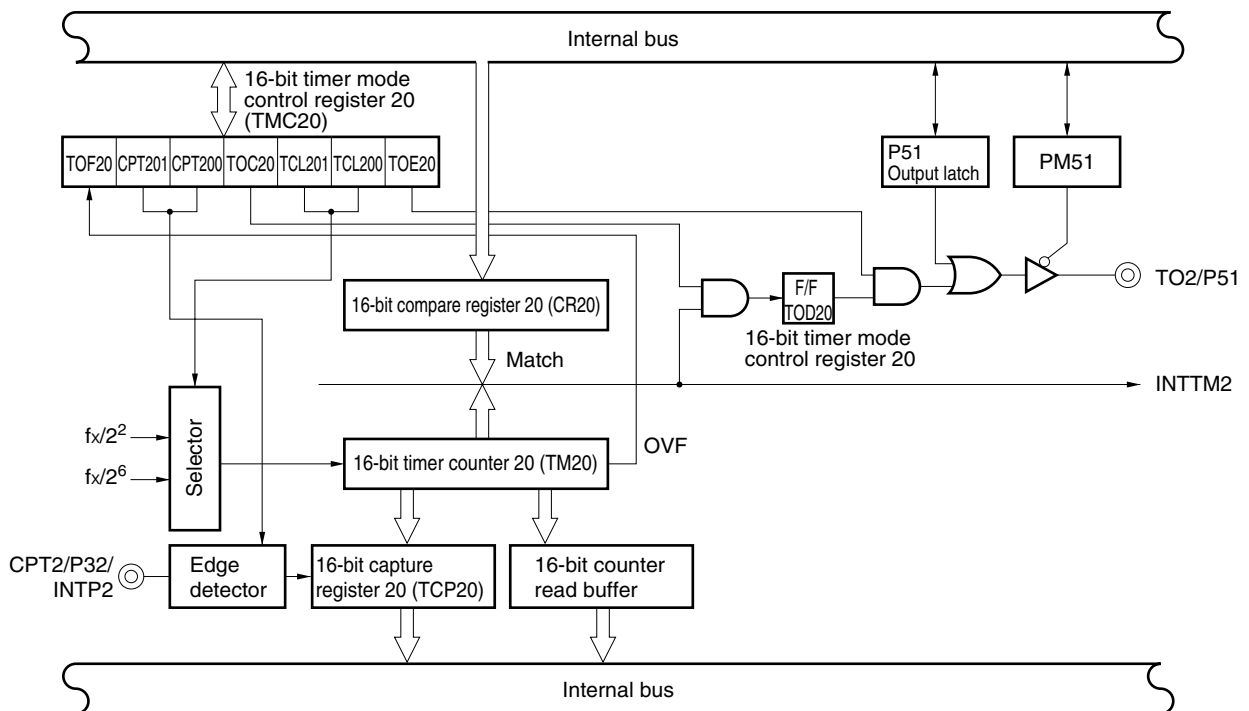
## 6.2 Configuration of 16-Bit Timer 20

16-bit timer 20 consists of the following hardware.

**Table 6-1. Configuration of 16-Bit Timer 20**

Item	Configuration
Timer counter	16 bits × 1 (TM20)
Registers	Compare register: 16 bits × 1 (CR20) Capture register: 16 bits × 1 (TCP20)
Timer outputs	1 (TO2)
Control registers	16-bit timer mode control register 20 (TMC20) Port mode register 5 (PM5)

**Figure 6-1. Block Diagram of 16-Bit Timer 20**



**(1) 16-bit compare register 20 (CR20)**

This register compares the value set to CR20 with the count value of 16-bit timer counter 20 (TM20), and when they match, generates an interrupt request (INTTM2).

CR20 is set with a 16-bit memory manipulation instruction. Values from 0000H to FFFFH can be set.

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

**Cautions** 1. This register is manipulated with a 16-bit memory manipulation instruction; however, an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, the accessing method should be direct addressing. This register can be accessed only in short direct addressing mode when a 16-bit memory manipulation instruction is used.

2. When rewriting CR20 during a count operation, set CR20 to interrupt disabled using interrupt mask flag register 1 (MK1) beforehand. Also, set the timer output data to inversion disabled using 16-bit timer mode control register 20 (TMC20).

If CR20 is rewritten with interrupts enabled, an interrupt request may be issued immediately.

**(2) 16-bit timer counter 20 (TM20)**

This is a 16-bit register that counts count pulses.

TM20 is read with a 16-bit memory manipulation instruction.

This register is in free-running mode during count clock input.

$\overline{\text{RESET}}$  input clears this register to 0000H, after which it enters free-running mode again.

**Cautions** 1. The count value after releasing stop becomes undefined because the count operation is executed during the oscillation stabilization time.

2. This register is manipulated with a 16-bit memory manipulation instruction; however, an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, the accessing method should be direct addressing. This register can be accessed only in short direct addressing mode when a 16-bit memory manipulation instruction is used.

3. When manipulated with an 8-bit memory manipulation instruction, readout should be performed in order from lower byte to higher byte and must be in pairs.

**(3) 16-bit capture register 20 (TCP20)**

This is a 16-bit register that captures the contents of 16-bit timer counter 20 (TM20).

TCP20 is set with a 16-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input makes this register undefined.

**Caution** This register is manipulated with a 16-bit memory manipulation instruction; however, an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, the accessing method should be direct addressing. This register can be accessed only in short direct addressing mode when a 16-bit memory manipulation instruction is used.

**(4) 16-bit counter read buffer**

This buffer latches the counter value and retains the count value of 16-bit timer counter 20 (TM20).



### 6.3 Registers Controlling 16-Bit Timer 20

The following two registers control 16-bit timer 20 (TM20).

- 16-bit timer mode control register 20 (TMC20)
- Port mode register 5 (PM5)

**(1) 16-bit timer mode control register 20 (TMC20)**

16-bit timer mode control register 20 (TMC20) controls the setting of the count clock, capture edge, etc.

TMC20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears TMC20 to 00H.

Figure 6-2. Format of 16-Bit Timer Mode Control Register 20

Symbol	7	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC20	TOD20	TOF20	CPT201	CPT200	TOC20	TCL201	TCL200	TOE20	FF5BH	00H	R/W <sup>Note</sup>

TOD20	Timer output data	
0	Timer output data is 0.	
1	Timer output data is 1.	

TOF20	Overflow flag set	
0	Clear by reset and software	
1	Set by overflow of 16-bit timer	

CPT201	CPT200	Capture edge selection
0	0	Capture operation disabled
0	1	Rising edge of CPT2
1	0	Falling edge of CPT2
1	1	Both edges of CPT2

TOC20	Timer output data inversion control	
0	Inversion disabled	
1	Inversion enabled	

TCL201	TCL200	16-bit timer counter 20 count clock selection
0	0	$f_x/2^2$ (1.25 MHz)
0	1	$f_x/2^6$ (78.1 kHz)
Other than above		Setting prohibited

TOE20	16-bit timer 20 output control	
0	Output disabled (port mode)	
1	Output enabled	

**Note** Bit 7 is read-only.

**Remarks** 1.  $f_x$ : System clock oscillation frequency  
 2. The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

(2) Port mode register 5 (PM5)

This register sets the input/output of port 5 in 1-bit units.  
To use the P51/TO2 pin for timer output, set PM51 and the output latch of P51 to 0.  
PM5 is set with a 1-bit or 8-bit memory manipulation instruction.  
RESET input sets PM5 to FFH.

Figure 6-3. Format of Port Mode Register 5

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM5	1	1	1	1	PM53	PM52	PM51	PM50	FF25H	FFH	R/W

PM51	P51 pin I/O mode selection
0	Output mode (output buffer on)
1	Input mode (output buffer off)

## 6.4 Operation of 16-Bit Timer 20

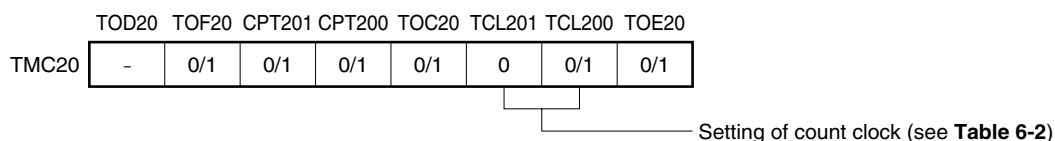
### 6.4.1 Operation as timer interrupt

- ★ 16-bit timer 20 can generate interrupts repeatedly each time the free-running counter value reaches the value set to CR20. Since this counter is not cleared and holds the count even after an interrupt is generated, the interval time is equal to one cycle of the count clock set in TCL201 and TCL200.

To operate 16-bit timer 20 as a timer interrupt, the following settings are required.

- Set count values in CR20
- Set 16-bit timer mode control register 20 (TMC20) as shown in Figure 6-4.

**Figure 6-4. Settings of 16-Bit Timer Mode Control Register 20 for Timer Interrupt Operation**



**Caution** If both the CPT201 and CPT200 flags are set to 0, the operation of the capture edge is disabled.

When the count value of 16-bit timer counter 20 (TM20) matches the value set to CR20, counting of TM20 continues and an interrupt request signal (INTTM2) is generated.

Table 6-2 shows the interval time, and Figure 6-5 shows the timing of the timer interrupt operation.

**Caution** Perform the following processing when rewriting CR20 during a count operation.

- ★
- <1> Disable interrupts (TMMK20 (bit 7 of interrupt mask flag register 1 (MK1)) = 1).
  - <2> Disable inversion control of timer output data (TOC20 = 0).
- If CR20 is rewritten with interrupts enabled, an interrupt request may be generated immediately.

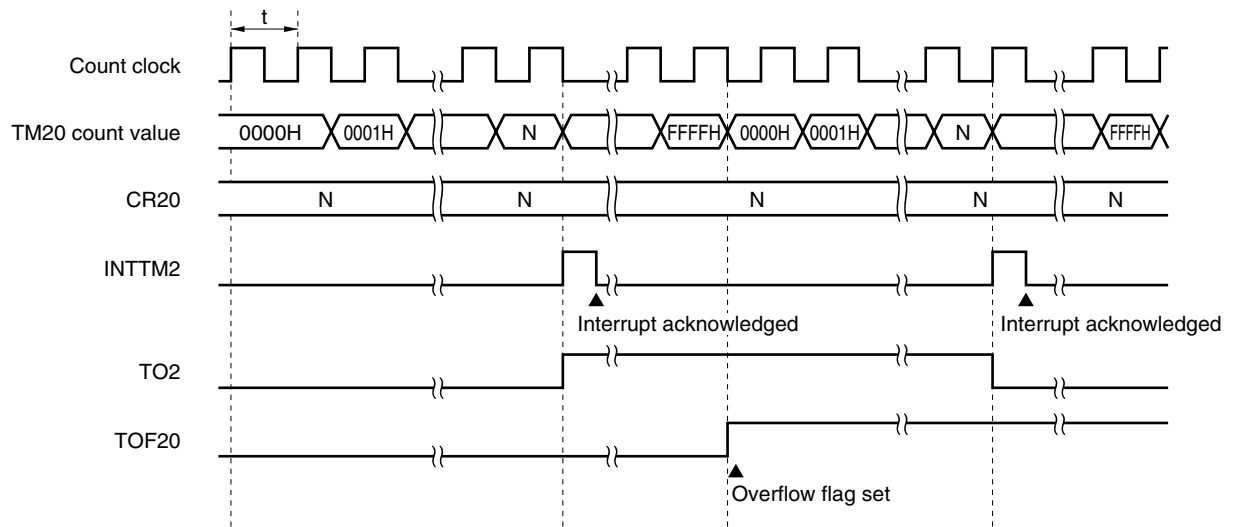
**Table 6-2. Interval Time of 16-Bit Timer 20**

TCL201	TCL200	Count Clock	Interval Time
0	0	$2^2/f_x$ (0.8 $\mu$ s)	$2^{18}/f_x$ (52.4 ms)
0	1	$2^6/f_x$ (12.8 $\mu$ s)	$2^{22}/f_x$ (838.9 ms)
Other than above		Setting prohibited	

**Remarks** 1.  $f_x$ : System clock oscillation frequency

2. The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

Figure 6-5. Timer Interrupt Operation Timing



**Remark** N = 0000H to FFFFH

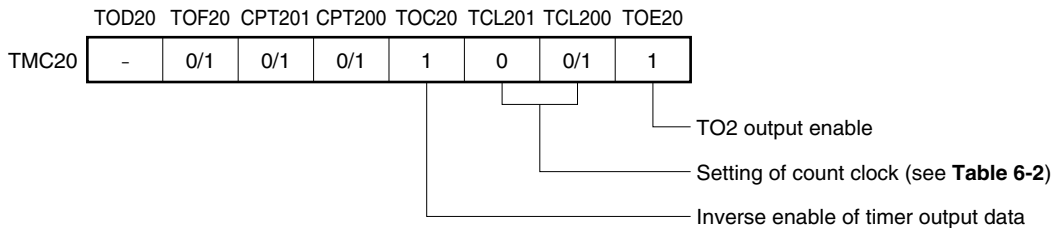
### 6.4.2 Operation as timer output

- ★ 16-bit timer 20 can invert the timer output repeatedly each time the free-running counter value reaches the value set to CR20. Since this counter is not cleared and holds the count even after the timer output is inverted, the interval time is equal to one cycle of the count clock set in TCL201 and TCL200.

To operate 16-bit timer 20 as a timer output, the following settings are required.

- Set P51 to output mode (PM51 = 0).
- Set 0 to the output latch of P51.
- Set the count value in CR20.
- Set 16-bit timer mode control register 20 (TMC20) as shown in Figure 6-6.

**Figure 6-6. Settings of 16-Bit Timer Mode Control Register 20 for Timer Output Operation**

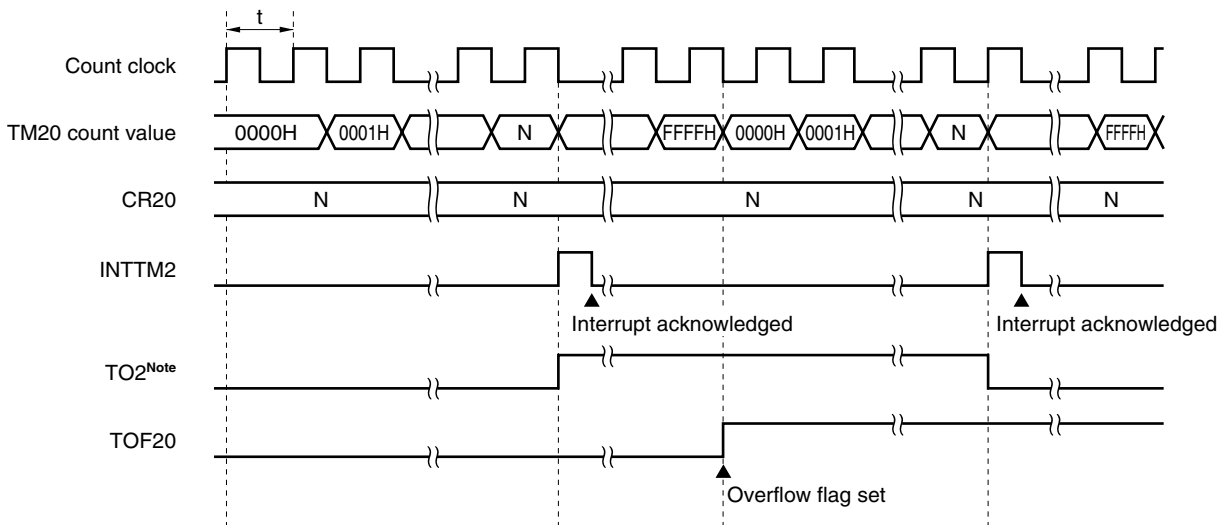


**Caution** If both the CPT201 and CPT200 flags are set to 0, the operation of the capture edge is disabled.

When the count value of 16-bit timer counter 20 (TM20) matches the value set in CR20, the output status of the TO2/P51 pin is inverted. This enables timer output. At that time, TM20 counting continues and an interrupt request signal (INTTM2) is generated.

Figure 6-7 shows the timing of timer output (see Table 6-2 for the interval time of the 16-bit timer).

**Figure 6-7. Timer Output Timing**



**Note** The TO2 initial value becomes low level during output enable (TOE20 = 1).

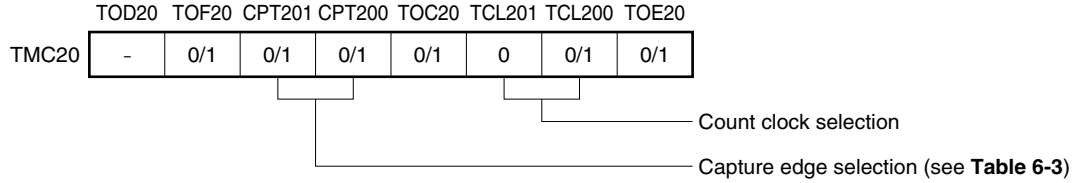
**Remark** N = 0000H to FFFFH

### 6.4.3 Capture operation

The capture operation functions to capture and latch the count value of 16-bit timer counter 20 (TM20) synchronized with a capture trigger.

Set as shown in Figure 6-8 to allow 16-bit timer 20 to start the capture operation.

**Figure 6-8. Settings of 16-Bit Timer Mode Control Register 20 for Capture Operation**



16-bit capture register 20 (TCP20) starts a capture operation after the CPT2 capture trigger edge is detected, and latches and retains the count value of 16-bit timer counter 20. TCP20 fetches the count value within 2 clocks and retains the count value until the next capture edge detection.

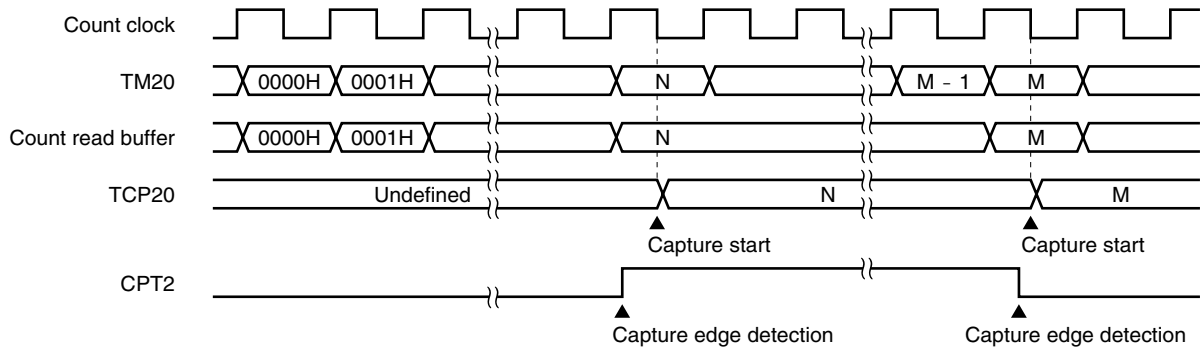
Table 6-3 and Figure 6-9 show the settings of the capture edge and the capture operation timing, respectively.

**Table 6-3. Settings of Capture Edge**

CPT201	CPT200	Capture Edge Selection
0	0	Capture operation disabled
0	1	CPT2 pin rising edge
1	0	CPT2 pin falling edge
1	1	CPT2 pin both edges

**Caution** Because TCP20 is rewritten when a capture trigger edge is detected during TCP20 read, disable capture trigger edge detection during TCP20 read.

**Figure 6-9. Capture Operation Timing (Both Edges of CPT2 Pin Are Specified)**



#### 6.4.4 16-bit timer counter 20 readout

The count value of 16-bit timer counter 20 (TM20) is read out by a 16-bit manipulation instruction.

TM20 readout is performed via a counter read buffer. The counter read buffer latches the TM20 count value. The buffer operation is then held pending at the CPU clock falling edge after the read signal of the TM20 lower byte rises and the count value is retained. The counter read buffer value in the retention state can be read out as the count value.

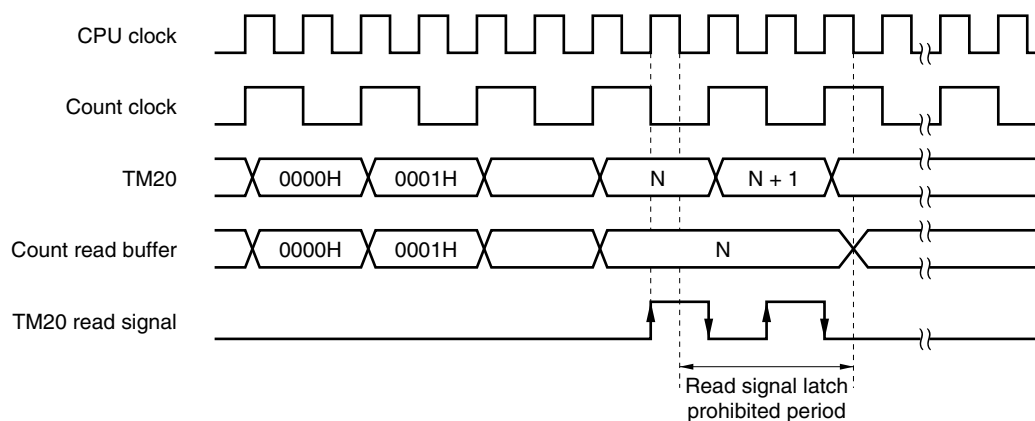
Cancellation of pending is performed at the CPU clock falling edge after the read signal of the TM20 higher byte falls.

$\overline{\text{RESET}}$  input clears TM20 to 0000H and starts freerunning.

Figure 6-10 shows the timing of 16-bit timer counter 20 readout.

- Cautions**
1. The count value after releasing stop becomes undefined because the count operation is executed during the oscillation stabilization time.
  2. Though TM20 is a dedicated 16-bit transfer instruction register, an 8-bit transfer instruction can be used.  
When using an 8-bit transfer instruction, execute by direct addressing.
  3. When using an 8-bit transfer instruction, execute in order from lower byte to higher byte in pairs. If the only lower byte is read, the pending state of the counter read buffer is not canceled, and if the only higher byte is read, an undefined count value is read.

Figure 6-10. Readout Timing of 16-Bit Timer Counter 20





## ★ 6.5 Notes on Using 16-Bit Timer 20

### 6.5.1 Restrictions when rewriting 16-bit compare register 20

- (1) Disable interrupts (TMMK20 = 1) and the inversion control of timer output (TOC20 = 0) before rewriting the compare register (CR20).

If CR20 is rewritten with interrupts enabled, an interrupt request may be generated immediately.

- (2) Depending on the timing of rewriting the compare register (CR20), the interval time may become twice as long as the intended time. Similarly, a shorter waveform or twice-longer waveform than the intended timer output waveform may be output.

To avoid this problem, rewrite the compare register using either of the following procedures.

<Countermeasure A> When rewriting using 8-bit access

<1> Disable interrupts (TMMK20 = 1) and the inversion control of timer output (TOC20 = 0).

<2> First rewrite the higher 1 byte of CR20 (16 bits).

<3> Then rewrite the lower 1 byte of CR20 (16 bits).

<4> Clear the interrupt request flag (TMIF20).

<5> Enable timer interrupts/timer output inversion after half a cycle or more of the count clock has elapsed from the beginning of the interrupt.

<Program example A> (count clock = 64/f<sub>x</sub>, CPU clock = f<sub>x</sub>)

TM90_VCT: SET1 TMMK20	; Disable timer interrupts (6 clocks)	} Total: 32 clocks or more <sup>Note</sup>
CLR1 TMC20.3	; Disable timer output inversion (6 clocks)	
MOV A, #xxH	; Set the rewrite value of higher byte (6 clocks)	
MOV !0FF17H, A	; Rewrite CR20 higher byte (8 clocks)	
MOV A, #yyH	; Set the rewrite value of lower byte (6 clocks)	
MOV !0FF16H, A	; Rewrite CR20 lower byte (8 clocks)	
CLR1 TMIF20	; Clear interrupt request flag (6 clocks)	
CLR1 TMMK20	; Enable timer interrupts (6 clocks)	
SET1 TMC20.3	; Enable timer output inversion	

**Note** Because the INTTM2 signal becomes high level for half a cycle of the count clock after an interrupt is generated, the output is inverted if TOC20 is set to 1 during this period.

<Countermeasure B> When rewriting using 16-bit access

<1> Disable interrupts (TMMK20 = 1) and the inversion control of timer output (TOC20 = 0).

<2> Rewrite CR20 (16 bits).

<3> Wait for one cycle or more of the count clock.

<4> Clear the interrupt request flag (TMIF20).

<5> Enable timer interrupts/timer output inversion.

<Program example B> (count clock =  $64/f_x$ , CPU clock =  $f_x$ )

```

TM20_VCT  SET1  TMMK20      ; Disable timer interrupts
          CLR1  TMC20.3     ; Disable timer output inversion
          MOVW  AX, #xyyyH  ; Set the rewrite value of CR20
          MOVW  CR20, AX    ; Rewrite CR20
          NOP
          NOP               }
          :                 ; 32 NOP instructions (wait for  $64/f_x$ )Note
          NOP
          NOP
          CLR1  TMIF20      ; Clear interrupt request flag
          CLR1  TMMK20      ; Enable timer interrupts
          SET1  TMC20.3     ; Enable timer output inversion

```

**Note** Clear the interrupt request flag (TMIF20) after waiting for one cycle or more of the count clock from the instruction rewriting CR20 (MOVW CR20, AX).

## CHAPTER 7 8-BIT TIMER/EVENT COUNTER 00

### 7.1 Functions of 8-Bit Timer/Event Counter 00

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

- Interval timer
- External event counter
- Square wave output

#### (1) 8-bit interval timer

When 8-bit timer/event counter 00 is used as an interval timer, it generates an interrupt at any time interval set in advance.

**Table 7-1. Interval Time of 8-Bit Timer/Event Counter 00**

Minimum Interval Time	Maximum Interval Time	Resolution
$1/f_x$ (200 ns)	$2^8/f_x$ (51.2 $\mu$ s)	$1/f_x$ (200 ns)
$2^5/f_x$ (6.4 $\mu$ s)	$2^{13}/f_x$ (1.64 ms)	$2^5/f_x$ (6.4 $\mu$ s)

- Remarks**
1.  $f_x$ : System clock oscillation frequency
  2. The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

#### (2) External event counter

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

#### (3) Square wave output

A square wave of arbitrary frequency can be output.

**Table 7-2. Square Wave Output Range of 8-Bit Timer/Event Counter 00**

Minimum Pulse Width	Maximum Pulse Width	Resolution
$1/f_x$ (200 ns)	$2^8/f_x$ (51.2 $\mu$ s)	$1/f_x$ (200 ns)
$2^5/f_x$ (6.4 $\mu$ s)	$2^{13}/f_x$ (1.64 ms)	$2^5/f_x$ (6.4 $\mu$ s)

- Remarks**
1.  $f_x$ : System clock oscillation frequency
  2. The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

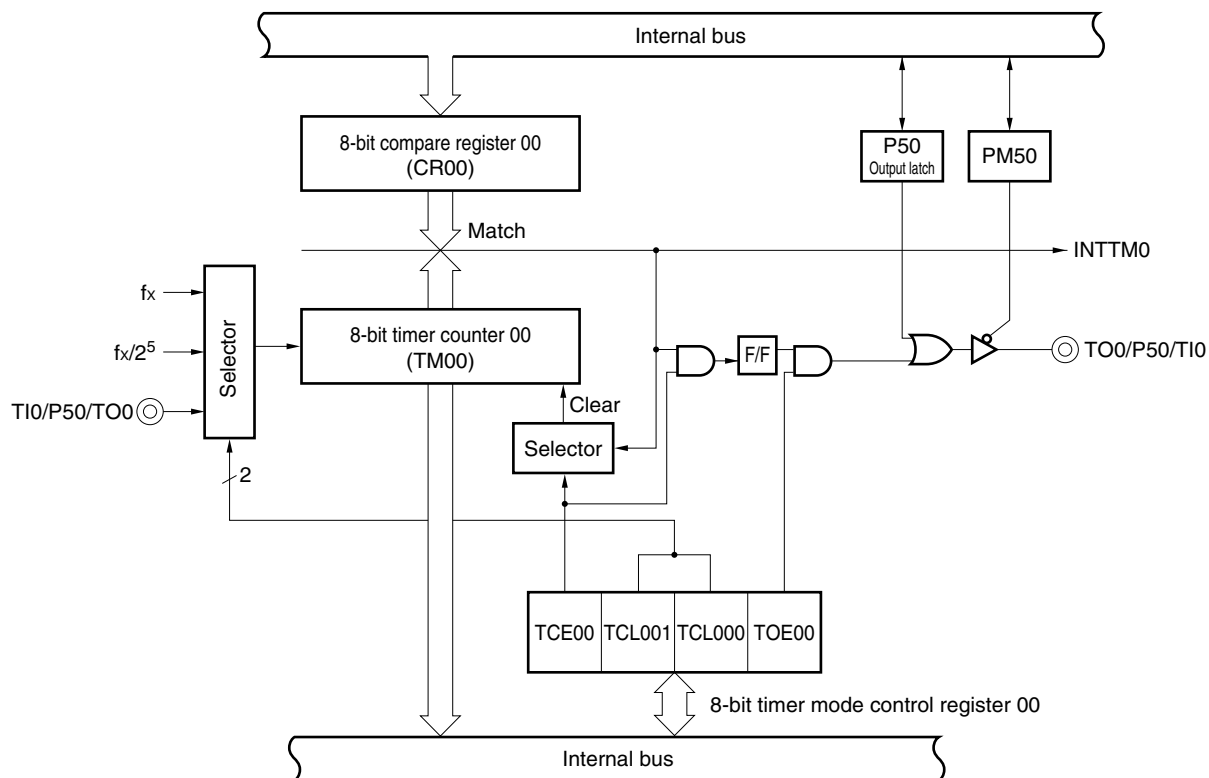
## 7.2 Configuration of 8-Bit Timer/Event Counter 00

8-bit timer/event counter 00 consists of the following hardware.

**Table 7-3. Configuration of 8-Bit Timer/Event Counter 00**

Item	Configuration
Timer counter	8 bits × 1 (TM00)
Register	Compare register: 8 bits × 1 (CR00)
Timer outputs	1 (TO0)
Control registers	8-bit timer mode control register 00 (TMC00) Port mode register 5 (PM5)

**Figure 7-1. Block Diagram of 8-Bit Timer/Event Counter 00**



### (1) 8-bit compare register 00 (CR00)

This is an 8-bit register that compares the value set to CR00 with the 8-bit timer register 00 (TM00) count value, and if they match, generates an interrupt request (INTTM0).

CR00 is written with an 8-bit memory manipulation instruction. Values from 00H to FFH can be set.

$\overline{\text{RESET}}$  input makes CR00 undefined.

**Caution** Before rewriting CR00, stop the timer operation. If CR00 is rewritten while the timer operation is enabled, the match interrupt request signal may be generated immediately.

### (2) 8-bit timer counter 00 (TM00)

This is an 8-bit register that counts pulses.

TM00 is read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears TM00 to 00H.

### 7.3 Registers Controlling 8-Bit Timer/Event Counter 00

The following two registers are used to control 8-bit timer/event counter 00.

- 8-bit timer mode control register 00 (TMC00)
- Port mode register 5 (PM5)

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

TMC00 determines whether to enable or disable 8-bit timer counter 00 (TM00), specifies the count clock for TM00, and controls the operation of the output controller of 8-bit timer/event counter 00.

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

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

**Figure 7-2. Format of 8-Bit Timer Mode Control Register 00**

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 counter 00 operation control										
0	Operation disabled (TM00 is cleared to 0)										
1	Operation enabled										

TCL001	TCL000	8-bit timer counter 00 count clock selection									
0	0	fx (5.0 MHz)									
0	1	fx/2 <sup>5</sup> (156 kHz)									
1	0	Rising edge of TIO <sup>Note</sup>									
1	1	Falling edge of TIO <sup>Note</sup>									

TOE00	8-bit timer/event counter 00 output control										
0	Output disabled (port mode)										
1	Output enabled										

**Note** When inputting a clock signal externally, timer output cannot be used.

**Caution** Always stop the timer before setting TMC00.

**Remarks**

1. fx: System clock oscillation frequency
2. The parenthesized values apply to operation at fx = 5.0 MHz.

**(2) Port mode register 5 (PM5)**

This register sets port 5 input/output in 1-bit units.

When using the P50/TI0/TO0 pin for timer output, set PM50 and the output latch of P50 to 0.

PM5 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input sets PM5 to FFH.

**Figure 7-3. Format of Port Mode Register 5**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM5	1	1	1	1	PM53	PM52	PM51	PM50	FF25H	FFH	R/W

PM50	P50 pin I/O mode selection
0	Output mode (output buffer on)
1	Input mode (output buffer off)

## 7.4 Operation of 8-Bit Timer/Event Counter 00

### 7.4.1 Operation as interval timer

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

To operate 8-bit timer/event counter 00 as an interval timer, the following settings are required.

- <1> Disable the operation of 8-bit timer counter 00 (TM00) (TCE00 (bit 7 of 8-bit timer mode control register 00 (TMC00)) = 0).
- <2> Set the count clock of 8-bit timer/event counter 00 (see **Table 7-4**).
- <3> Set a count value in CR00.
- <4> Enable the operation of TM00 (TCE00 = 1).

When the count value of 8-bit timer counter 00 (TM00) matches the value set to CR00, the value of TM00 is cleared to 0 and TM00 continues counting. At the same time, an interrupt request signal (INTTM0) is generated.

Table 7-4 shows the interval time, and Figure 7-4 shows the timing of interval timer operation.

- Cautions**
1. Before rewriting CR00, stop the timer operation. If CR00 is rewritten while the timer operation is enabled, the match interrupt request signal may be generated immediately.
  2. If setting the count clock in TMC00 and enabling the operation of TM00 are performed at the same time with an 8-bit memory manipulation instruction, the error one cycle after the timer has been started may exceed one clock. To use 8-bit timer/event counter 00 as an interval timer, therefore, perform the setting in the above sequence.

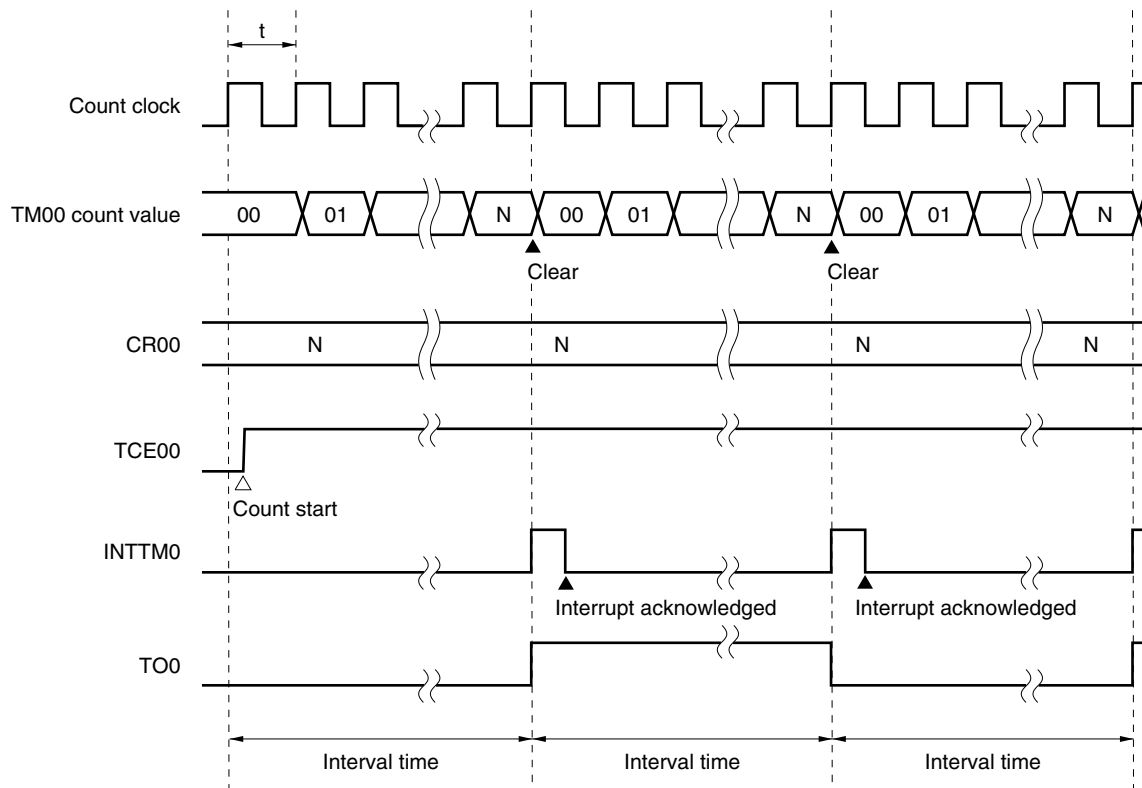
**Table 7-4. Interval Time of 8-Bit Timer/Event Counter 00**

TCL001	TCL000	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	1/fx (200 ns)	2 <sup>8</sup> /fx (51.2 μs)	1/fx (200 ns)
0	1	2 <sup>5</sup> /fx (6.4 μs)	2 <sup>13</sup> /fx (1.64 ms)	2 <sup>5</sup> /fx (6.4 μs)
1	0	Ti0 input cycle	2 <sup>8</sup> × Ti0 input cycle	Ti0 input edge cycle
1	1	Ti0 input cycle	2 <sup>8</sup> × Ti0 input cycle	Ti0 input edge cycle

**Remarks** 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 5.0 MHz.

Figure 7-4. Interval Timer Operation Timing



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



### 7.4.2 Operation as external event counter

The external event counter counts the number of external clock pulses input to the TIO/P50/TO0 pin by using timer counter 00 (TM00).

To operate 8-bit timer/event counter 00 as an external event counter, the following settings are required.

- <1> Set P50 to input mode (PM50 = 1).
- <2> Disable the operation of 8-bit timer counter 00 (TM00) (TCE00 (bit 7 of 8-bit timer mode control register 00 (TMC00)) = 0).
- <3> Specify the rising or falling edge of TIO (see **Table 7-4**).
- <4> Set a count value in CR00.
- <5> Enable the operation of TM00 (TCE00 = 1).

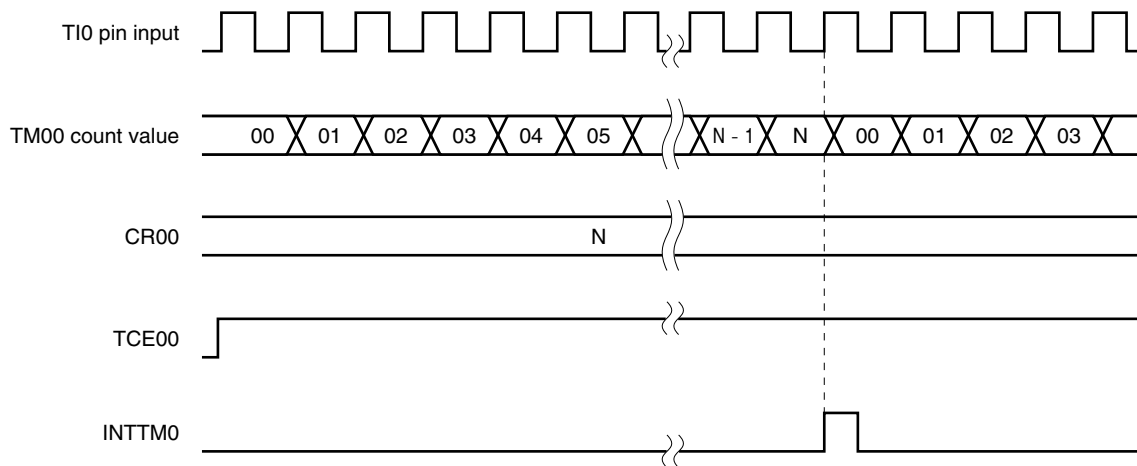
Each time the valid edge specified by the bit 1 or 2 (TCL001 or TCL000) of TMC00 is input, the value of 8-bit timer counter 00 (TM00) is incremented.

When the count value of TM00 matches the value set to CR00, the value of TM00 is cleared to 0 and TM00 continues counting. At the same time, an interrupt request signal (INTTM0) is generated.

Figure 7-5 shows the timing of external event counter operation (with rising edge specified).

- Cautions**
1. Before rewriting CR00, stop the timer operation. If CR00 is rewritten while the timer operation is enabled, the match interrupt request signal may be generated immediately.
  2. If setting the count clock in TMC00 and enabling the operation of TM00 are performed at the same time with an 8-bit memory manipulation instruction, the error one cycle after the timer has been started may exceed one clock. To use 8-bit timer/event counter 00 as an external event counter, therefore, perform the setting in the above sequence.

**Figure 7-5. External Event Counter Operation Timing (with Rising Edge Specified)**



**Remark** N = 00H to FFH

### 7.4.3 Operation as square wave output

8-bit timer/event counter 00 can generate a square wave output of arbitrary frequency at intervals specified by the count value set to 8-bit compare register 00 (CR00) in advance.

To operate 8-bit timer/event counter 00 as a square wave output, the following settings are required.

- <1> Set P50 to output mode (PM50 = 0).
- <2> Set 0 to the output latch of P50.
- <3> Disable the operation of 8-bit timer counter 00 (TM00) (TCE00 (bit 7 of 8-bit timer mode control register 00 (TMC00)) = 1).
- <4> Set the count clock for 8-bit timer/event counter 00 and enable output of TO0 (TOE00 (bit 0 of TMC00) = 1).
- <5> Set a count value in CR00.
- <6> Enable the operation of TM00 (TCE00 = 1).

When the count value of 8-bit timer counter 00 (TM00) matches the value set in CR00, the TO0/P50/TI0 pin output is inverted. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, the TM00 value is cleared to 0 then TM00 resumes counting, generating an interrupt request signal (INTTM0).

Setting bit 7 in TMC00 (TCE00) to 0 clears the square-wave output to 0.

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

- Cautions**
1. Before rewriting CR00, stop the timer operation. If CR00 is rewritten while the timer operation is enabled, the match interrupt request signal may be generated immediately.
  2. If setting the count clock in TMC00 and enabling the operation of TM00 are performed at the same time with an 8-bit memory manipulation instruction, the error one cycle after the timer has been started may exceed one clock. To use 8-bit timer/event counter 00 as a square wave output, therefore, perform the setting in the above sequence.

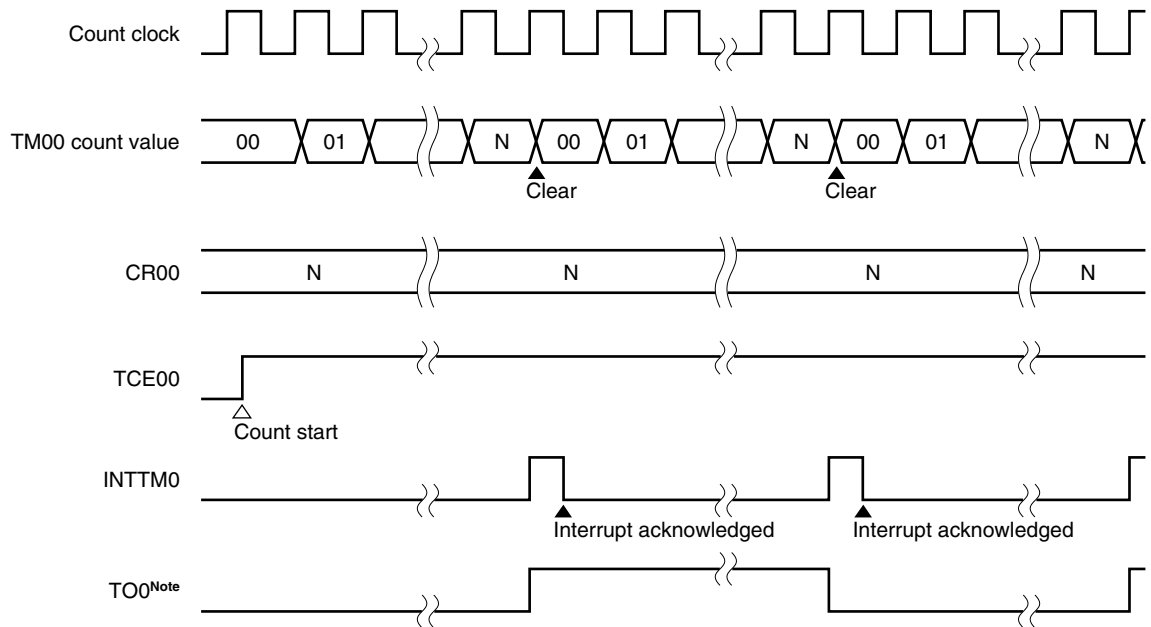
**Table 7-5. Square Wave Output Range of 8-Bit Timer/Event Counter 00**

TCL001	TCL000	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	1/f <sub>x</sub> (200 ns)	2 <sup>8</sup> /f <sub>x</sub> (51.2 μs)	1/f <sub>x</sub> (200 ns)
0	1	2 <sup>5</sup> /f <sub>x</sub> (6.4 μs)	2 <sup>13</sup> /f <sub>x</sub> (1.64 ms)	2 <sup>5</sup> /f <sub>x</sub> (6.4 μs)

**Remarks** 1. f<sub>x</sub>: System clock oscillation frequency

2. The parenthesized values apply to operation at f<sub>x</sub> = 5.0 MHz.

Figure 7-6. Square Wave Output Timing



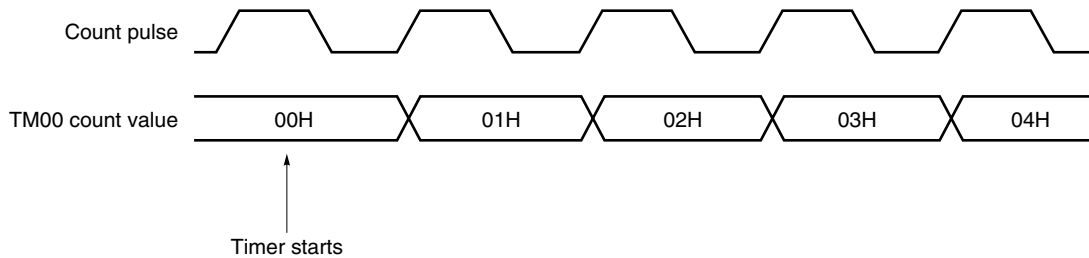
**Note** The initial value of TO0 at output enable (TOE00 = 1) becomes low level.

## 7.5 Notes on Using 8-Bit Timer/Event Counter 00

### (1) Error on starting timer

An error of up to 1 clock occurs after the timer has been started until a match signal is generated. This is because 8-bit timer counter 00 (TM00) is started asynchronously to the count pulse.

**Figure 7-7. Start Timing of 8-Bit Timer Counter 00**

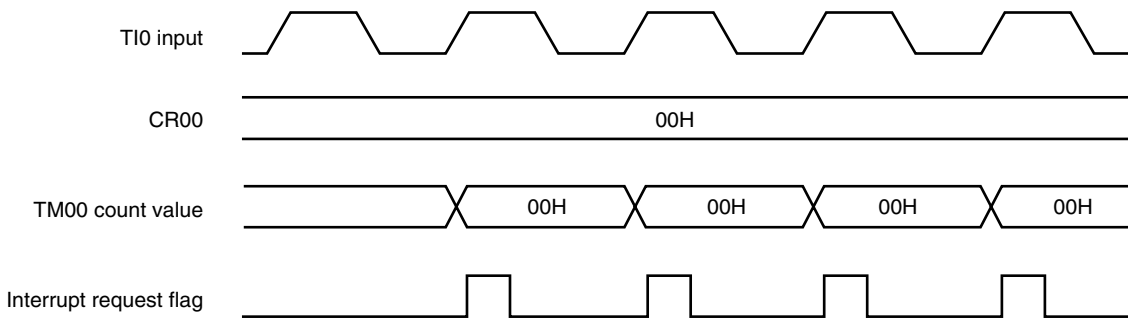


### (2) Setting of 8-bit compare register

8-bit compare register 00 (CR00) can be set to 00H.

Therefore, one pulse can be counted when 8-bit timer/event counter 00 operates as an event counter.

**Figure 7-8. One Pulse Count Operation Timing**



## CHAPTER 8 WATCHDOG TIMER

### 8.1 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 inadvertent program loops. When an inadvertent loop is detected, a non-maskable interrupt or the  $\overline{\text{RESET}}$  signal can be generated.

**Table 8-1. Inadvertent Loop Detection Time of Watchdog Timer**

Inadvertent Loop Detection Time	At $f_x = 5.0 \text{ MHz}$
$2^{11} \times 1/f_x$	410 $\mu\text{s}$
$2^{13} \times 1/f_x$	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 8-2. Interval Time**

Interval	At $f_x = 5.0 \text{ MHz}$
$2^{11} \times 1/f_x$	410 $\mu\text{s}$
$2^{13} \times 1/f_x$	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

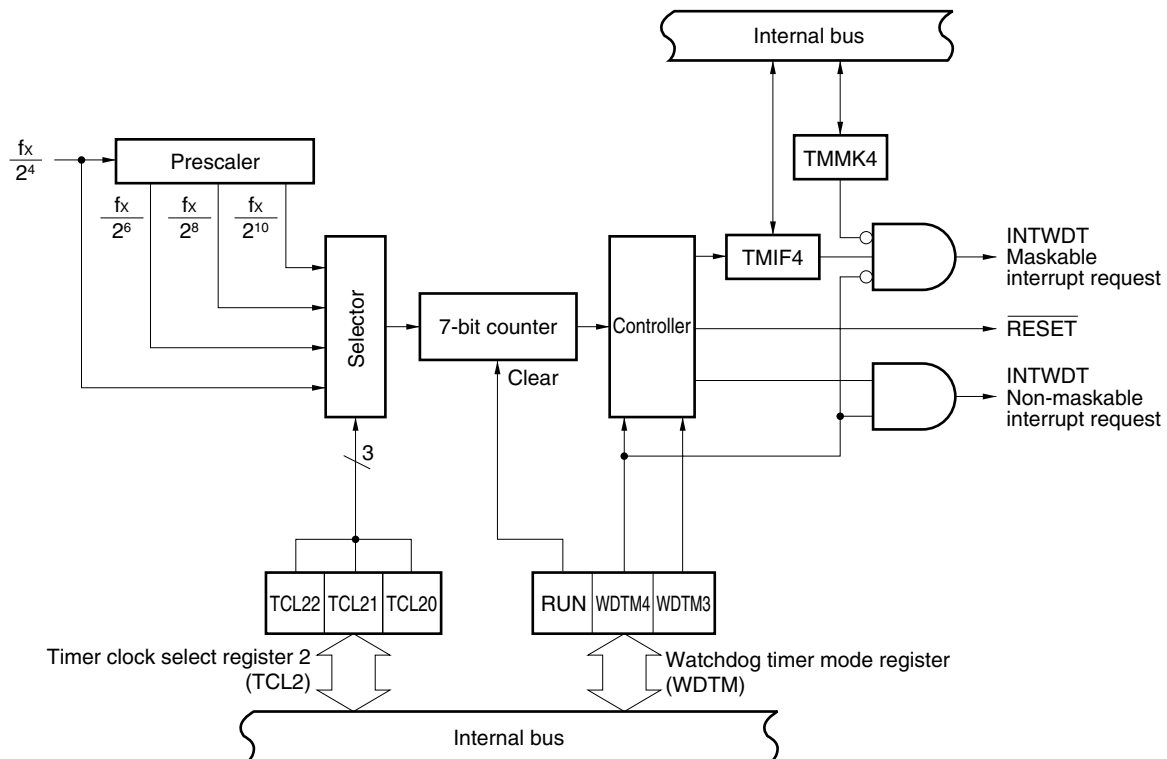
## 8.2 Configuration of Watchdog Timer

The watchdog timer consists of the following hardware.

**Table 8-3. Configuration of Watchdog Timer**

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

**Figure 8-1. Block Diagram of Watchdog Timer**



### 8.3 Registers Controlling Watchdog Timer

The following two 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 clears TCL2 to 00H.

**Figure 8-2. Format of Timer Clock Select Register 2**

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 $\mu\text{s}$ )
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.** The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

**(2) Watchdog timer mode register (WDTM)**

This register sets the 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 clears WDTM to 00H.

**Figure 8-3. Format of Watchdog Timer Mode Register**

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	Watchdog timer operation selection <sup>Note 1</sup>
0	Stop counting.
1	Clear counter and start counting.

WDTM4	WDTM3	Watchdog timer operation mode selection <sup>Note 2</sup>
0	0	Operation stop
0	1	Interval timer mode (overflow and maskable interrupt occur) <sup>Note 3</sup>
1	0	Watchdog timer mode 1 (overflow and non-maskable interrupt occur)
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 operation as an interval timer when RUN is set to 1.

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

**2.** In watchdog timer mode 1 or 2, set WDTM4 to 1 after confirming that TMIF4 (bit 0 of interrupt request flag register 0 (IF0)) is 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.



## 8.4 Operation of Watchdog Timer

### 8.4.1 Operation as watchdog timer

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

The count clock (inadvertent loop detection time interval) of the watchdog timer can be selected by bits 0 to 2 (TCL20 to TCL22) of 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 inadvertent loop 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 inadvertent loop 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 inadvertent loop detection time may be up to 0.8% shorter than the set time.

**Table 8-4. Inadvertent Loop Detection Time of Watchdog Timer**

TCL22	TCL21	TCL20	Inadvertent Loop Detection Time	At $f_x = 5.0 \text{ MHz}$
0	0	0	$2^{11} \times 1/f_x$	410 $\mu\text{s}$
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

### 8.4.2 Operation as interval timer

When bits 4 and 3 (WDTM4, WDTM3) of the watchdog timer mode register (WDTM) are set to 0 and 1 respectively, 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 the count clock (or interval time) by setting bits 0 to 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 8-5. Interval Time of Interval Timer**

TCL22	TCL21	TCL20	Interval Time	At $f_x = 5.0 \text{ MHz}$
0	0	0	$2^{11} \times 1/f_x$	410 $\mu\text{s}$
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 9 SERIAL INTERFACE 00

### 9.1 Functions of Serial Interface 00

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 UART-dedicated baud rate generator is incorporated, allowing communication over a wide range of baud rates. In addition, the baud rate can be defined by dividing the clock input 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 the serial clock ( $\overline{\text{SCK0}}$ ) and two for serial data (SI0, SO0).

The 3-wire serial I/O mode supports simultaneous transmit and receive operations, 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 a conventional on-chip synchronous serial interface.

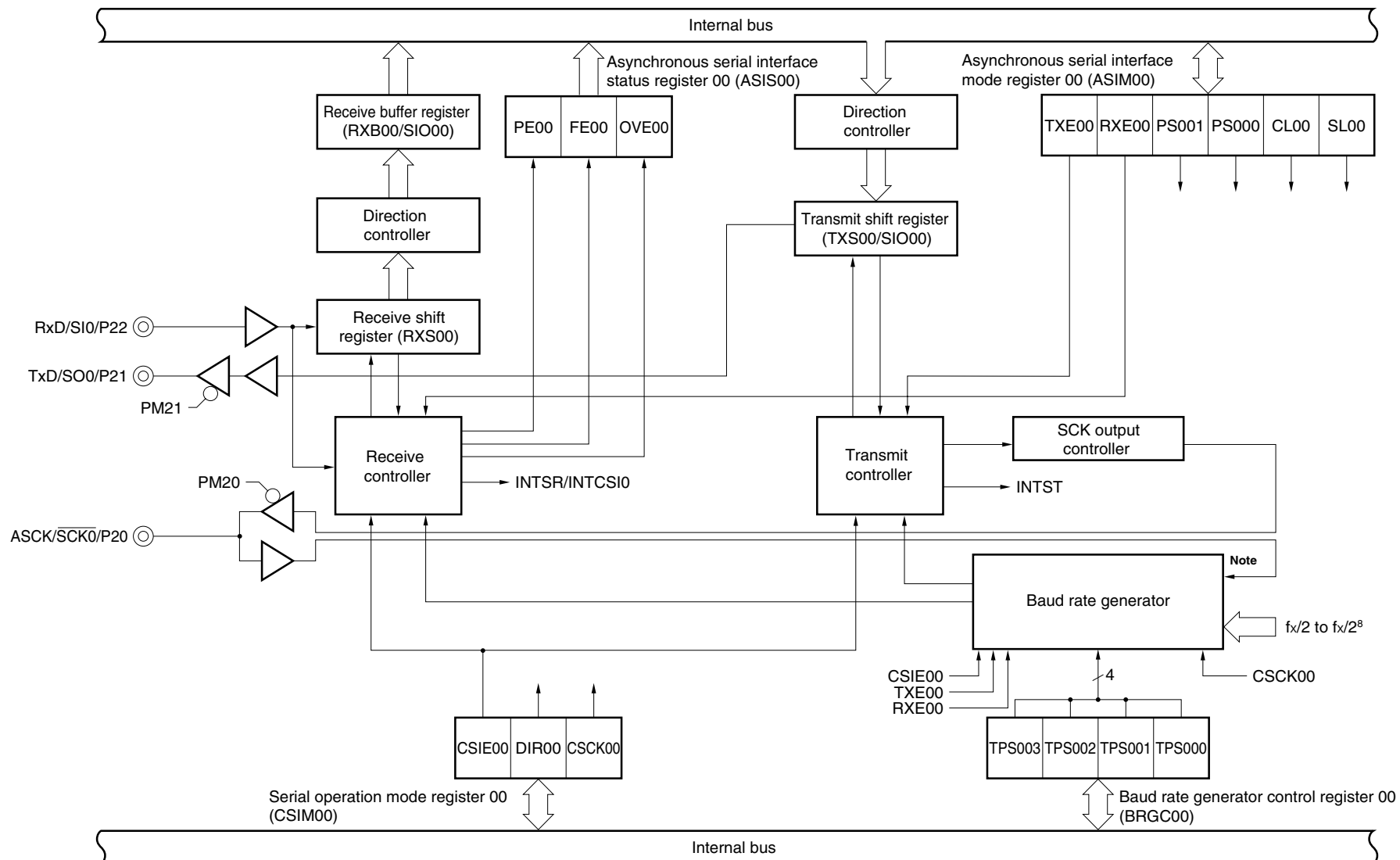
### 9.2 Configuration of Serial Interface 00

Serial interface 00 consists of the following hardware.

**Table 9-1. Configuration of Serial Interface 00**

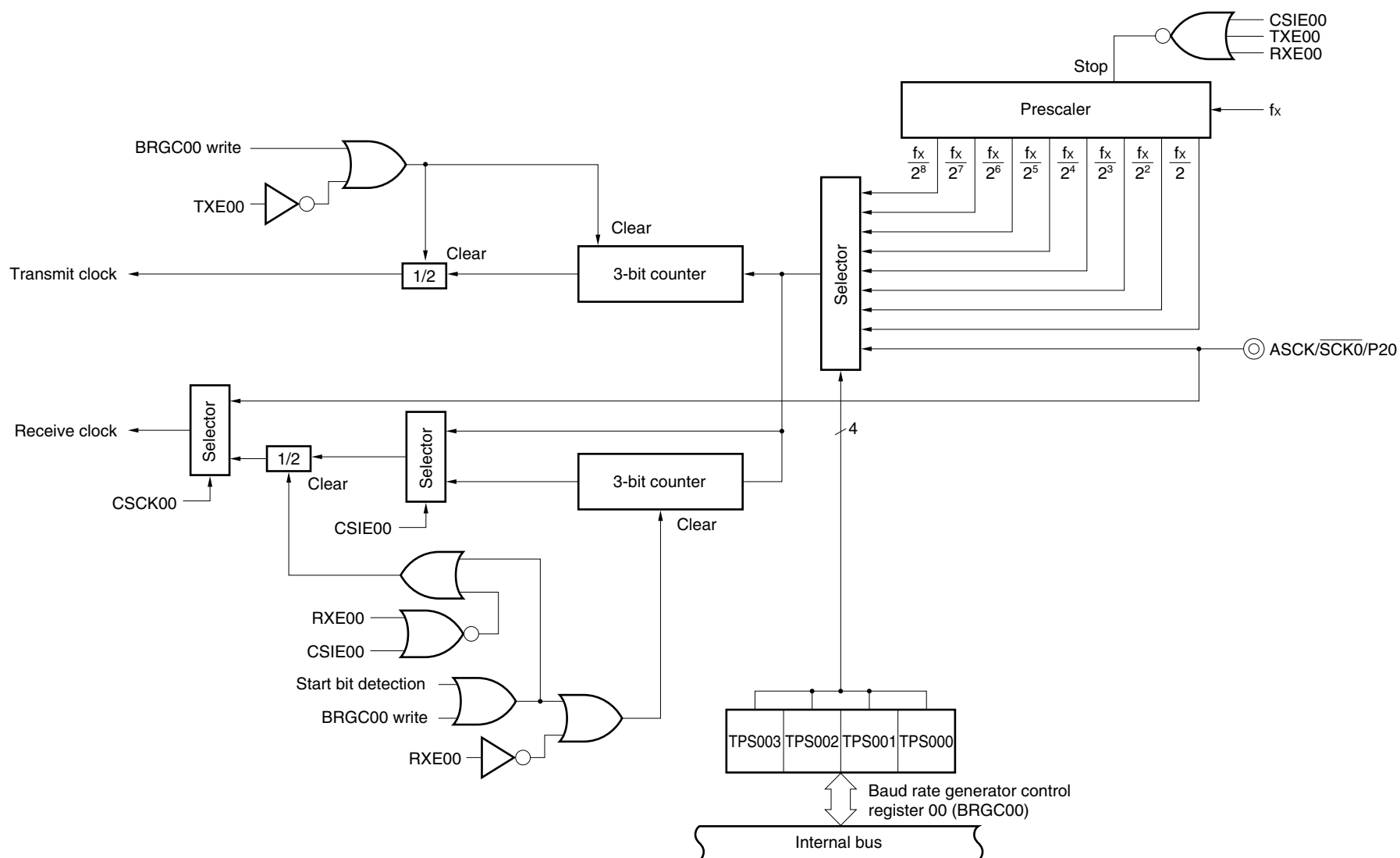
Item	Configuration
Registers	Transmit shift register 00 (TXS00) Receive shift register 00 (RXS00) Receive buffer register 00 (RXB00)
Control registers	Serial operation mode register 00 (CSIM00) Asynchronous serial interface mode register 00 (ASIM00) Asynchronous serial interface status register 00 (ASIS00)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

Figure 9-1. Block Diagram of Serial Interface 00



**Note** For the configuration of the baud rate generator, see Figure 9-2.

Figure 9-2. 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 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 receive buffer register 00 (RXB00).

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 byte of data is transferred from 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 makes RXB00 undefined.

**Caution** RXB00 and transmit shift register 00 (TXS00) are allocated to the same address, and when writing is performed, the values are written to TXS00.

**(4) Transmit controller**

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

**(5) Receive controller**

This circuit controls receive operations according to the data set to 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 asynchronous serial interface status register 00 (ASIS00) depending on the nature of the error.

### 9.3 Registers Controlling Serial Interface 00

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

- Serial operation 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 operation mode register 00 (CSIM00)

This register is set when using 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 clears CSIM00 to 00H.

**Figure 9-3. Format of Serial Operation Mode Register 00**

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 stopped
1	Operation enabled

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 bits 0 and 3 to 6 to 0.
  2. Set CSIM00 to 00H in the UART mode.

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

This register is set when using serial interface 00 in the asynchronous serial interface mode.

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

RESET input clears ASIM00 to 00H.

**Figure 9-4. Format of Asynchronous Serial Interface Mode Register 00**

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 stopped	
1	Transmit operation enabled	

RXE00	Receive operation control	
0	Receive operation stopped	
1	Receive operation enabled	

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 occurs)
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 bits 0 and 1 to 0.
  2. Set ASIM00 to 00H in the 3-wire serial I/O mode.
  3. Switching operating modes must be performed after the serial transmit/receive operation has been stopped.



Table 9-2. Operating Mode Settings of Serial Interface 00

## (1) Operation stop mode

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

## (2) 3-wire serial I/O mode

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

## (3) Asynchronous serial interface mode

ASIM00		CSIM00			PM22	P22	PM21	P21	PM20	P20	Start Bit	Shift Clock	P22/SI0/RxD	P21/SO0/TxD	P20/SCK0/ASCK
TXE00	RXE00	CSIE00	DIR00	CSCK00									Pin Function	Pin Function	Pin Function
1	0	0	0	0	× <sup>Note 1</sup>	× <sup>Note 1</sup>	0	1	1	×	LSB	External clock	P22	TxD (CMOS output)	ASCK input
									× <sup>Note 1</sup>	× <sup>Note 1</sup>		Internal clock			P20
0	1	0	0	0	1	×	× <sup>Note 1</sup>	× <sup>Note 1</sup>	1	×		External clock	RxD	P21	ASCK input
									× <sup>Note 1</sup>	× <sup>Note 1</sup>		Internal clock			P20
1	1	0	0	0	1	×	0	1	1	×		External clock		TxD (CMOS output)	ASCK input
									× <sup>Note 1</sup>	× <sup>Note 1</sup>		Internal clock			P20
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 I/O).

**Remark** ×: Don't care

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

This register indicates the type of error when a reception error occurs in the asynchronous serial 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 clears ASIS00 to 00H.

**Figure 9-5. Format of Asynchronous Serial Interface Status Register 00**

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 did not occur
1	Parity error occurred (when the transmit parity did not match the receive parity)

★

FE00	Framing error flag
0	Framing error did not occur
1	Framing error occurred (when stop bit was not detected) <sup>Note 1</sup>

OVE00	Overrun error flag
0	Overrun error did not occur
1	Overrun error occurred <sup>Note 2</sup> (when the next receive operation was completed before the data was 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 asynchronous serial interface mode register 00 (ASIM00), stop bit detection in the case of reception is performed with 1 bit.

2. When an overrun error occurs, be sure to read out 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 clears BRGC00 to 00H.

**Figure 9-6. Format of Baud Rate Generator Control Register 00**

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	Clock input from external to ASCK pin <sup>Note</sup>	–
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 a communication operation.
  2. Be sure not to select  $n = 1$  during operation at  $f_x = 5.0$  MHz because the  $n = 1$  setting exceeds the baud rate limit.
  3. When selecting the clock input from an external source, set port mode register 2 (PM2) to the input mode.

- Remarks**
1.  $f_x$ : System clock oscillation frequency
  2.  $n$ : Value determined by setting TPS000 to TPS003 ( $1 \leq n \leq 8$ )
  3. The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

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

**(a) Generation of baud rate transmit/receive clock from system clock**

The transmit/receive clock is generated by dividing 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 determined by values of TPS000 to TPS003 as shown in Figure 9-6 ( $2 \leq n \leq 8$ )

**Table 9-3. Example of Relationship Between System Clock and Baud Rate**

Baud Rate (bps)	n	BRGC00 Set Value	Error (%)	
			$f_x = 5.0 \text{ MHz}$	$f_x = 4.9152 \text{ MHz}$
1,200	8	70H	1.73	0
2,400	7	60H		
4,800	6	50H		
9,600	5	40H		
19,200	4	30H		
38,400	3	20H		
76,800	2	10H		

**Caution** Be sure not to select  $n = 1$  during operation at  $f_x = 5.0 \text{ MHz}$  because the  $n = 1$  setting exceeds the baud rate limit.

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

The transmit/receive clock is generated by dividing 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.

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

$f_{\text{ASCK}}$ : Frequency of clock input to the ASCK pin

**Table 9-4. 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
1,200	19.2
2,400	38.4
4,800	76.8
9,600	153.6
19,200	307.2
31,250	500.0
38,400	614.4

## 9.4 Operation of Serial Interface 00

Serial interface 00 provides the following three modes.

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

### 9.4.1 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 operation mode register 00 (CSIM00) and asynchronous serial interface mode register 00 (ASIM00).

##### (a) Serial operation mode register 00 (CSIM00)

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

$\overline{\text{RESET}}$  input clears 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 stopped
1	Operation enabled

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

**(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 clears 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 stopped
1	Transmit operation enabled

RXE00	Receive operation control
0	Receive operation stopped
1	Receive operation enabled

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

### 9.4.2 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 the desired transfer rate from many options. In addition, the baud rate can also be defined by dividing the clock input 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 operation 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 operation mode register 00 (CSIM00)

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

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

Set CSIM00 to 00H 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	CSCCK00	0	FF72H	00H	R/W

CSIE00	Operation control in 3-wire serial I/O mode
0	Operation stopped
1	Operation enabled

DIR00	Start bit specification
0	MSB
1	LSB

CSCCK00	Clock selection in 3-wire serial I/O mode
0	Clock input to $\overline{\text{SCK0}}$ pin from external
1	Dedicated baud rate generator output

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



**(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 clears 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 stopped
1	Transmit operation enabled

RXE00	Receive operation control
0	Receive operation stopped
1	Receive operation enabled

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 occurs)
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 bits 0 and 1 to 0.
  2. Switching operating modes must be performed after the serial transmit/receive operation has been stopped.

**(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 clears 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 did not occur
1	Parity error occurred (when the transmit parity did not match the receive parity)

FE00	Framing error flag
0	Framing error did not occur
1	Framing error occurred (when stop bit was not detected) <sup>Note 1</sup>

OVE00	Overrun error flag
0	Overrun error did not occur
1	Overrun error occurred <sup>Note 2</sup> (when the next receive operation was completed before the data was 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 asynchronous serial interface mode register 00 (ASIM00), stop bit detection in the case of reception is performed with 1 bit.
  2. When an overrun error occurs, be sure to read out 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 clears 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	Clock input from external to ASCK pin	–
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 a communication operation.
- 2.** Be sure not to select  $n = 1$  during operation at  $f_x = 5.0$  MHz because the  $n = 1$  setting exceeds the baud rate limit.
- 3.** When selecting the clock input from an external source, set port mode register 2 (PM2) to the input mode.

- Remarks 1.**  $f_x$ : System clock oscillation frequency
- 2.**  $n$ : Value determined by setting TPS000 to TPS003 ( $1 \leq n \leq 8$ )
- 3.** The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

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

**(i) Generation of baud rate transmit/receive clock from system clock**

The transmit/receive clock is generated by dividing the system clock. The baud rate generated from the system clock is estimated by using the following expression.

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

$f_x$ : System clock oscillation frequency

$n$ : Value determined by setting TPS000 to TPS003 as shown in the above table ( $2 \leq n \leq 8$ )

**Table 9-5. Example of Relationship Between System Clock and Baud Rate**

Baud Rate (bps)	n	BRGC00 Set Value	Error (%)	
			f <sub>x</sub> = 5.0 MHz	f <sub>x</sub> = 4.9152 MHz
1,200	8	70H	1.73	0
2,400	7	60H		
4,800	6	50H		
9,600	5	40H		
19,200	4	30H		
38,400	3	20H		
76,800	2	10H		

**Caution** Be sure not to select n = 1 during operation at f<sub>x</sub> = 5.0 MHz because the n = 1 setting exceeds the baud rate limit.

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

The transmit/receive clock is generated by dividing the clock input from the ASCK pin. The baud rate generated from the clock input from the ASCK pin is estimated by using the following expression.

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

f<sub>ASCK</sub>: Frequency of clock input to the ASCK pin

**Table 9-6. Relationship Between ASCK Pin Input Frequency and Baud Rate (When BRGC00 Is Set to 80H)**

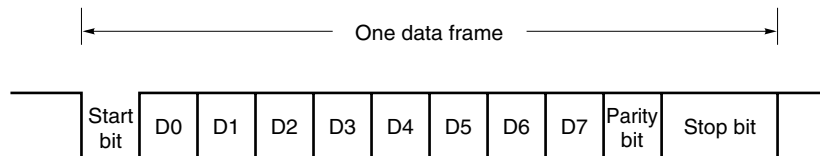
Baud Rate (bps)	ASCK Pin Input Frequency (kHz)
75	1.2
150	2.4
300	4.8
600	9.6
1,200	19.2
2,400	38.4
4,800	76.8
9,600	153.6
19,200	307.2
31,250	500.0
38,400	614.4

**(2) Communication operation****(a) Data format**

The transmit/receive data format is as shown in Figure 9-7. 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 one data frame is carried out using asynchronous serial interface mode register 00 (ASIM00).

**Figure 9-7. Format of Asynchronous Serial Interface Transmit/Receive Data**



- Start bit ..... 1 bit
- Character bits ..... 7 bits/8 bits
- Parity bit..... Even parity/odd parity/0 parity/no parity
- Stop bit..... 1 bit/2 bits

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 baud rate generator control register 00 (BRGC00).

If a serial data receive error occurs, the receive error contents can be determined by reading the status of 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 “1” bit (odd number) error can be detected. With 0 parity and no parity, an error cannot be detected.

**(i) Even parity****• At transmission**

The parity bit is determined 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 occurs.

**(ii) Odd parity****• At transmission**

Conversely to the even parity, the parity bit is determined 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 occurs.

**(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 does not occur, 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.

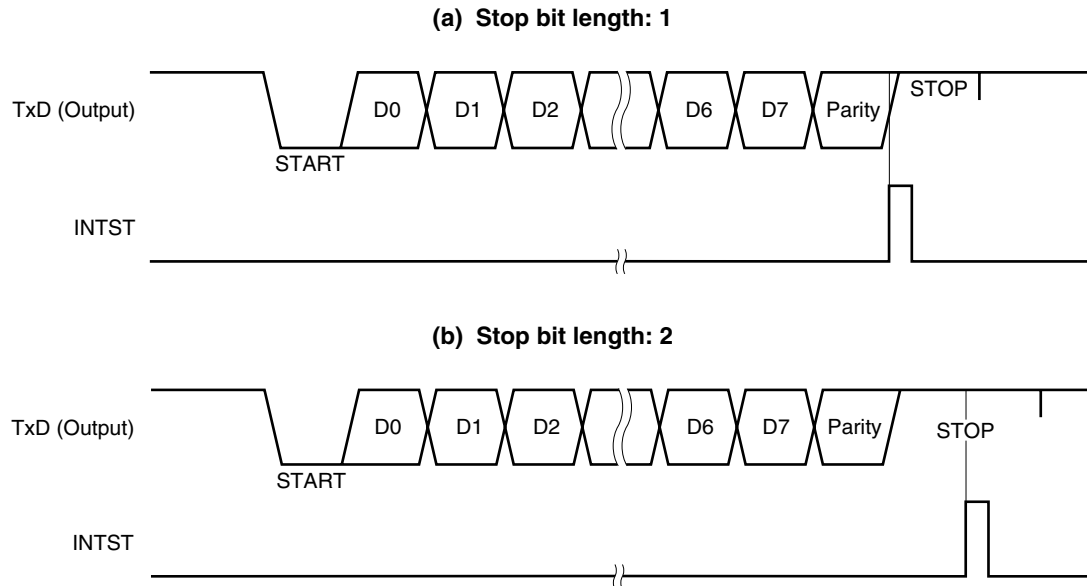
At reception, data is received assuming that there is no parity bit. Since there is no parity bit, a parity error does not occur.

**(c) Transmission**

A transmit operation is started by writing transmit data to 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 9-8. Asynchronous Serial Interface Transmission Completion Interrupt Timing**



**Caution** Do not rewrite asynchronous serial interface mode register 00 (ASIM00) during a transmit operation. If the ASIM00 register is rewritten during transmission, subsequent transmission may not be performed (the normal state is restored by 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 (RXE00) of asynchronous serial interface mode register 00 (ASIM00) is set to 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.

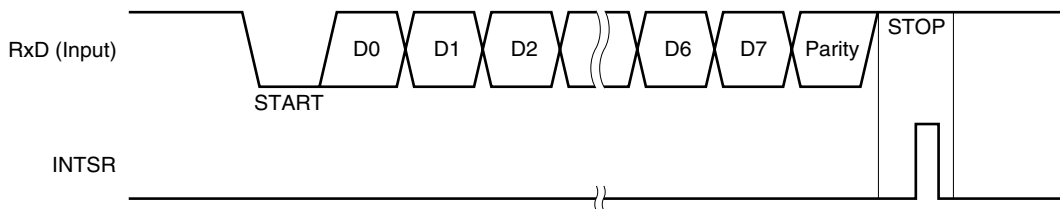
When the RxD pin input becomes low, the 3-bit counter starts counting, and when half the time determined by the specified baud rate has passed, the data sampling start timing signal is output. If the RxD pin input sampled again as a result of this start timing signal is low, it is identified as a start 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 receive buffer register 00 (RXB00), and a reception completion interrupt (INTSR) is generated.

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

If the RXE00 bit is reset to 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 9-9. Asynchronous Serial Interface Reception Completion Interrupt Timing**



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



**(e) Receive errors**

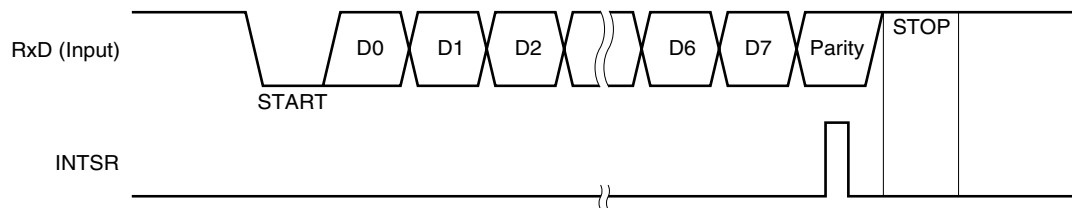
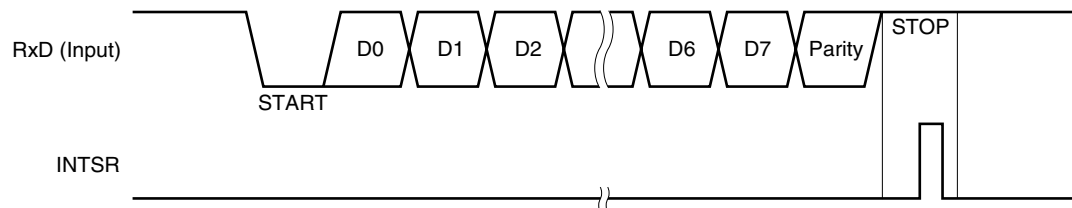
The following three errors may occur during a receive operation: a parity error, framing error, or overrun error. Upon data reception, an error flag is set in asynchronous serial interface status register 00 (ASIS00). Receive error causes are shown in Table 9-7.

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

The contents of ASIS00 are reset to 0 by reading 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 9-7. Receive Error Causes**

Receive Error	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 9-10. Receive Error Timing****(a) Parity error occurs****(b) Framing error or overrun error occurs**

**Cautions** 1. The contents of the ASIS00 register are reset to 0 by reading 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 receive buffer register 00 (RXB00) even if a receive error occurred. If RXB00 is not read, an overrun error will occur when the next data is received, and the receive error state will continue indefinitely.

## ★ (f) Reading receive data

When the reception completion interrupt (INTSR) is generated, receive data can be read by reading the value of receive buffer register 00 (RXB00).

To read the receive data stored in receive buffer register 00 (RXB00), read while reception is enabled (RXE00 = 1).

**Remark** However, if it is necessary to read receive data after reception has stopped (RXE00 = 0), read using either of the following methods.

(a) Read after setting RXE00 = 0 after waiting for one cycle or more of the source clock selected by BRGC00.

(b) Read after bit 2 (DIR00) of serial operation mode register 00 (CSIM00) is set to 1.

Program example of (a) (BRGC00 = 00H (source clock =  $f_x/2$ ))

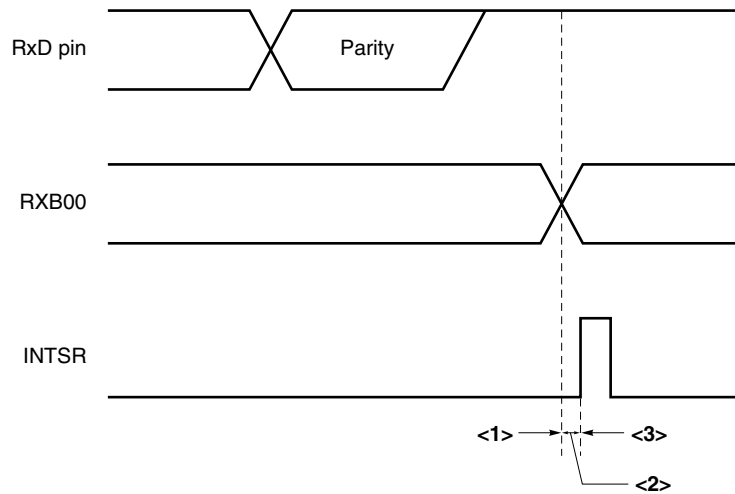
```
INTRXE:                                ; <Reception completion interrupt routine>
      NOP                               ; 2 clocks
      CLR1 RXE00                        ; Reception stopped
      MOV  A, RXB00                     ; Read receive data
```

Program example of (b)

```
INTRXE:                                ; <Reception completion interrupt routine>
      SET1 CSIM00.2                     ; DIR00 flag is set to LSB first
      CLR1 RXE00                        ; Reception stopped
      MOV  A, RXB00                     ; Read receive data
```

**(3) Cautions related to UART mode**

- (a) When bit 7 (TXE00) of asynchronous serial interface mode register 00 (ASIM00) is cleared during transmission, be sure to set transmit shift register 00 (TXS00) to FFH, then set TXE00 to 1 before executing the next transmission.
- (b) When bit 6 (RXE00) of asynchronous serial interface mode register 00 (ASIM00) is cleared during reception, receive buffer register 00 (RXB00) and reception completion interrupt (INTSR) are as follows.



When RXE00 is set to 0 at the time indicated by <1>, RXB00 holds the previous data and INTSR is not generated.

When RXE00 is set to 0 at the time indicated by <2>, RXB00 renews the data and INTSR is not generated.

When RXE00 is set to 0 at the time indicated by <3>, RXB00 renews the data and INTSR is generated.

### 9.4.3 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 serial interface, such as the 75X/XL 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 operation mode register 00 (CSIM00), asynchronous serial interface mode register 00 (ASIM00), and baud rate generator control register 00 (BRGC00).

##### (a) Serial operation mode register 00 (CSIM00)

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

$\overline{\text{RESET}}$  input clears 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 stopped
1	Operation enabled

DIR00	Start bit specification
0	MSB
1	LSB

CSCK00	Clock selection in 3-wire serial I/O mode
0	Clock input to $\overline{\text{SCK0}}$ pin from external
1	Dedicated baud rate generator output

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

**(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 clears ASIM00 to 00H.

When the 3-wire serial I/O mode is selected, ASIM00 must be set 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 stopped
1	Transmit operation enabled

RXE00	Receive operation control
0	Receive operation stopped
1	Receive operation enabled

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 occurs)
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 bits 0 and 1 to 0.

2. Switching operating modes must be performed after the serial transmit/receive operation has been stopped.

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

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

$\overline{\text{RESET}}$  input clears 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 a communication operation.
  2. Be sure not to select  $n = 1$  during operation at  $f_x = 5.0$  MHz because the  $n = 1$  setting exceeds the baud rate limit.
  3. When selecting the clock input from an external source, set port mode register 2 (PM2) to the input mode.

- Remarks**
1.  $f_x$ : System clock oscillation frequency
  2.  $n$ : Value determined by setting TPS000 to TPS003 ( $1 \leq n \leq 8$ )
  3. The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

If the internal clock is used as the serial clock for 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 external, setting BRGC00 is not necessary.

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

$f_x$ : System clock oscillation frequency

$n$ : Value determined by setting TPS000 to TPS003 as shown in the above table ( $1 \leq n \leq 8$ )

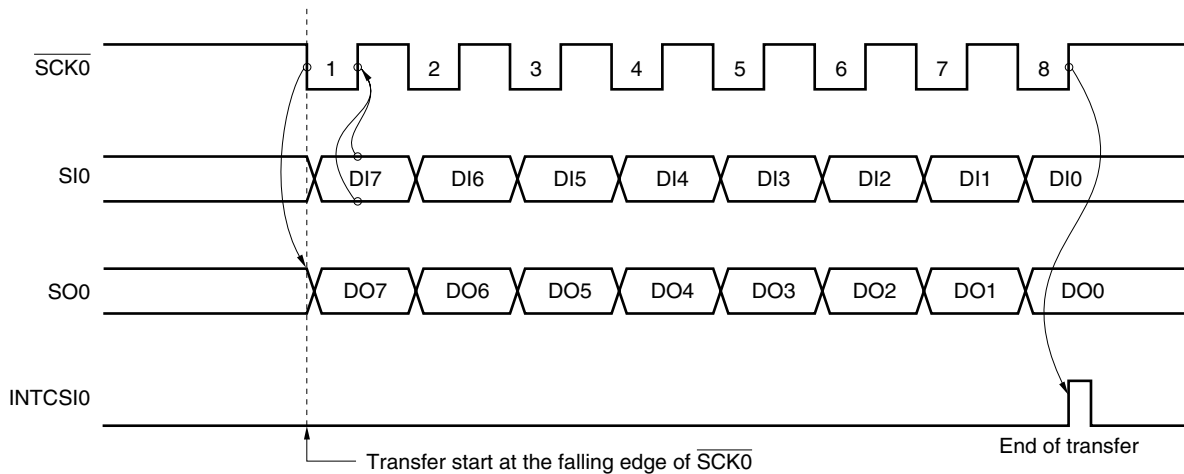
**(2) Communication operation**

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 S00 latch and output from the S00 pin. Also, receive data input to the SI0 pin is latched in receive buffer register 00 (RXB00/SIO00) on the rise of  $\overline{\text{SCK0}}$ .

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

**Figure 9-11. 3-Wire Serial I/O Mode Timing**

**(3) Transfer start**

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

- Serial operation mode register 00 (CSIM00) bit 7 (CSIE00) = 1
- Internal serial clock is stopped or  $\overline{\text{SCK0}}$  is at 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).

## CHAPTER 10 INTERRUPT FUNCTIONS

### 10.1 Interrupt Function Types

The following two types of interrupt functions are used.

#### (1) **Non-maskable interrupt**

This interrupt is acknowledged unconditionally. It does not undergo interrupt priority control and is given top priority over all other interrupt requests.

A standby release signal is generated.

The interrupt from the watchdog timer is the one non-maskable interrupt source.

#### (2) **Maskable interrupts**

These interrupts undergo mask control. If two or more interrupts are simultaneously generated, each interrupt has a predetermined priority as shown in Table 10-1.

A standby release signal is generated.

Maskable interrupts have four external interrupt and five internal interrupt sources.

### 10.2 Interrupt Sources and Configuration

There are a total of ten non-maskable and maskable interrupt sources (see **Table 10-1**).



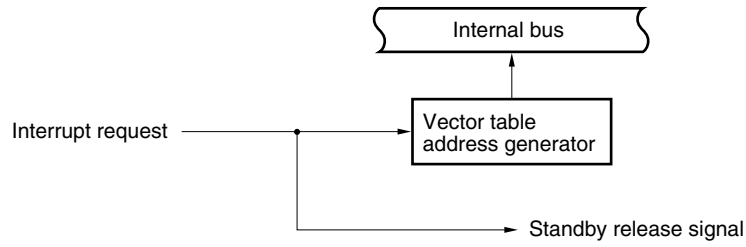
**Table 10-1. Interrupt Source List**

Interrupt Type	Priority <sup>Note 1</sup>	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type <sup>Note 2</sup>
		Name	Trigger			
Non-maskable	–	INTWDT	Watchdog timer overflow (watchdog timer mode 1 selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (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 8-bit timer/event counter 00 match signal		0010H	
	7	INTTM2	Generation of 16-bit timer 20 match signal		0014H	
	8	INTKR	Key return signal detection	External	002AH	(C)

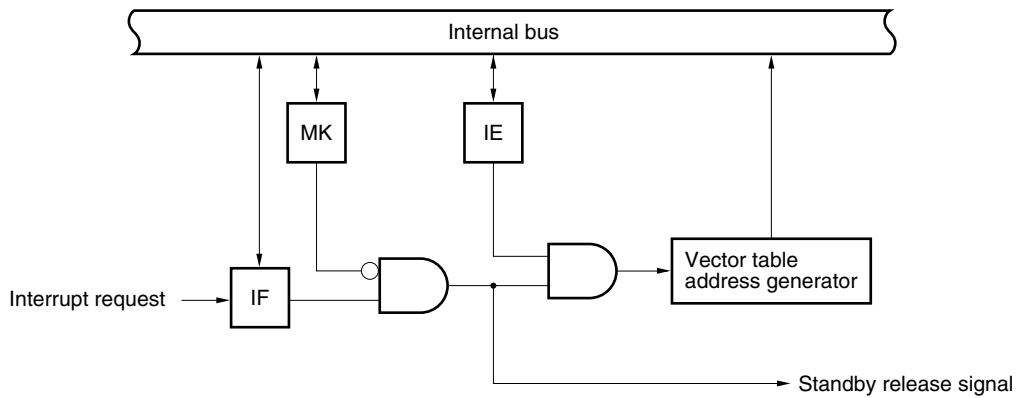
- Notes**
1. The priority is the priority applicable when two or more maskable interrupts are generated simultaneously. 0 is the highest priority and 8 is the lowest.
  2. Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 10-1.

Figure 10-1. Basic Configuration of Interrupt Function

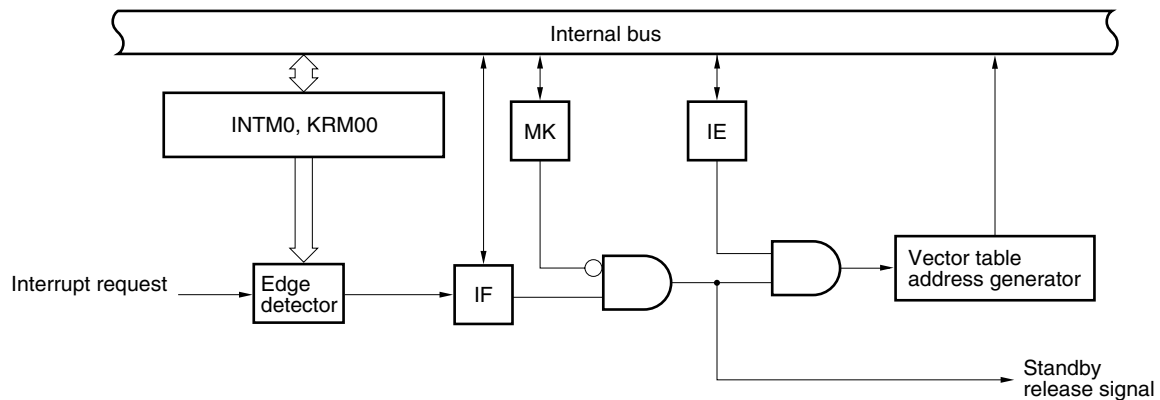
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt



INTM0: External interrupt mode register 0

KRM00: Key return mode register 00

IF: Interrupt request flag

IE: Interrupt enable flag

MK: Interrupt mask flag

### 10.3 Registers Controlling Interrupt Function

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

- Interrupt request flag registers (IF0 and IF1)
- Interrupt mask flag registers (MK0 and MK1)
- External interrupt mode register 0 (INTM0)
- Program status word (PSW)
- Key return mode register 00 (KRM00)

Table 10-2 lists the interrupt request flag and interrupt mask flag names corresponding to interrupt requests.

**Table 10-2. Flags Corresponding to Interrupt Request Signal Names**

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
INTTM2	TMIF20	TMMK20
INTKR	KRIF00	KRMK00

**(1) Interrupt request flag registers (IF0 and IF1)**

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 upon acknowledgement of an interrupt request, upon  $\overline{\text{RESET}}$  input, or when an instruction is executed.

IF0 and IF1 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears IF0 and IF1 to 00H.

**Figure 10-2. Format of Interrupt Request Flag Register**

Symbol	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF0	0	TMIF00	STIF00	SRIF00	PIF2	PIF1	PIF0	TMIF4	FFE0H	00H	R/W
	<7>	6	5	4	3	2	1	<0>			
IF1	TMIF20	0	0	0	0	0	0	KRIF00	FFE1H	00H	R/W

xxIFx	Interrupt request flag
0	No interrupt request signal is generated
1	Interrupt request signal is generated; interrupt request state

**Cautions** 1. Be sure to clear bit 7 of IF0 and bits 1 to 6 of IF1 to 0.

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

3. Because port 3 has an alternate function as an 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, the interrupt mask flag should be set to 1 before using the output mode.

4. If an interrupt is acknowledged, the interrupt request flag is automatically cleared before the interrupt routine is entered.

★

**(2) Interrupt mask flag registers (MK0 and MK1)**

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

MK0 and MK1 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input sets MK0 and MK1 to FFH.

**Figure 10-3. Format of Interrupt Mask Flag Register**

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

	<7>	6	5	4	3	2	1	<0>			
MK1	TMMK20	1	1	1	1	1	1	KRMK00	FFE5H	FFH	R/W

xxMKx	Interrupt servicing control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

**Cautions** 1. Be sure to set bit 7 of MK0 and bits 1 to 6 of MK1 to 1.

2. If the TMMK4 flag is read when the watchdog timer is used in watchdog timer mode 1 or 2, its value becomes undefined.
3. Because port 3 has an alternate function as an 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, the interrupt mask flag should be set to 1 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.

RESET input clears INTM0 to 00H.

**Figure 10-4. Format of External Interrupt Mode Register 0**

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 bits 0 and 1 to 0.

- Before setting the INTM0 register, be sure to set the corresponding interrupt mask flag ( $\times\times\text{MK}\times = 1$ ) to disable interrupts. After setting the INTM0 register, clear the interrupt request flag ( $\times\times\text{IF}\times = 0$ ), then clear the interrupt mask flag ( $\times\times\text{MK}\times = 0$ ) to enable interrupts.

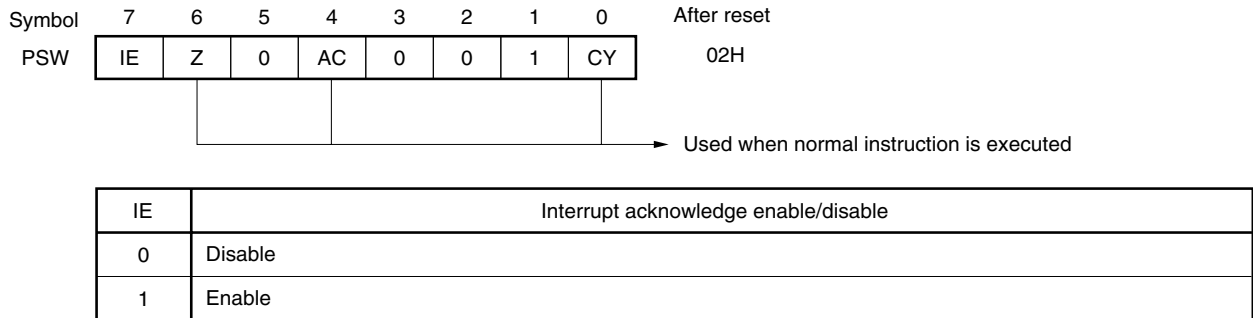
#### (4) Program status word (PSW)

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

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

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

**Figure 10-5. Configuration of Program Status Word**



**(5) Key return mode register 00 (KRM00)**

KRM00 is used to specify the pin at which a key return signal is detected.

KRM00 is set with a 1-bit or 8-bit memory manipulation instruction.

Bit 0 (KRM000) is set for the four pins from KR0/P40 to KR3/P43. Bits 4 to 7 (KRM004 to KRM007) are set in 1-bit units for pins KR4/P44 to KR7/P47, respectively.

RESET input clears KRM00 to 00H.

**Figure 10-6. Format of Key Return Mode Register 00**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
KRM00	KRM007	KRM006	KRM005	KRM004	0	0	0	KRM000	FFF5H	00H	R/W

KRM00n	Key return signal detection selection
0	Undetected
1	Detected (at the falling edge of port 4)

**Cautions** 1. Be sure to set bits 1 to 3 to 0.

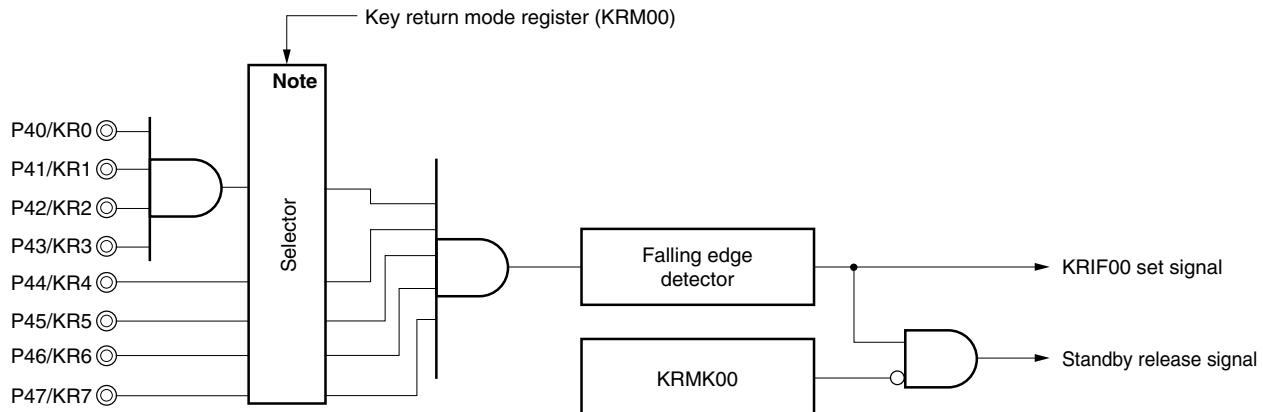
2. When KRM00n is set to 1, the corresponding pin is connected to a pull-up resistor unless it is in output mode. In output mode, the pull-up resistor is not connected.

3. Before setting KRM00, set bit 0 of MK1 (KRMK00 = 1) to disable interrupts.

To enable interrupts, clear bit 0 of IF1 (KRIF00 = 0), then bit 0 of MK1 (KRMK00 = 0).

**Remark** n = 0, 4 to 7

**Figure 10-7. Block Diagram of Falling Edge Detector**



**Note** Selector used to select the pin to be used for falling edge input



## 10.4 Interrupt Servicing Operation

### 10.4.1 Non-maskable interrupt request acknowledgement operation

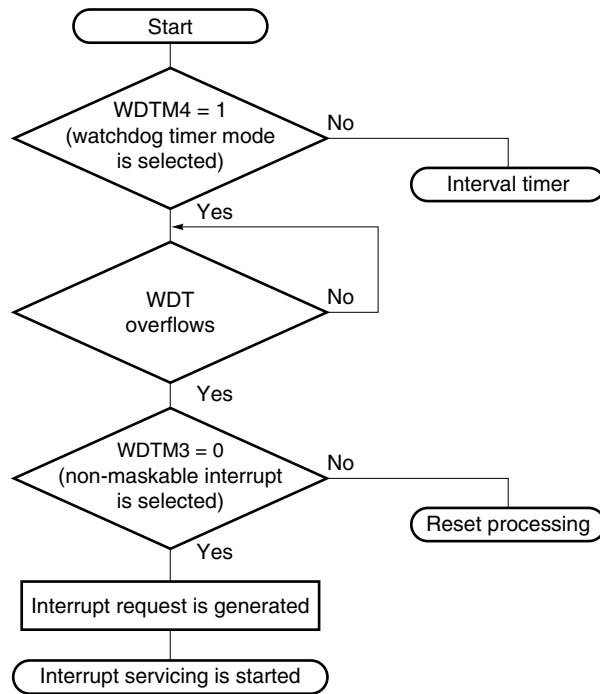
The non-maskable interrupt request is unconditionally acknowledged even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

When the non-maskable interrupt request is acknowledged, the 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 10-8 shows the flowchart from non-maskable interrupt request generation to acknowledgement. Figure 10-9 shows the timing of non-maskable interrupt request acknowledgement. Figure 10-10 shows the acknowledgement operation if multiple non-maskable interrupts are generated.

**Caution** During 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 interrupt request will be acknowledged.

Figure 10-8. Flowchart from Non-Maskable Interrupt Request Generation to Acknowledgement



WDTM: Watchdog timer mode register

WDT: Watchdog timer

Figure 10-9. Non-Maskable Interrupt Request Acknowledgement Timing

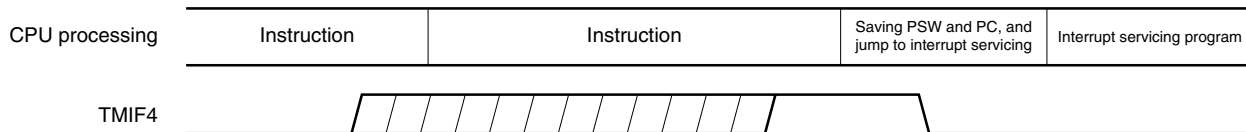
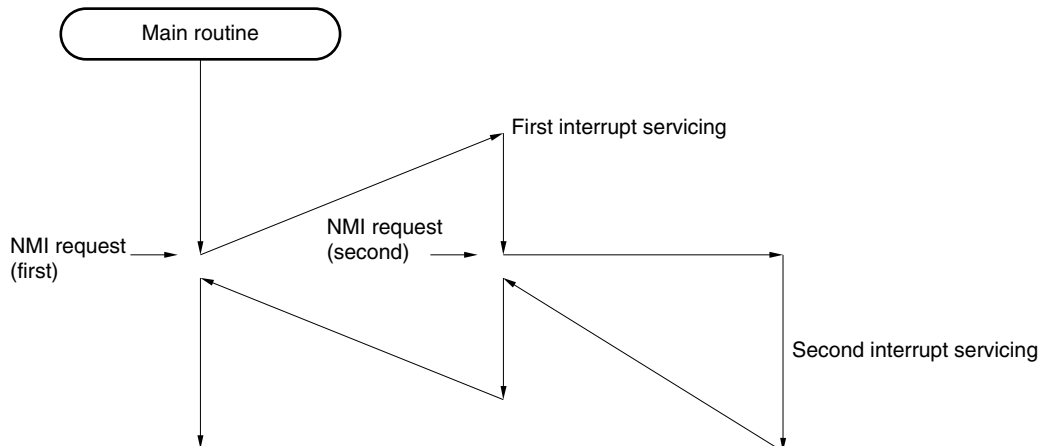


Figure 10-10. Acknowledgement of Non-Maskable Interrupt Request



### 10.4.2 Maskable interrupt request acknowledgement operation

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

The time required to start the interrupt servicing after a maskable interrupt request has been generated is shown in Table 10-3.

See Figures 10-12 and 10-13 for the interrupt request acknowledgement timing.

**Table 10-3. Time from Generation of Maskable Interrupt Request to Servicing**

Minimum Time	Maximum Time <sup>Note</sup>
9 clocks	19 clocks

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

**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 acknowledged starting from the interrupt request assigned the highest priority.

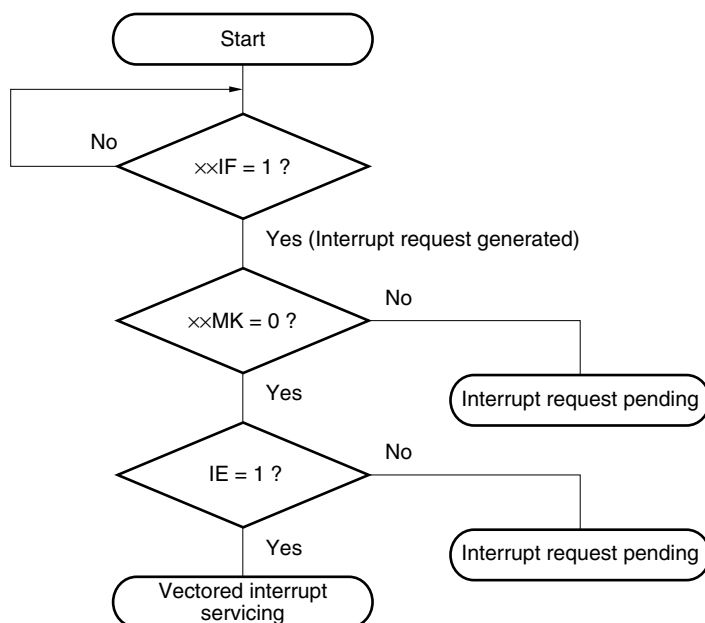
A pending interrupt is acknowledged when the status in which it can be acknowledged is set.

Figure 10-11 shows the algorithm of acknowledging interrupt requests.

When a maskable interrupt request is acknowledged, the contents of the 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 servicing, use the RETI instruction.

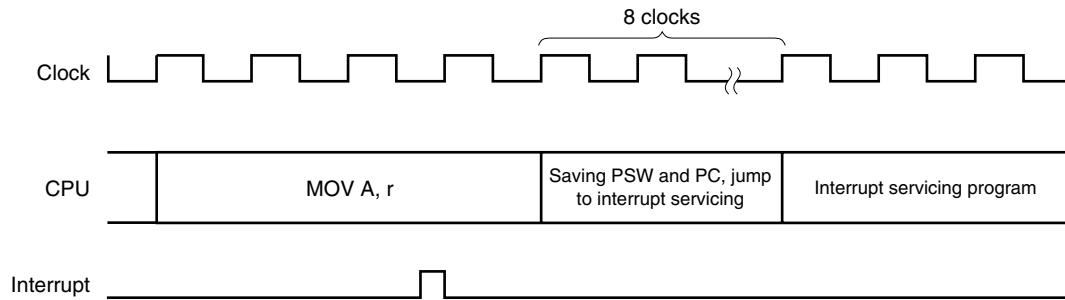
Figure 10-11. Interrupt Request Acknowledgement Processing Algorithm



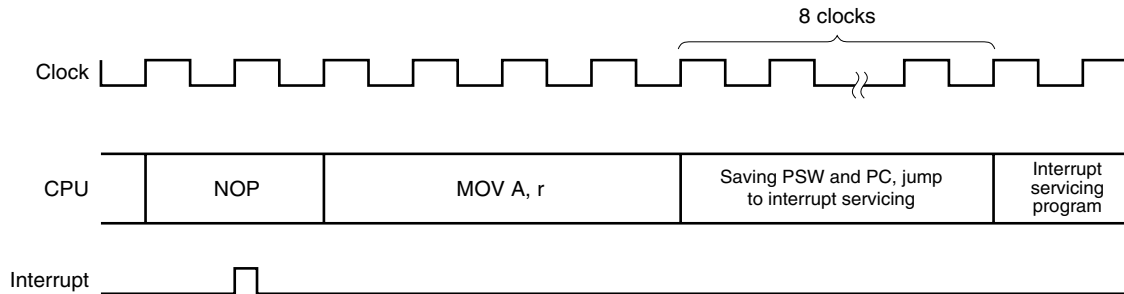
xxIF: Interrupt request flag

xxMK: Interrupt mask flag

IE: Flag to control maskable interrupt request acknowledgement (1 = enable, 0 = disable)

**Figure 10-12. Interrupt Request Acknowledgement Timing (Example of MOV A, r)**

If an interrupt request flag ( $\times\times IF$ ) is set before instruction clock  $n$  ( $n = 4$  to  $10$ ) under execution becomes  $n - 1$ , the interrupt is acknowledged after the instruction under execution is complete. Figure 10-12 shows an example of the interrupt request acknowledgement timing for an 8-bit data transfer instruction MOV A, r. Since this instruction is executed for 4 clocks, if an interrupt occurs within 3 clocks after the execution starts, the interrupt acknowledgement processing is performed after the MOV A, r instruction is completed.

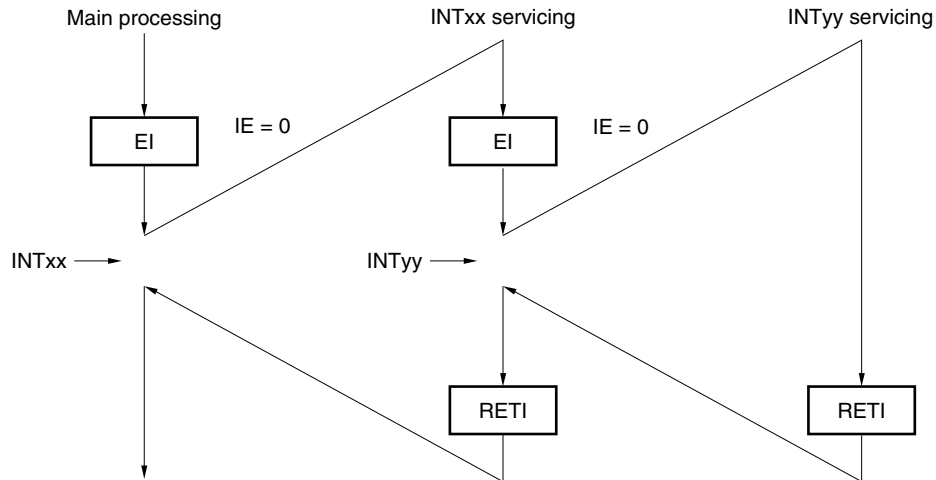
**Figure 10-13. Interrupt Request Acknowledgement Timing (When Interrupt Request Flag Is Set at Last Clock During Instruction Execution)**

If an interrupt request flag ( $\times\times IF$ ) is set at the last clock of the instruction, the interrupt acknowledgement processing starts after the next instruction is executed. Figure 10-13 shows an example of the interrupt acknowledgement 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 acknowledgement processing is performed.

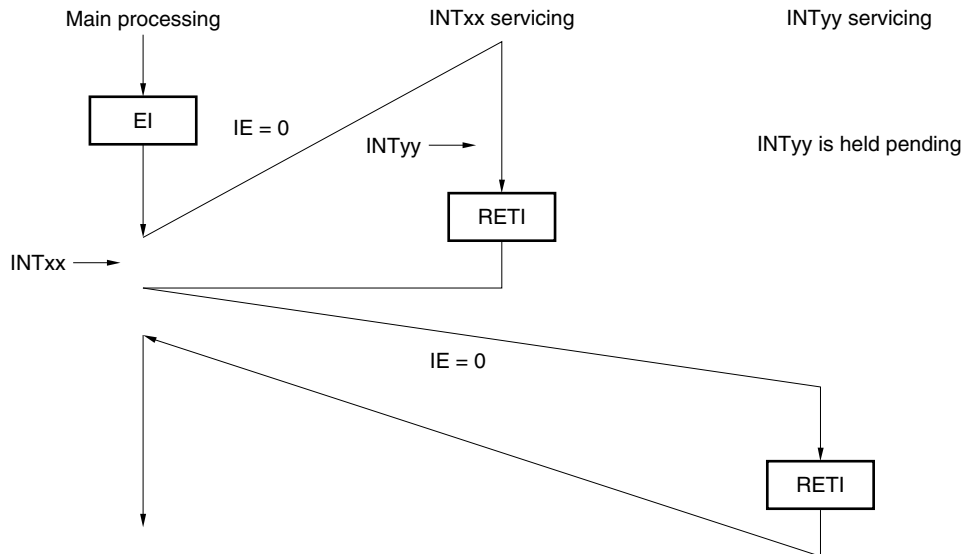
**Caution** Interrupt requests are held pending while the interrupt request flag register (IF0 or IF1) or the interrupt mask flag register (MK0 or MK1) is being accessed.

### 10.4.3 Nesting processing

Nesting processing in which another interrupt is acknowledged while an interrupt is serviced can be executed by priority. When two or more interrupts are generated at once, interrupt servicing is performed according to the priority assigned to each interrupt request in advance (see **Table 10-1**).

**Figure 10-14. Example of Nesting****Example 1. Nesting is acknowledged**

During interrupt INTxx servicing, interrupt request INTyy is acknowledged, and nesting occurs. The EI instruction is issued before each interrupt request acknowledgement, and the interrupt request acknowledgement enable state is set.

**Example 2. Nesting does not occur because interrupts are not enabled**

Because interrupts are not enabled in interrupt INTxx servicing (the EI instruction is not issued), interrupt request INTyy is not acknowledged, and nesting does not occur. The INTyy request is held pending and acknowledged after the INTxx servicing is performed.

IE = 0: Interrupt request acknowledgement disabled

#### 10.4.4 Interrupt request hold

Some instructions may hold the acknowledgement of an interrupt request pending 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 of that instruction. These instructions (interrupt request hold instructions) are shown below.

- Manipulation instruction for interrupt request flag registers 0 and 1 (IF0 and IF1)
- Manipulation instruction for interrupt mask flag registers 0 and 1 (MK0 and MK1)

## CHAPTER 11 STANDBY FUNCTION

### 11.1 Standby Function and Configuration

#### 11.1.1 Standby function

The standby function is used to reduce the power consumption 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 current consumption 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 current consumption of the CPU can be substantially reduced in this mode.

Data memory can be retained at a low voltage ( $V_{DD} = 1.8 \text{ V min.}$ ). Therefore, this mode is useful for retaining the contents of the data memory with an extremely low current consumption.

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 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.



### 11.1.2 Standby function control register

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

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

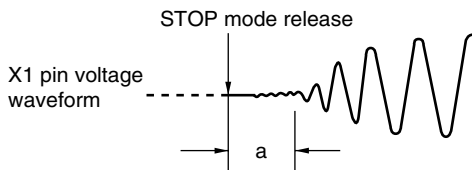
$\overline{\text{RESET}}$  input sets OSTS to 04H. However, the oscillation stabilization time after  $\overline{\text{RESET}}$  input is  $2^{15}/f_x$ , instead of  $2^{17}/f_x$ .

**Figure 11-1. Format of Oscillation Stabilization Time Select Register**

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 $\mu$ s)
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. The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

## 11.2 Operation of Standby Function

### 11.2.1 HALT mode

#### (1) HALT mode

HALT mode is set by executing the HALT instruction.

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

**Table 11-1. HALT Mode Operating Status**

Item	HALT Mode Operating Status
Clock generator	System clock oscillation enabled Clock supply to CPU stopped
CPU	Operation stopped
Port (output latch)	Retains the status before setting the HALT mode
16-bit timer	Operation enabled
8-bit timer/event counter	Operation enabled
Watchdog timer	Operation enabled
Serial interface	Operation enabled
External interrupt	Operation enabled
Key return	Only the pin set to key return mode is enabled

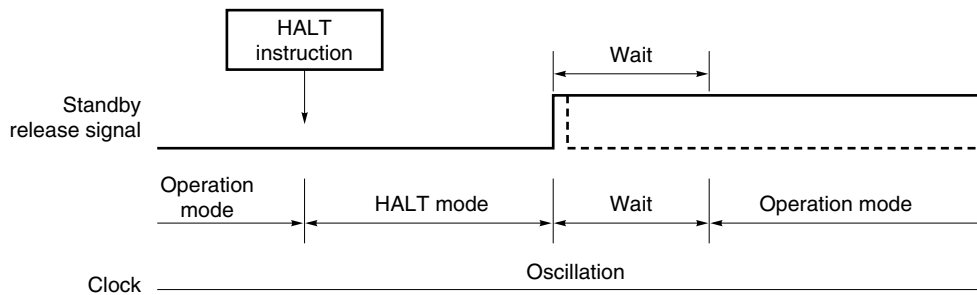
**(2) Releasing HALT mode**

The HALT mode can be released by the following three 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 acknowledged, vectored interrupt servicing is performed. If interrupt acknowledgement is disabled, the instruction at the next address is executed.

**Figure 11-2. Releasing HALT Mode by Interrupt**



**Remarks 1.** The broken lines indicate the case where the interrupt request that has released the standby mode is acknowledged.

**2.** The wait time is as follows.

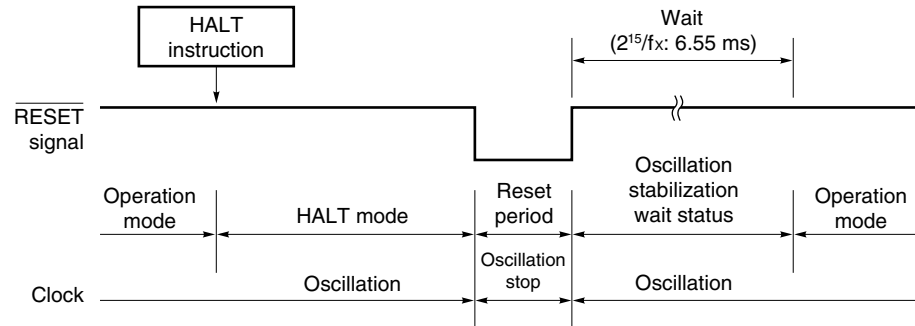
- When vectored interrupt servicing is performed: 9 to 10 clocks
- When vectored interrupt servicing is not performed: 1 to 2 clocks

**(b) Releasing by non-maskable interrupt request**

The HALT mode is released regardless of whether interrupts are enabled or disabled, and vectored interrupt servicing 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 11-3. Releasing HALT Mode by  $\overline{\text{RESET}}$  Input**

- Remarks**
1. f<sub>x</sub>: System clock oscillation frequency
  2. The parenthesized values apply to operation at f<sub>x</sub> = 5.0 MHz.

**Table 11-2. Operation After Release of HALT Mode**

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

×: Don't care

## 11.2.2 STOP mode

## (1) Setting and operation status of STOP mode

The STOP mode is set by executing the STOP instruction.

**Caution** 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 operation mode is set.

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

Table 11-3. 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
16-bit timer	Stops operation
8-bit timer/event counter	Operation is enabled only when TI0 is selected as the count clock
Watchdog timer	Stops operation
Serial interface	Operation is enabled only when clock input from external is selected as serial clock
External interrupt	Operation enabled
Key return	Only the pin set to key return mode is enabled

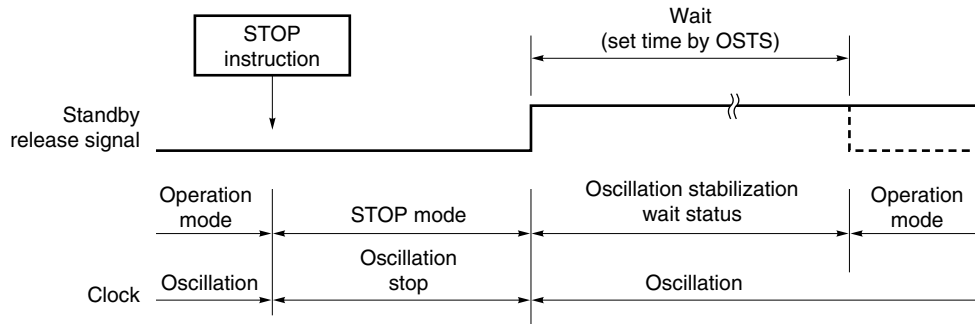
**(2) Releasing STOP mode**

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 acknowledged, vectored interrupt servicing is performed, after the oscillation stabilization time has elapsed. If interrupt acknowledgement is disabled, the instruction at the next address is executed.

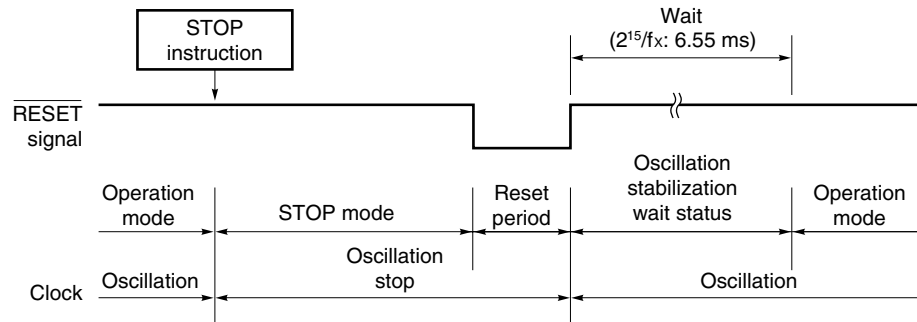
**Figure 11-4. Releasing STOP Mode by Interrupt**



**Remark** The broken lines indicate the case where the interrupt request that has released the standby mode is acknowledged.

**(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 11-5. Releasing STOP Mode by  $\overline{\text{RESET}}$  Input**

- Remarks**
1.  $f_x$ : System clock oscillation frequency
  2. The parenthesized values apply to operation at  $f_x = 5.0$  MHz.

**Table 11-4. Operation After Release of STOP Mode**

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

$\times$ : Don't care

## CHAPTER 12 RESET FUNCTION

The following two operations are available to generate reset signals.

- (1) External reset input via  $\overline{\text{RESET}}$  pin
- (2) Internal reset by program loop time detected by 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 12-1. Each pin is high impedance during reset input or during oscillation stabilization time just after reset release.

When a high level is input to the  $\overline{\text{RESET}}$  pin, the reset is released and program execution is started after the oscillation stabilization time ( $2^{15}/f_x$ ) has elapsed. The reset applied by the watchdog timer overflow is automatically released after reset, and program execution is started after the oscillation stabilization time ( $2^{15}/f_x$ ) has elapsed (see **Figures 12-2 through 12-4**).

- Cautions**
1. For an external reset, input a low level for 10  $\mu\text{s}$  or more to the  $\overline{\text{RESET}}$  pin.
  2. When the STOP mode is released by reset, the STOP mode contents are held during reset input. However, the port pins become high impedance.

**Figure 12-1. Block Diagram of Reset Function**

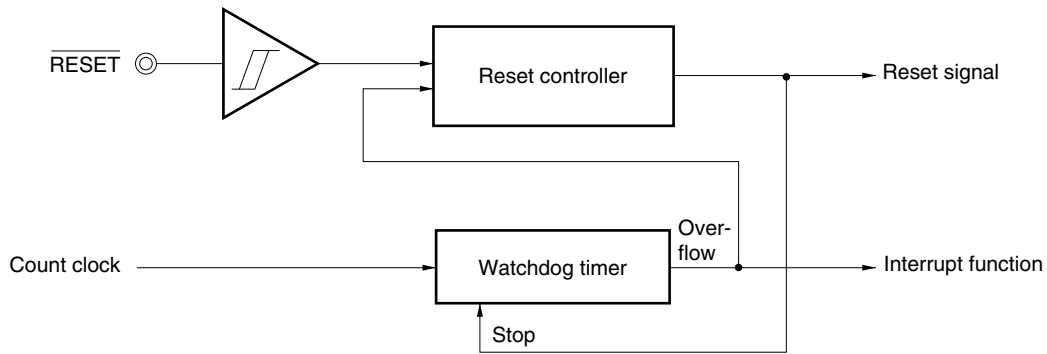




Figure 12-2. Reset Timing by  $\overline{\text{RESET}}$  Input

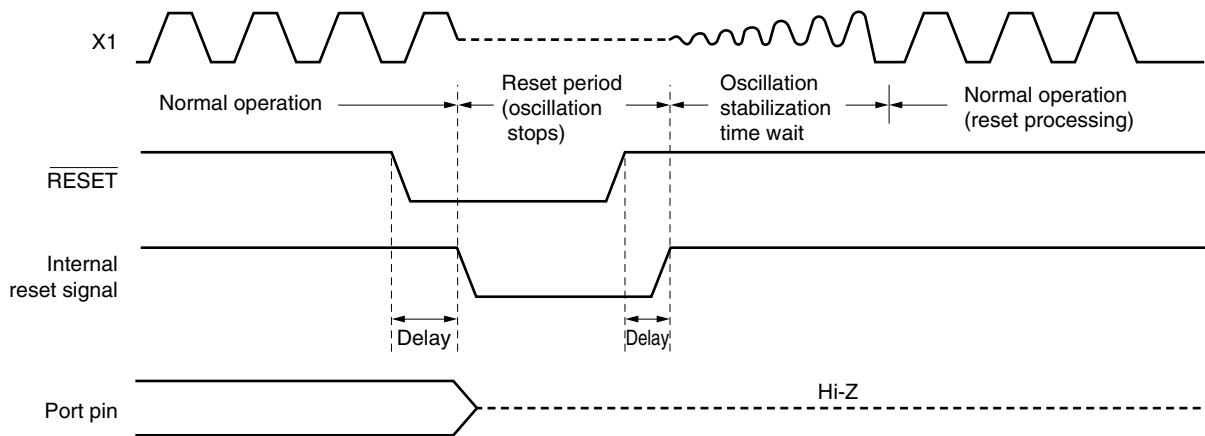


Figure 12-3. Reset Timing by Overflow in Watchdog Timer

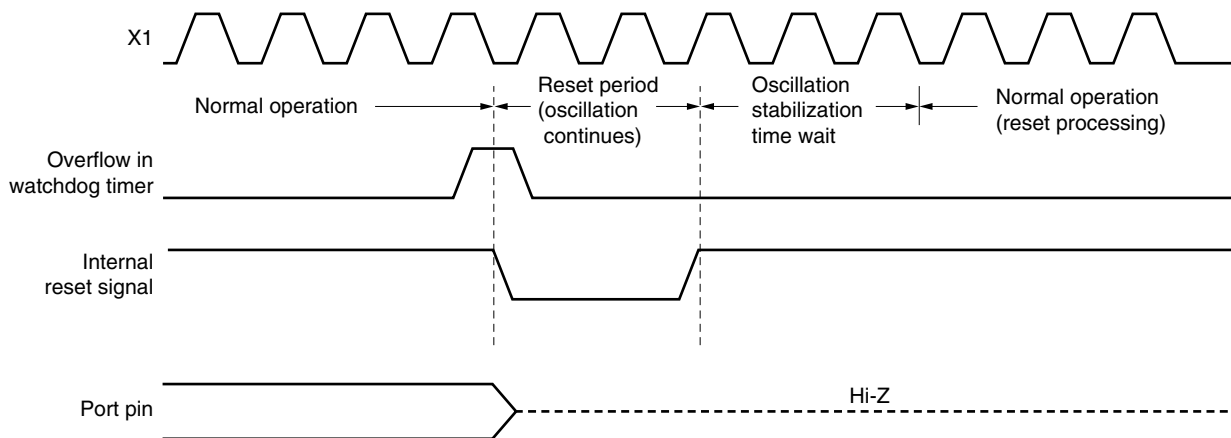


Figure 12-4. Reset Timing by  $\overline{\text{RESET}}$  Input in STOP Mode

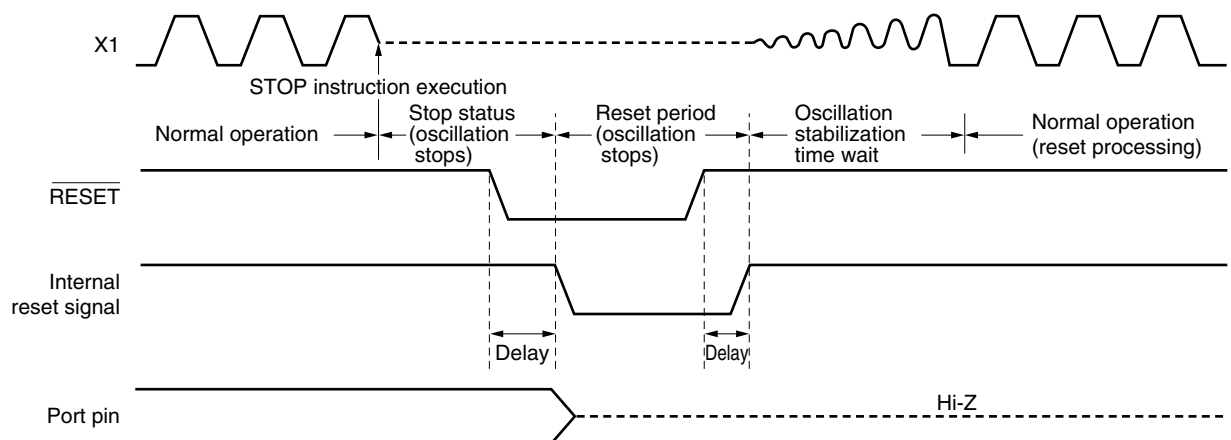


Table 12-1. Hardware Status After Reset

Hardware		Status After Reset
Program counter (PC) <sup>Note 1</sup>		The contents of reset vector tables (0000H and 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined <sup>Note 2</sup>
	General-purpose registers	Undefined <sup>Note 2</sup>
Ports (P0 to P5) (output latch)		00H
Port mode registers (PM0 to PM5)		FFH
Pull-up resistor option register (PUO)		00H
Processor clock control register (PCC)		02H
Oscillation stabilization time select register (OSTS)		04H
16-bit timer 20	Timer counter (TM20)	0000H
	Compare register (CR20)	FFFFH
	Mode control register (TMC20)	00H
	Capture register (TCP20)	Undefined
8-bit timer/event counter 00	Timer counter (TM00)	00H
	Compare register (CR00)	Undefined
	Mode control register (TMC00)	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
Interrupts	Request flag registers (IF0, IF1)	00H
	Mask flag registers (MK0, MK1)	FFH
	External interrupt mode register (INTM0)	00H
	Key return mode register (KRM00)	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.

## CHAPTER 13 $\mu$ PD78F9026A

The  $\mu$ PD78F9026A is a version with the internal ROM of the mask ROM versions replaced with a flash memory. The differences between the  $\mu$ PD78F9026A and the mask ROM versions are shown in Table 13-1.

**Table 13-1. Differences Between  $\mu$ PD78F9026A and Mask ROM Versions**

Item		Flash Memory Version	Mask ROM Versions			
		μPD78F9026A	μPD789022	μPD789024	μPD789025	μPD789026
Internal memory	ROM	16 KB (flash memory)	4 KB	8 KB	12 KB	16 KB
	Internal high-speed RAM	512 bytes	256 bytes		512 bytes	
IC pin		Not provided	Provided			
V <sub>PP</sub> pin		Provided	Not provided			
Electrical specifications		Refer to <b>CHAPTER 15 ELECTRICAL SPECIFICATIONS.</b>				

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

### ★ 13.1 Flash Memory Characteristics

Flash memory programming is performed by connecting a dedicated flash programmer (Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4)) to the target system with the  $\mu$ PD78F9026A mounted on the target system (on-board). A flash memory program adapter (FA adapter), which is a target board used exclusively for programming, is also provided.

**Remark** FL-PR3, FL-PR4, and the program adapter are products made by Naito Densai Machida Mfg. Co., Ltd. (TEL +81-45-475-4191).

Programming using flash memory has the following advantages.

- Software can be modified after the microcontroller is solder-mounted on the target system.
- Distinguishing software facilities small-quantity, varied model production
- Easy data adjustment when starting mass production

#### 13.1.1 Programming environment

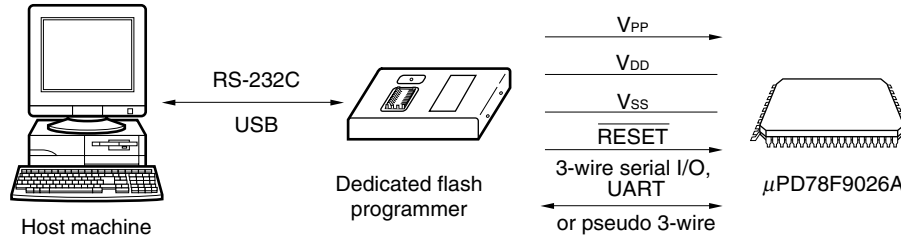
The following shows the environment required for  $\mu$ PD78F9026A flash memory programming.

When Flashpro III (part no. FL-PR3, PG-FP3) or Flashpro IV (part no. FL-PR4, PG-FP4) is used as a dedicated flash programmer, a host machine is required to control the dedicated flash programmer. Communication between the host machine and flash programmer is performed via RS-232C/USB (Rev. 1.1).

For details, refer to the manuals for Flashpro III/Flashpro IV.

**Remark** USB is supported by Flashpro IV only.

**Figure 13-1. Environment for Writing Program to Flash Memory**



### 13.1.2 Communication mode

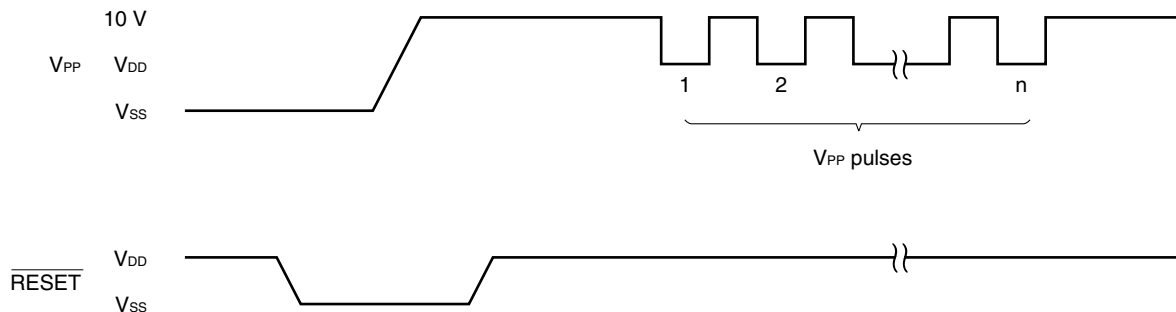
Use the communication mode shown in Table 13-2 to perform communication between the dedicated flash programmer and  $\mu$ PD78F9026A.

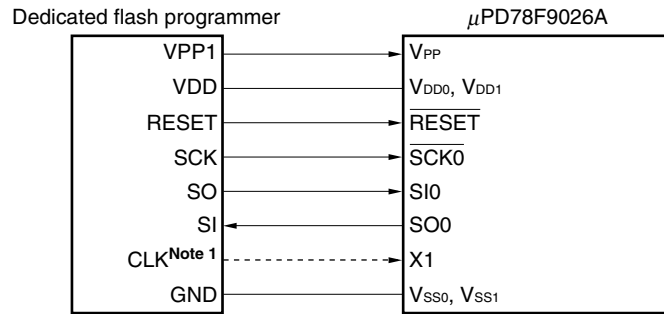
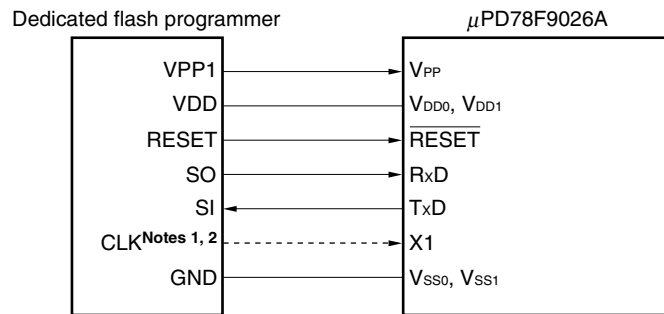
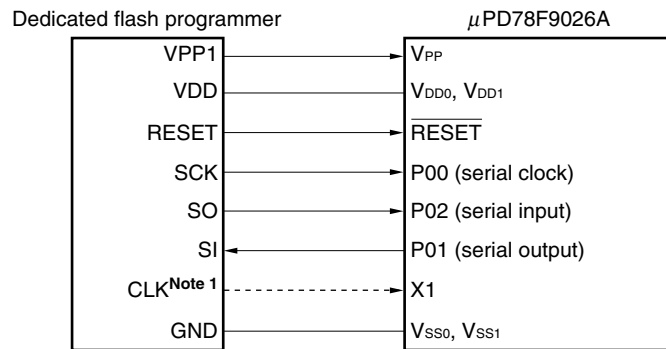
**Table 13-2. Communication Mode List**

Communication Mode	TYPE Setting <sup>Note 1</sup>					Pins Used	Number of V <sub>PP</sub> Pulses
	COMM PORT	SIO Clock	CPU Clock		Multiple Rate		
			In Flashpro	On Target Board			
3-wire serial I/O	SIO ch-0 (3-wire, sync.)	100 Hz to 1.25 MHz <sup>Note 2</sup>	1, 2, 4, 5 MHz <sup>Notes 2, 3</sup>	1 to 5 MHz <sup>Note 2</sup>	1.0	SI0/RxD/P22 SO0/TxD/P21 SCK0/ASCK/P20	0
UART	UART ch-0 (Async.)	4,800 to 76,800 bps <sup>Notes 2, 4</sup>	5 MHz <sup>Note 5</sup>	4.91 or 5 MHz <sup>Note 2</sup>	1.0	RxD/SI0/P22 TxD/SO0/P21	8
Pseudo 3-wire	Port A (Pseudo-3 wire)	100 Hz to 1 kHz	1, 2, 4, 5 MHz <sup>Notes 2, 3</sup>	1 to 5 MHz <sup>Note 2</sup>	1.0	P01 P02 P00	12
	Port B (Pseudo-3 wire)					P40/KR0 P41/KR1 P42/KR2	13

- Notes**
1. Selection items for TYPE settings on the dedicated flash programmer (Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4)).
  2. The possible setting range differs depending on the voltage. For details, refer to **CHAPTER 15 ELECTRICAL SPECIFICATIONS**.
  3. 2 or 4 MHz only for Flashpro III
  4. Because signal wave slew also affects UART communication, in addition to the baud rate error, thoroughly evaluate the slew and baud rate error.
  5. Only for Flashpro IV. However, when using Flashpro III, be sure to select the clock of the resonator on the board. UART cannot be used with the clock supplied by Flashpro III.

**Figure 13-2. Communication Mode Selection Format**



**Figure 13-3. Example of Connection with Dedicated Flash Programmer****(a) 3-wire serial I/O****(b) UART****(c) Pseudo 3-wire (when P0 is used)**

- Notes**
1. Connect this pin when the system clock is supplied from the dedicated flash programmer. If a resonator is already connected to the X1 pin, the CLK pin does not need to be connected.
  2. When using UART with Flashpro III, the clock of the resonator connected to the X1 pin must be used, so connection to the CLK pin is not necessary.

**Caution** The V<sub>DD</sub> pin, if already connected to the power supply, must be connected to the VDD pin of the dedicated flash programmer. When using the power supply connected to the V<sub>DD</sub> pin, supply voltage before starting programming.

If Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4) is used as a dedicated flash programmer, the following signals are generated for the  $\mu$ PD78F9026A. For details, refer to the manual of Flashpro III/Flashpro IV.

Table 13-3. Pin Connection List

Signal Name	I/O	Pin Function	Pin Name	3-Wire Serial I/O	UART	Pseudo 3-Wire
VPP1	Output	Write voltage	V <sub>PP</sub>	◎	◎	◎
VPP2	—	—	—	×	×	×
VDD	I/O	V <sub>DD</sub> voltage generation/ voltage monitoring	V <sub>DD0</sub> , V <sub>DD1</sub>	◎ <sup>Note</sup>	◎ <sup>Note</sup>	◎ <sup>Note</sup>
GND	—	Ground	V <sub>SS0</sub> , V <sub>SS1</sub>	◎	◎	◎
CLK	Output	Clock output	X1	○	○	○
RESET	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎	◎
SI	Input	Receive signal	SO0/TxD/P01/P41	◎	◎	◎
SO	Output	Transmit signal	SI0/RxD/P02/P42	◎	◎	◎
SCK	Output	Transfer clock	$\overline{\text{SCK0}}$ /P00/P40	◎	×	◎
HS	Input	Handshake signal	—	×	×	×

**Note** V<sub>DD</sub> voltage must be supplied before programming is started.

**Remark** ◎: Pin must be connected.

○: If the signal is supplied on the target board, pin does not need to be connected.

×: Pin does not need to be connected.

### 13.1.3 On-board pin processing

When performing programming on the target system, provide a connector on the target system to connect the dedicated flash programmer.

An on-board function that allows switching between normal operation mode and flash memory programming mode may be required in some cases.

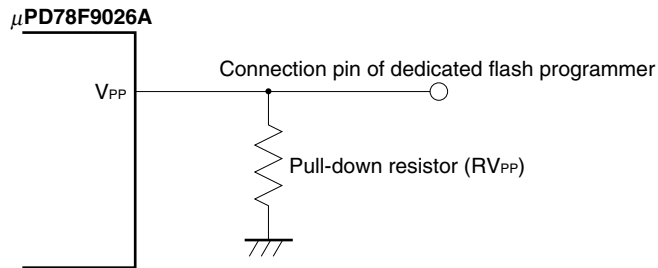
#### <V<sub>PP</sub> pin>

In normal operation mode, input 0 V to the V<sub>PP</sub> pin. In flash memory programming mode, a write voltage of 10.0 V (TYP.) is supplied to the V<sub>PP</sub> pin, so perform either of the following.

- (1) Connect a pull-down resistor (R<sub>VPP</sub> = 10 k $\Omega$ ) to the V<sub>PP</sub> pin.
- (2) Use the jumper on the board to switch the V<sub>PP</sub> pin input to either the writer or directly to GND.

A V<sub>PP</sub> pin connection example is shown below.

**Figure 13-4. V<sub>PP</sub> Pin Connection Example**



#### <Serial interface pin>

The following shows the pins used by the serial interface.

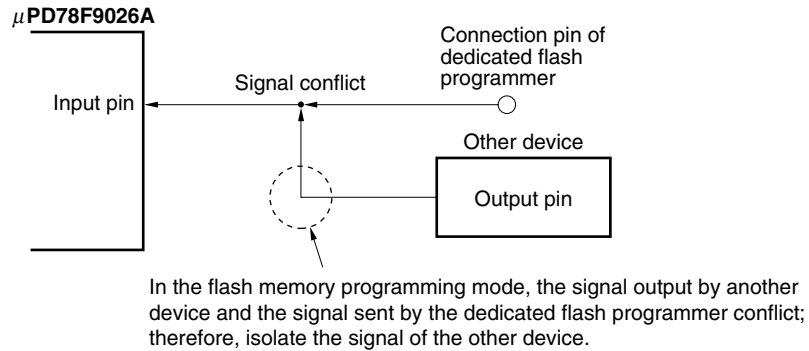
Serial Interface	Pins Used
3-wire serial I/O	SI0, SO0, $\overline{\text{SCK0}}$
UART	RxD, TxD
Pseudo 3-wire	P00, P01, P02
	P40, P41, P42

When connecting the dedicated flash programmer to a serial interface pin that is connected to another device on-board, signal conflict or abnormal operation of the other device may occur. Care must therefore be taken with such connections.

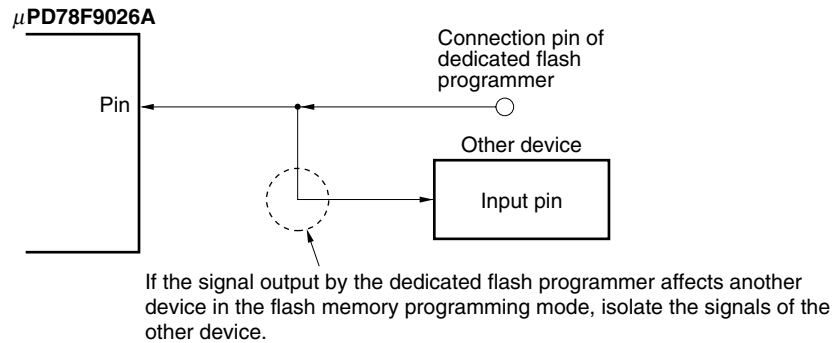
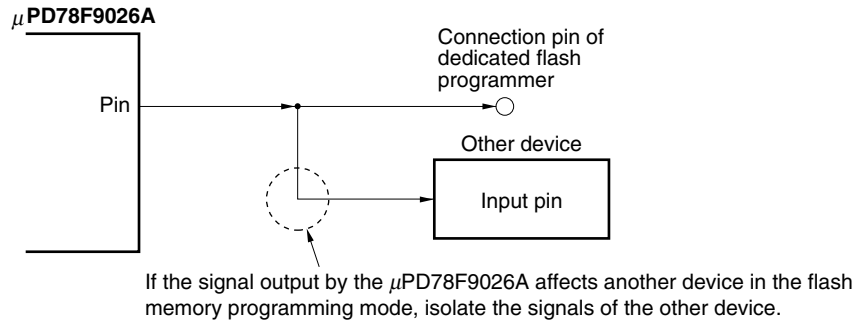


**(1) Signal conflict**

If the dedicated flash programmer (output) is connected to a serial interface pin (input) that is connected to another device (output), a signal conflict occurs. To prevent this, isolate the connection with the other device or set the other device to the output high impedance status.

**Figure 13-5. Signal Conflict (Input Pin of Serial Interface)****(2) Abnormal operation of other device**

If the dedicated flash programmer (output or input) is connected to a serial interface pin (input or output) that is connected to another device (input), a signal is output to the device, and this may cause an abnormal operation. To prevent this abnormal operation, isolate the connection with the other device or set so that the input signals to the other device are ignored.

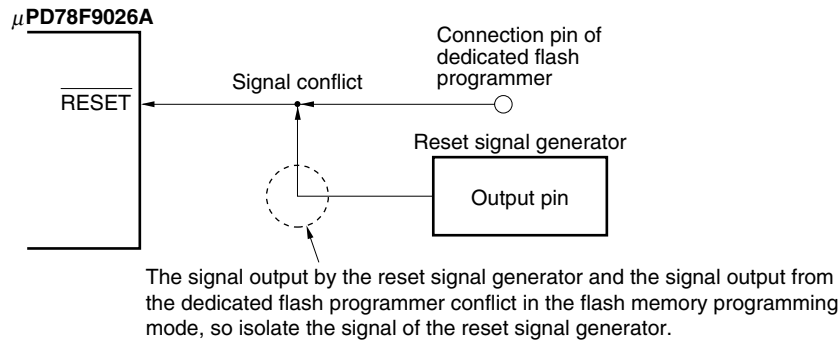
**Figure 13-6. Abnormal Operation of Other Device**

## &lt;RESET pin&gt;

If the reset signal of the dedicated flash programmer is connected to the  $\overline{\text{RESET}}$  pin connected to the reset signal generator on-board, a signal conflict occurs. To prevent this, isolate the connection with the reset signal generator.

If the reset signal is input from the user system in the flash memory programming mode, a normal programming operation cannot be performed. Therefore, do not input reset signals from other than the dedicated flash programmer.

**Figure 13-7. Signal Conflict ( $\overline{\text{RESET}}$  Pin)**



## &lt;Port pins&gt;

When the  $\mu$ PD78F9026A enters the flash memory programming mode, all the pins other than those that communicate with flash programmer are in the same status as immediately after reset.

If the external device does not recognize initial statuses such as the output high impedance status, therefore, connect the external device to  $V_{DD0}$ ,  $V_{DD1}$ ,  $V_{SS0}$ , or  $V_{SS1}$  via a resistor.

## &lt;Resonator&gt;

When using the on-board clock, connect X1 and X2 as required in the normal operation mode.

When using the clock output of the flash programmer, connect it directly to X1, disconnecting the main resonator on-board, and leave the X2 pin open.

## &lt;Power supply&gt;

To use the power output from the flash programmer, connect the  $V_{DD0}$  or  $V_{DD1}$  pin to VDD of the flash programmer, and  $V_{SS0}$  or  $V_{SS1}$  pin to GND of the flash programmer.

To use the on-board power supply, make connections in accordance with the normal operation mode. However, because the voltage is monitored by the flash programmer, be sure to connect VDD of the flash programmer.

### 13.1.4 Connection of adapter for flash writing

The following figure shows an example of recommended connection when the adapter for flash writing is used.

**Figure 13-8. Wiring Example for Flash Writing Adapter with 3-Wire Serial I/O**

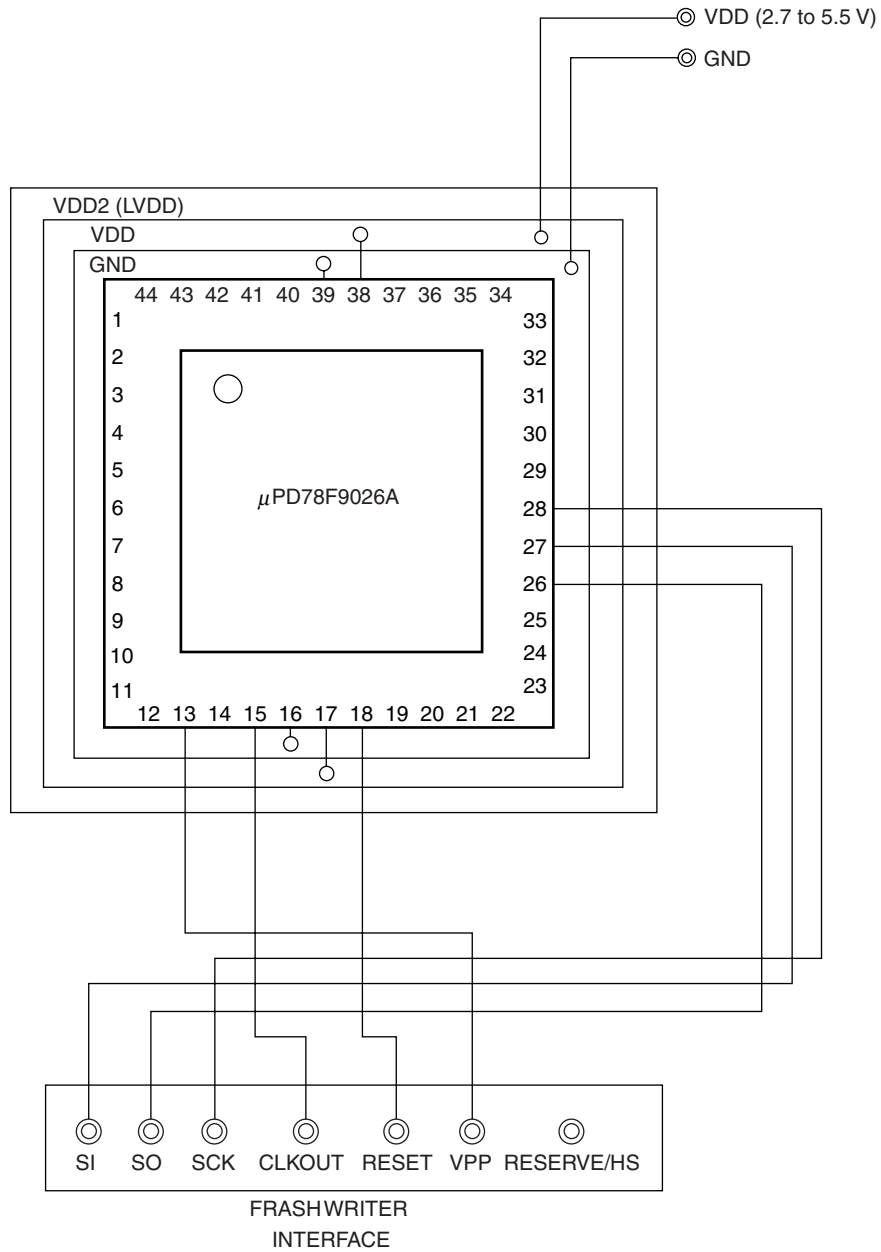
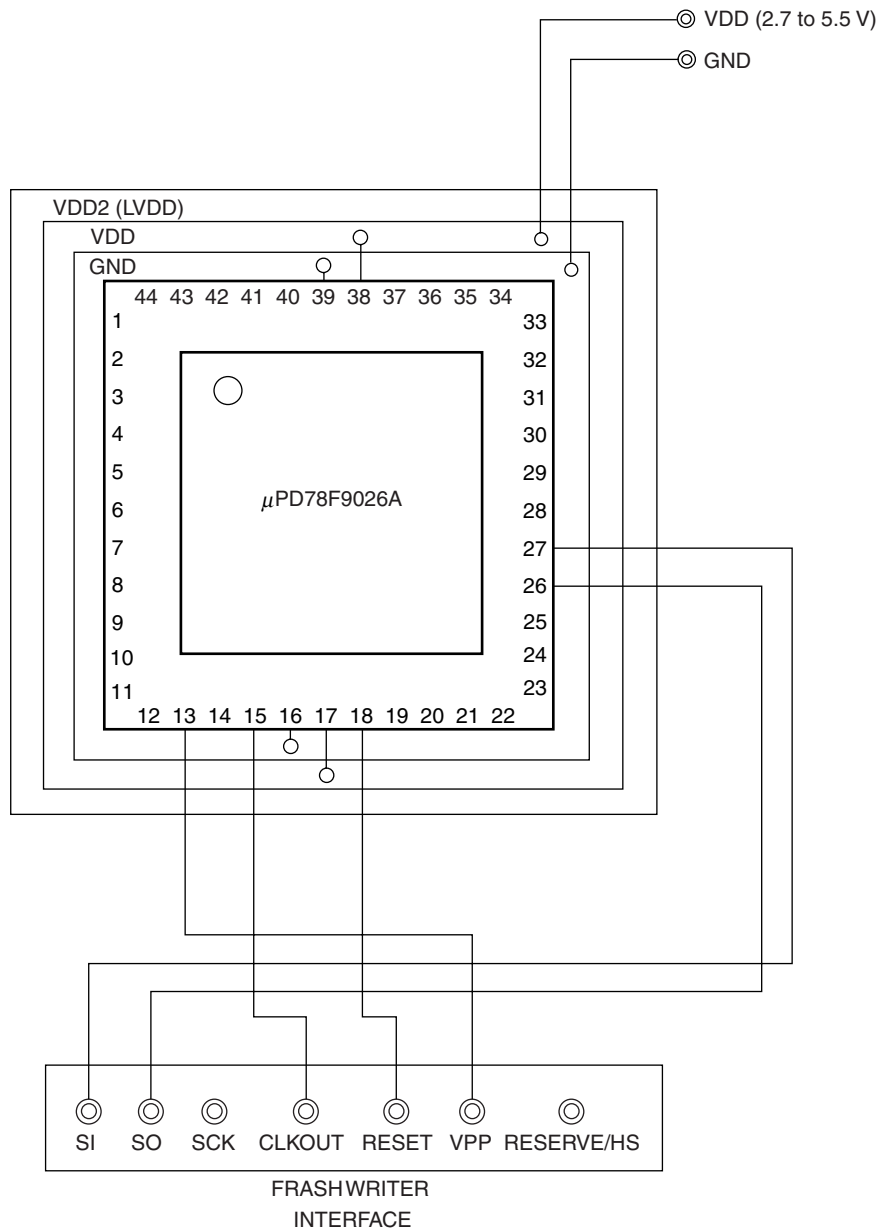
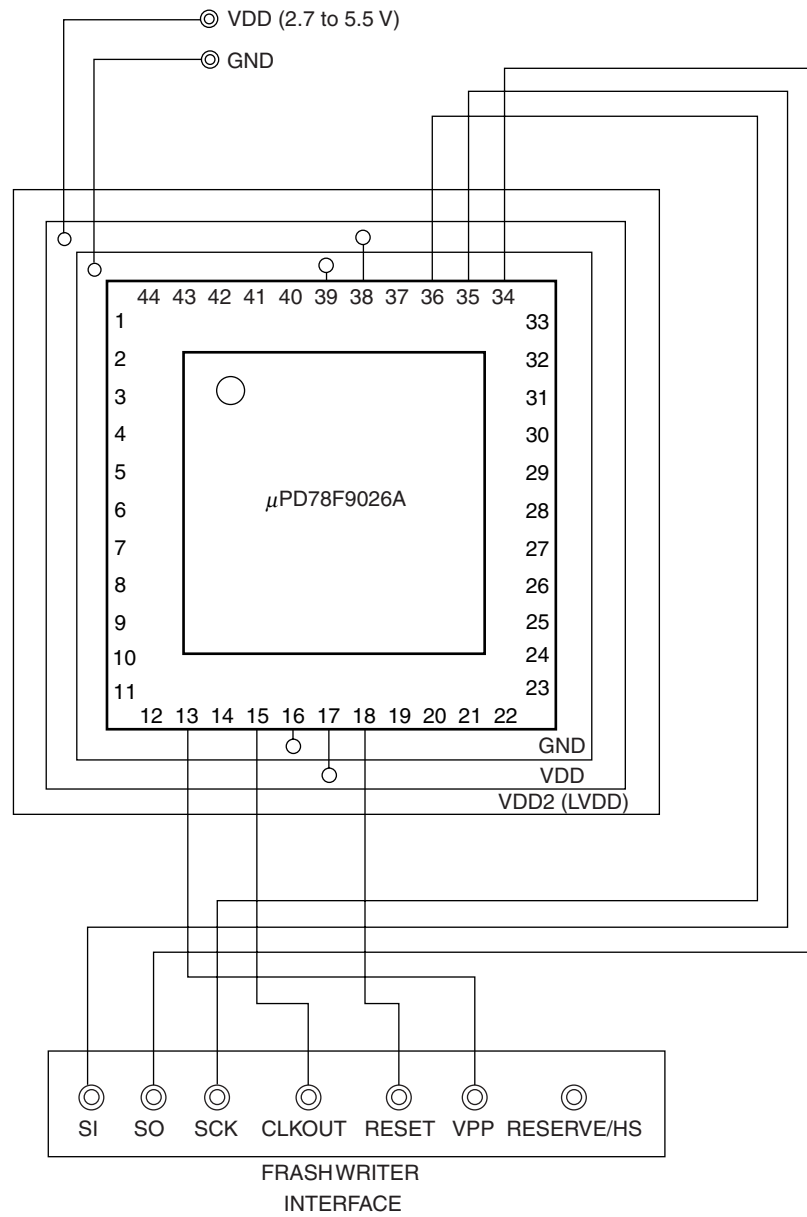


Figure 13-9. Wiring Example for Flash Writing Adapter with UART



**Figure 13-10. Wiring Example for Flash Writing Adapter with Pseudo 3-Wire (When P0 Is Used)**

## CHAPTER 14 INSTRUCTION SET

This chapter lists the instruction set of the  $\mu$ PD789026 Subseries. For details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series Instructions User's Manual (U11047E)**.

### 14.1 Operation

#### 14.1.1 Operand identifiers and writing methods

Operands are written in the Operands column of each instruction in accordance with the writing method of the instruction operand identifier (refer to the assembler specifications for details). When there are two or more writing methods, select one of them. Uppercase letters and the symbols #, !, \$, and [ ] are keywords and are written as they are. Each symbol has the following meaning.

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

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

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

**Table 14-1. Operand Identifiers and Writing Methods**

Identifier	Writing Method
<i>r</i> <i>rp</i> <i>sfr</i>	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7) AX (RP0), BC (RP1), DE (RP2), HL (RP3) Special-function register symbol
<i>saddr</i> <i>saddrp</i>	FE20H to FF1FH Immediate data or label FE20H to FF1FH Immediate data or label (even addresses only)
<i>addr16</i> <i>addr5</i>	0000H to FFFFH Immediate data or label (only even addresses for 16-bit data transfer instructions) 0040H to 007FH Immediate data or label (even addresses only)
<i>word</i> <i>byte</i> <i>bit</i>	16-bit immediate data or label 8-bit immediate data or label 3-bit immediate data or label

**Remark** See **Table 3-4 Special Function Registers** for symbols of special function registers.

**14.1.2 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 parentheses
×H, ×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
jdsp8:	Signed 8-bit data (displacement value)

**14.1.3 Description of “Flag” column**

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

## 14.2 Operation List

Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
MOV	r, #byte	3	6	$r \leftarrow \text{byte}$			
	saddr, #byte	3	6	$(\text{saddr}) \leftarrow \text{byte}$			
	sfr, #byte	3	6	$\text{sfr} \leftarrow \text{byte}$			
	A, r <sup>Note 1</sup>	2	4	$A \leftarrow r$			
	r, A <sup>Note 1</sup>	2	4	$r \leftarrow A$			
	A, saddr	2	4	$A \leftarrow (\text{saddr})$			
	saddr, A	2	4	$(\text{saddr}) \leftarrow A$			
	A, sfr	2	4	$A \leftarrow \text{sfr}$			
	sfr, A	2	4	$\text{sfr} \leftarrow A$			
	A, !addr16	3	8	$A \leftarrow (\text{addr16})$			
	!addr16, A	3	8	$(\text{addr16}) \leftarrow A$			
	PSW, #byte	3	6	$\text{PSW} \leftarrow \text{byte}$	×	×	×
	A, PSW	2	4	$A \leftarrow \text{PSW}$			
	PSW, A	2	4	$\text{PSW} \leftarrow A$	×	×	×
	A, [DE]	1	6	$A \leftarrow (\text{DE})$			
	[DE], A	1	6	$(\text{DE}) \leftarrow A$			
	A, [HL]	1	6	$A \leftarrow (\text{HL})$			
	[HL], A	1	6	$(\text{HL}) \leftarrow A$			
	A, [HL+byte]	2	6	$A \leftarrow (\text{HL}+\text{byte})$			
	[HL+byte], A	2	6	$(\text{HL}+\text{byte}) \leftarrow A$			
XCH	A, X	1	4	$A \leftrightarrow X$			
	A, r <sup>Note 2</sup>	2	6	$A \leftrightarrow r$			
	A, saddr	2	6	$A \leftrightarrow (\text{saddr})$			
	A, sfr	2	6	$A \leftrightarrow \text{sfr}$			
	A, [DE]	1	8	$A \leftrightarrow (\text{DE})$			
	A, [HL]	1	8	$A \leftrightarrow (\text{HL})$			
	A, [HL+byte]	2	8	$A \leftrightarrow (\text{HL}+\text{byte})$			

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

**2.** Except  $r = A, X$ .

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



Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
MOVW	rp, #word	3	6	$rp \leftarrow \text{word}$			
	AX, saddrp	2	6	$AX \leftarrow (\text{saddrp})$			
	saddrp, AX	2	8	$(\text{saddrp}) \leftarrow AX$			
	AX, rp <sup>Note</sup>	1	4	$AX \leftarrow rp$			
	rp, AX <sup>Note</sup>	1	4	$rp \leftarrow AX$			
XCHW	AX, rp <sup>Note</sup>	1	8	$AX \leftrightarrow rp$			
ADD	A, #byte	2	4	$A, CY \leftarrow A + \text{byte}$	×	×	×
	saddr, #byte	3	6	$(\text{saddr}), CY \leftarrow (\text{saddr}) + \text{byte}$	×	×	×
	A, r	2	4	$A, CY \leftarrow A + r$	×	×	×
	A, saddr	2	4	$A, CY \leftarrow A + (\text{saddr})$	×	×	×
	A, !addr16	3	8	$A, CY \leftarrow A + (\text{addr16})$	×	×	×
	A, [HL]	1	6	$A, CY \leftarrow A + (\text{HL})$	×	×	×
	A, [HL+byte]	2	6	$A, CY \leftarrow A + (\text{HL} + \text{byte})$	×	×	×
ADDC	A, #byte	2	4	$A, CY \leftarrow A + \text{byte} + CY$	×	×	×
	saddr, #byte	3	6	$(\text{saddr}), CY \leftarrow (\text{saddr}) + \text{byte} + CY$	×	×	×
	A, r	2	4	$A, CY \leftarrow A + r + CY$	×	×	×
	A, saddr	2	4	$A, CY \leftarrow A + (\text{saddr}) + CY$	×	×	×
	A, !addr16	3	8	$A, CY \leftarrow A + (\text{addr16}) + CY$	×	×	×
	A, [HL]	1	6	$A, CY \leftarrow A + (\text{HL}) + CY$	×	×	×
	A, [HL+byte]	2	6	$A, CY \leftarrow A + (\text{HL} + \text{byte}) + CY$	×	×	×
SUB	A, #byte	2	4	$A, CY \leftarrow A - \text{byte}$	×	×	×
	saddr, #byte	3	6	$(\text{saddr}), CY \leftarrow (\text{saddr}) - \text{byte}$	×	×	×
	A, r	2	4	$A, CY \leftarrow A - r$	×	×	×
	A, saddr	2	4	$A, CY \leftarrow A - (\text{saddr})$	×	×	×
	A, !addr16	3	8	$A, CY \leftarrow A - (\text{addr16})$	×	×	×
	A, [HL]	1	6	$A, CY \leftarrow A - (\text{HL})$	×	×	×
	A, [HL+byte]	2	6	$A, CY \leftarrow A - (\text{HL} + \text{byte})$	×	×	×

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

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

Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
SUBC	A, #byte	2	4	$A, CY \leftarrow A - \text{byte} - CY$	×	×	×
	saddr, #byte	3	6	$(saddr), CY \leftarrow (saddr) - \text{byte} - CY$	×	×	×
	A, r	2	4	$A, CY \leftarrow A - r - CY$	×	×	×
	A, saddr	2	4	$A, CY \leftarrow A - (saddr) - CY$	×	×	×
	A, !addr16	3	8	$A, CY \leftarrow A - (\text{addr16}) - CY$	×	×	×
	A, [HL]	1	6	$A, CY \leftarrow A - (HL) - CY$	×	×	×
	A, [HL+byte]	2	6	$A, CY \leftarrow A - (HL+\text{byte}) - CY$	×	×	×
AND	A, #byte	2	4	$A \leftarrow A \wedge \text{byte}$	×		
	saddr, #byte	3	6	$(saddr) \leftarrow (saddr) \wedge \text{byte}$	×		
	A, r	2	4	$A \leftarrow A \wedge r$	×		
	A, saddr	2	4	$A \leftarrow A \wedge (saddr)$	×		
	A, !addr16	3	8	$A \leftarrow A \wedge (\text{addr16})$	×		
	A, [HL]	1	6	$A \leftarrow A \wedge (HL)$	×		
	A, [HL+byte]	2	6	$A \leftarrow A \wedge (HL+\text{byte})$	×		
OR	A, #byte	2	4	$A \leftarrow A \vee \text{byte}$	×		
	saddr, #byte	3	6	$(saddr) \leftarrow (saddr) \vee \text{byte}$	×		
	A, r	2	4	$A \leftarrow A \vee r$	×		
	A, saddr	2	4	$A \leftarrow A \vee (saddr)$	×		
	A, !addr16	3	8	$A \leftarrow A \vee (\text{addr16})$	×		
	A, [HL]	1	6	$A \leftarrow A \vee (HL)$	×		
	A, [HL+byte]	2	6	$A \leftarrow A \vee (HL+\text{byte})$	×		
XOR	A, #byte	2	4	$A \leftarrow A \nabla \text{byte}$	×		
	saddr, #byte	3	6	$(saddr) \leftarrow (saddr) \nabla \text{byte}$	×		
	A, r	2	4	$A \leftarrow A \nabla r$	×		
	A, saddr	2	4	$A \leftarrow A \nabla (saddr)$	×		
	A, !addr16	3	8	$A \leftarrow A \nabla (\text{addr16})$	×		
	A, [HL]	1	6	$A \leftarrow A \nabla (HL)$	×		
	A, [HL+byte]	2	6	$A \leftarrow A \nabla (HL+\text{byte})$	×		

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

Mnemonic	Operands	Bytes	Clocks	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 <sub>7</sub> ← A <sub>0</sub> , A <sub>m-1</sub> ← A <sub>m</sub> ) × 1			×
ROL	A, 1	1	2	(CY, A <sub>0</sub> ← A <sub>7</sub> , A <sub>m+1</sub> ← A <sub>m</sub> ) × 1			×
RORC	A, 1	1	2	(CY ← A <sub>0</sub> , A <sub>7</sub> ← CY, A <sub>m-1</sub> ← A <sub>m</sub> ) × 1			×
ROLC	A, 1	1	2	(CY ← A <sub>7</sub> , A <sub>0</sub> ← CY, A <sub>m+1</sub> ← A <sub>m</sub> ) × 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<sub>cpu</sub>) selected by the processor clock control register (PCC).

Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
CALL	!addr16	3	6	$(SP-1) \leftarrow (PC+3)_H$ , $(SP-2) \leftarrow (PC+3)_L$ , $PC \leftarrow \text{addr16}$ , $SP \leftarrow SP - 2$			
CALLT	[addr5]	1	8	$(SP-1) \leftarrow (PC+1)_H$ , $(SP-2) \leftarrow (PC+1)_L$ , $PC_H \leftarrow (00000000, \text{addr5}+1)$ , $PC_L \leftarrow (00000000, \text{addr5})$ , $SP \leftarrow SP - 2$			
RET		1	6	$PC_H \leftarrow (SP+1)$ , $PC_L \leftarrow (SP)$ , $SP \leftarrow SP + 2$			
RETI		1	8	$PC_H \leftarrow (SP+1)$ , $PC_L \leftarrow (SP)$ , $PSW \leftarrow (SP+2)$ , $SP \leftarrow SP + 3$ , $NMIS \leftarrow 0$	R	R	R
PUSH	PSW	1	2	$(SP-1) \leftarrow PSW$ , $SP \leftarrow SP - 1$			
	rp	1	4	$(SP-1) \leftarrow rp_H$ , $(SP-2) \leftarrow rp_L$ , $SP \leftarrow SP - 2$			
POP	PSW	1	4	$PSW \leftarrow (SP)$ , $SP \leftarrow SP + 1$	R	R	R
	rp	1	6	$rp_H \leftarrow (SP+1)$ , $rp_L \leftarrow (SP)$ , $SP \leftarrow SP + 2$			
MOVW	SP, AX	2	8	$SP \leftarrow AX$			
	AX, SP	2	6	$AX \leftarrow SP$			
BR	!addr16	3	6	$PC \leftarrow \text{addr16}$			
	\$addr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$			
	AX	1	6	$PC_H \leftarrow A$ , $PC_L \leftarrow X$			
BC	\$saddr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $CY = 1$			
BNC	\$saddr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $CY = 0$			
BZ	\$saddr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $Z = 1$			
BNZ	\$saddr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$ if $Z = 0$			
BT	saddr.bit, \$addr16	4	10	$PC \leftarrow PC + 4 + \text{jdisp8}$ if (saddr.bit) = 1			
	sfr.bit, \$addr16	4	10	$PC \leftarrow PC + 4 + \text{jdisp8}$ if sfr.bit = 1			
	A.bit, \$addr16	3	8	$PC \leftarrow PC + 3 + \text{jdisp8}$ if A.bit = 1			
	PSW.bit, \$addr16	4	10	$PC \leftarrow PC + 4 + \text{jdisp8}$ if PSW.bit = 1			
BF	saddr.bit, \$addr16	4	10	$PC \leftarrow PC + 4 + \text{jdisp8}$ if (saddr.bit) = 0			
	sfr.bit, \$addr16	4	10	$PC \leftarrow PC + 4 + \text{jdisp8}$ if sfr.bit = 0			
	A.bit, \$addr16	3	8	$PC \leftarrow PC + 3 + \text{jdisp8}$ if A.bit = 0			
	PSW.bit, \$addr16	4	10	$PC \leftarrow PC + 4 + \text{jdisp8}$ if PSW.bit = 0			
DBNZ	B, \$addr16	2	6	$B \leftarrow B - 1$ , then $PC \leftarrow PC + 2 + \text{jdisp8}$ if $B \neq 0$			
	C, \$addr16	2	6	$C \leftarrow C - 1$ , then $PC \leftarrow PC + 2 + \text{jdisp8}$ if $C \neq 0$			
	saddr, \$addr16	3	8	$(\text{saddr}) \leftarrow (\text{saddr}) - 1$ , then $PC \leftarrow PC + 3 + \text{jdisp8}$ if $(\text{saddr}) \neq 0$			
NOP		1	2	No Operation			
EI		3	6	$IE \leftarrow 1$ (Enable Interrupt)			
DI		3	6	$IE \leftarrow 0$ (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 the processor clock control register (PCC).

## 14.3 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	laddr16	PSW	[DE]	[HL]	[HL+byte]	\$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV <sup>Note</sup> XCH <sup>Note</sup> 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
laddr16		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 <sup>Note</sup>	saddrp	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	
rp	MOVW	MOVW <sup>Note</sup>				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

**Absolute Maximum Ratings ( $T_A = 25^\circ\text{C}$ )**

Parameter	Symbol	Conditions	Ratings	Unit
Supply voltage	$V_{DD}$		−0.3 to +6.5	V
	$V_{PP}$	$\mu\text{PD78F9026A}$ only <b>Note</b>	−0.3 to +10.5	V
Input voltage	$V_I$		−0.3 to $V_{DD} + 0.3$	V
Output voltage	$V_O$		−0.3 to $V_{DD} + 0.3$	V
Output current, high	$I_{OH}$	Per pin	−10	mA
		Total for all pins	−30	mA
Output current, low	$I_{OL}$	Per pin	30	mA
		Total for all pins	160	mA
Operating ambient temperature	$T_A$	In normal operation mode	−40 to +85	$^\circ\text{C}$
		During flash memory programming	10 to 40	$^\circ\text{C}$
Storage temperature	$T_{stg}$	Mask ROM version	−65 to +150	$^\circ\text{C}$
		$\mu\text{PD78F9026A}$	−40 to +125	$^\circ\text{C}$

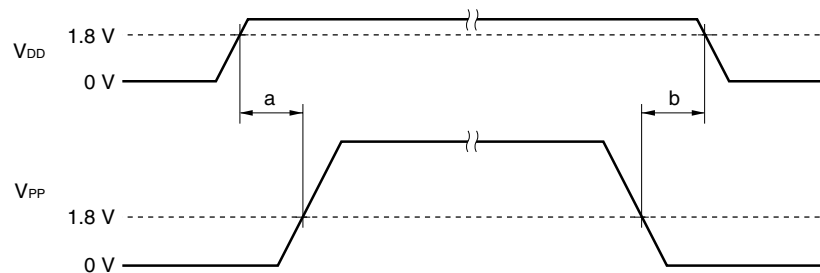
**Note** Make sure that the following conditions of the  $V_{PP}$  voltage application timing are satisfied when the flash memory is written.

- **When supply voltage rises**

$V_{PP}$  must exceed  $V_{DD}$  10  $\mu\text{s}$  or more after  $V_{DD}$  has reached the lower-limit value (1.8 V) of the operating voltage range (see a in the figure below).

- **When supply voltage drops**

$V_{DD}$  must be lowered 10  $\mu\text{s}$  or more after  $V_{PP}$  falls below the lower-limit value (1.8 V) of the operating voltage range of  $V_{DD}$  (see b in the figure below).

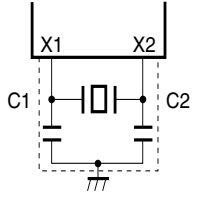
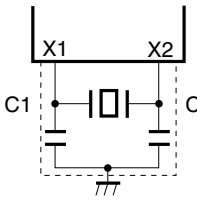
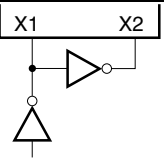
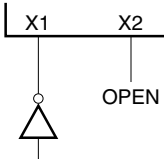


**Caution** Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.



System Clock Oscillator Characteristics ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $V_{DD} = 1.8$  to  $5.5$  V)

Resonator	Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Ceramic resonator		Oscillation frequency ( $f_x$ ) <sup>Note 1</sup>	$V_{DD}$ = Oscillation voltage range	1.0		5.0	MHz
		Oscillation stabilization time <sup>Note 2</sup>	After $V_{DD}$ reaches the oscillation voltage range MIN.			4	ms
Crystal resonator		Oscillation frequency ( $f_x$ ) <sup>Note 1</sup>		1.0		5.0	MHz
		Oscillation stabilization time <sup>Note 2</sup>	$V_{DD} = 4.5$ to $5.5$ V			10	ms
			$V_{DD} = 1.8$ to $5.5$ V			30	
External clock		X1 input frequency ( $f_x$ ) <sup>Note 1</sup>		1.0		5.0	MHz
		X1 input high-/low-level width ( $t_{xH}$ , $t_{xL}$ )		85		500	ns
		X1 input frequency ( $f_x$ ) <sup>Note 1</sup>	$V_{DD} = 2.7$ to $5.5$ V	1.0		5.0	MHz
		X1 input high-/low-level width ( $t_{xH}$ , $t_{xL}$ )	$V_{DD} = 2.7$ to $5.5$ V	85		500	ns

- Notes**
1. Indicates only oscillator characteristics. Refer to **AC Characteristics** for instruction execution time.
  2. Time required to stabilize oscillation after reset or STOP mode release. Use a resonator whose oscillation is stabilized within the oscillation stabilization wait time.

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

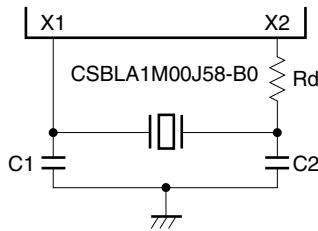
- Keep the wiring length as short as possible.
- Do not cross the wiring with the other signal lines.
- Do not route the wiring near a signal line through which a high fluctuating current flows.
- Always make the ground point of the oscillator capacitor the same potential as  $V_{SS}$ .
- Do not ground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

## Recommended Oscillator Constant

(1) Ceramic resonator ( $T_A = -40$  to  $+85^\circ\text{C}$ ) (mask ROM version)

Manufacturer	Part Number	Frequency (MHz)	Recommended Circuit Constant (pF)		Oscillation Voltage Range ( $V_{DD}$ )		Remarks
			C1	C2	MIN.	MAX.	
Murata Mfg. Co., Ltd.	CSBLA1M00J58-B0 <sup>Note</sup>	1.0	100	100	2.2	5.5	$R_d = 4.7\text{ k}\Omega$
TDK	CCR4.19MC3	4.19	—	—	2.0	5.5	On-chip capacitor
	FCR4.19MC5		—	—			On-chip capacitor
	CCR5.0MC3	5.0	—	—	2.0	5.5	On-chip capacitor
	FCR5.0MC5		—	—			On-chip capacitor

**Note** A limiting resistor ( $R_d = 4.7\text{ k}\Omega$ ) is required when CSBLA1M00J58-B0 (1.0 MHz) manufactured by Murata Mfg. Co., Ltd. is used as the ceramic resonator (see the figure below). This is not necessary when using one of the other recommended resonators.



**Caution** The oscillator constant is a reference value based on evaluation in specific environments by the resonator manufacturer. If the oscillator characteristics need to be optimized in the actual application, request the resonator manufacturer for evaluation on the implementation circuit. Note that the oscillation voltage and oscillation frequency merely indicate the characteristics of the oscillator. Use the internal operation conditions of the  $\mu\text{PD78902x}$  within the specifications of the DC and AC characteristics.

**Remark** The resonators of Murata Mfg. Co., Ltd. other than the above are currently under evaluation.

(2) Ceramic resonator ( $T_A = -40$  to  $+85^\circ\text{C}$ ) ( $\mu\text{PD78F9026A}$ )

Manufacturer	Part Number	Frequency (MHz)	Recommended Circuit Constant (pF)		Oscillation Voltage Range (V <sub>DD</sub> )		Remarks
			C1	C2	MIN.	MAX.	
Murata Mfg. Co., Ltd.	CSBLA1M00J58-B0	1.0	100	100	2.3	5.5	—
	CSBFB1M00J58-R1						
	CST2.00MG	2.0	—	—	2.4	5.5	On-chip capacitor
	CSTLS2M00G56-B0				2.0	5.5	
	CSTCC2M00G56-R0				2.1	5.5	
	CSTLS4M00G53-B0	4.0			2.0	5.5	
	CSTCR4M00G53-R0				2.1	5.5	
	CSTLS4M19G53-B0	4.194			2.0	5.5	
	CSTCR4M19G53-R0				2.2	5.5	
	CSTLS4M91G53-B0	4.915			2.1	5.5	
	CSTCR4M91G53-R0				2.2	5.5	
	CSTLS5M00G53-B0	5.0			2.1	5.5	
	CSTCR5M00G53-R0				2.2	5.5	

**Caution** The oscillator constant is a reference value based on evaluation in specific environments by the resonator manufacturer. If the oscillator characteristics need to be optimized in the actual application, request the resonator manufacturer for evaluation on the implementation circuit. Note that the oscillation voltage and oscillation frequency merely indicate the characteristics of the oscillator. Use the internal operation conditions of the  $\mu\text{PD78F9026A}$  within the specifications of the DC and AC characteristics.

**DC Characteristics ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $V_{DD} = 1.8$  to  $5.5$  V) (1/2)**

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Output current, high	$I_{OH}$	Per pin				-1	mA
		Total for all pins				-15	mA
Output current, low	$I_{OL}$	Per pin				10	mA
		Total for all pins				80	mA
Input voltage, high	$V_{IH1}$	P00 to P07, P10 to P17, P21, P51 to P53	$V_{DD} = 2.7$ to $5.5$ V	$0.7V_{DD}$		$V_{DD}$	V
			$V_{DD} = 1.8$ to $5.5$ V	$0.9V_{DD}$		$V_{DD}$	V
	$V_{IH2}$	P20, P22, P30 to P32, P40 to P47, P50, RESET	$V_{DD} = 2.7$ to $5.5$ V	$0.8V_{DD}$		$V_{DD}$	V
			$V_{DD} = 1.8$ to $5.5$ V	$0.9V_{DD}$		$V_{DD}$	V
	$V_{IH3}$	X1, X2	$V_{DD} = 4.5$ to $5.5$ V	$V_{DD} - 0.5$		$V_{DD}$	V
			$V_{DD} = 1.8$ to $5.5$ V	$V_{DD} - 0.1$		$V_{DD}$	V
Input voltage, low	$V_{IL1}$	P00 to P07, P10 to P17, P21, P51 to P53	$V_{DD} = 2.7$ to $5.5$ V	0		$0.3V_{DD}$	V
			$V_{DD} = 1.8$ to $5.5$ V	0		$0.1V_{DD}$	V
	$V_{IL2}$	P20, P22, P30 to P32, P40 to P47, P50, RESET	$V_{DD} = 2.7$ to $5.5$ V	0		$0.2V_{DD}$	V
			$V_{DD} = 1.8$ to $5.5$ V	0		$0.1V_{DD}$	V
	$V_{IL3}$	X1, X2	$V_{DD} = 4.5$ to $5.5$ V	0		0.4	V
			$V_{DD} = 1.8$ to $5.5$ V	0		0.1	V
Output voltage, high	$V_{OH}$	$V_{DD} = 4.5$ to $5.5$ V, $I_{OH} = -1$ mA		$V_{DD} - 1.0$			V
		$V_{DD} = 1.8$ to $5.5$ V, $I_{OH} = -100$ $\mu\text{A}$		$V_{DD} - 0.5$			V
Output voltage, low	$V_{OL}$	$V_{DD} = 4.5$ to $5.5$ V, $I_{OL} = 10$ mA				1.0	V
		$V_{DD} = 1.8$ to $5.5$ V, $I_{OL} = 400$ $\mu\text{A}$				0.5	V
Input leakage current, high	$I_{LIH1}$	$V_{IN} = V_{DD}$	Pins other than X1 and X2			3	$\mu\text{A}$
	$I_{LIH2}$		X1, X2			20	$\mu\text{A}$
Input leakage current, low	$I_{LIL1}$	$V_{IN} = 0$ V	Pins other than X1 and X2			-3	$\mu\text{A}$
	$I_{LIL2}$		X1, X2			-20	$\mu\text{A}$
Output leakage current, high	$I_{LOH}$	$V_{OUT} = V_{DD}$				3	$\mu\text{A}$
Output leakage current, low	$I_{LOL}$	$V_{OUT} = 0$ V				-3	$\mu\text{A}$
Software pull-up resistor	R	$V_{IN} = 0$ V		50	100	200	k $\Omega$

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

**DC Characteristics ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $V_{DD} = 1.8$  to  $5.5$  V) (2/2)**

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Supply current <sup>Note 1</sup> (mask ROM version)	I <sub>DD1</sub>	5.0 MHz crystal oscillation operating mode	V <sub>DD</sub> = 5.0 V±10% <sup>Note 2</sup>		1.3	2.5	mA
			V <sub>DD</sub> = 3.0 V±10% <sup>Note 3</sup>		0.26	0.45	mA
			V <sub>DD</sub> = 2.0 V±10% <sup>Note 3</sup>		0.14	0.30	mA
	I <sub>DD2</sub>	5.0 MHz crystal oscillation HALT mode	V <sub>DD</sub> = 5.0 V±10% <sup>Note 2</sup>		0.41	0.85	mA
			V <sub>DD</sub> = 3.0 V±10% <sup>Note 3</sup>		0.16	0.35	mA
			V <sub>DD</sub> = 2.0 V±10% <sup>Note 3</sup>		0.07	0.15	mA
	I <sub>DD3</sub>	STOP mode	V <sub>DD</sub> = 5.0 V±10%		0.1	10	μA
			V <sub>DD</sub> = 3.0 V±10%		0.05	5.0	μA
			T <sub>A</sub> = 25°C		0.05	3.0	μA
			V <sub>DD</sub> = 2.0 V±10%		0.05	3.0	μA
Supply current <sup>Note 1</sup> (μPD78F9026A)	I <sub>DD1</sub>	5.0 MHz crystal oscillation operating mode (C1 = C2 = 22 pF)	V <sub>DD</sub> = 5.0 V±10% <sup>Note 2</sup>		4.0	15.0	mA
			V <sub>DD</sub> = 3.0 V±10% <sup>Note 3</sup>		1.0	5.0	mA
			V <sub>DD</sub> = 2.0 V±10% <sup>Note 3</sup>		0.8	3.0	mA
	I <sub>DD2</sub>	5.0 MHz crystal oscillation HALT mode (C1 = C2 = 22 pF)	V <sub>DD</sub> = 5.0 V±10% <sup>Note 2</sup>		0.8	5.0	mA
			V <sub>DD</sub> = 3.0 V±10% <sup>Note 3</sup>		0.5	2.5	mA
			V <sub>DD</sub> = 2.0 V±10% <sup>Note 3</sup>		0.3	1.0	mA
	I <sub>DD3</sub>	STOP mode	V <sub>DD</sub> = 5.0 V±10%		0.1	30	μA
			V <sub>DD</sub> = 3.0 V±10%		0.05	10	μA
			V <sub>DD</sub> = 2.0 V±10%		0.05	10	μA

- Notes**
1. The current flowing to the ports (including the current flowing through the on-chip pull-up resistors) is not included.
  2. High-speed mode operation (when the processor clock control register (PCC) is set to 00H)
  3. Low-speed mode operation (when PCC is set to 02H)

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

**Flash Memory Write/Erase Characteristics**

( $T_A = 10$  to  $40^\circ\text{C}$ ,  $V_{DD} = 1.8$  to  $5.5$  V, in 5.0 MHz crystal oscillation operating mode)

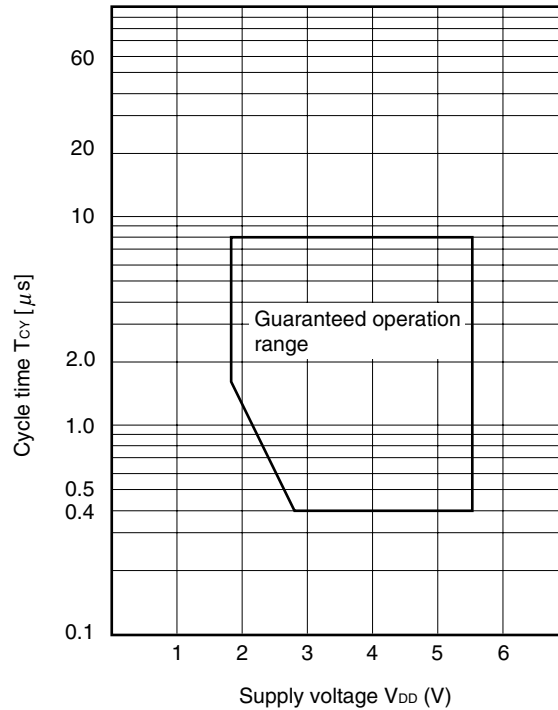
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Write current <sup>Note</sup> ( $V_{DD}$ pin)	$I_{DDW}$	When $V_{PP}$ supply voltage = $V_{PP1}$			18	mA
Write current <sup>Note</sup> ( $V_{PP}$ pin)	$I_{PPW}$	When $V_{PP}$ supply voltage = $V_{PP1}$			22.5	mA
Erase current <sup>Note</sup> ( $V_{DD}$ pin)	$I_{DDE}$	When $V_{PP}$ supply voltage = $V_{PP1}$			18	mA
Erase current <sup>Note</sup> ( $V_{PP}$ pin)	$I_{PPE}$	When $V_{PP}$ supply voltage = $V_{PP1}$			115	mA
Unit erase time	$t_{er}$		0.5	1	1	s
Total erase time	$t_{era}$				20	s
Write count		Erase/write are regarded as 1 cycle	20	20	20	Times
$V_{PP}$ supply voltage	$V_{PP0}$	In normal operation	0		$0.2V_{DD}$	V
	$V_{PP1}$	During flash memory programming	9.7	10.0	10.3	V

**Note** The current flowing to the ports (including the current flowing through the on-chip pull-up resistors) is not included.

## AC Characteristics

(1) Basic operation ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $V_{DD} = 1.8$  to  $5.5$  V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Cycle time (minimum instruction execution time)	$T_{CY}$	$V_{DD} = 2.7$ to $5.5$ V	0.4		8	$\mu\text{s}$
		$V_{DD} = 1.8$ to $5.5$ V	1.6		8	$\mu\text{s}$
TIO input high-/low- level width	$t_{TIH}, t_{TIL}$	$V_{DD} = 2.7$ to $5.5$ V	0.1			$\mu\text{s}$
		$V_{DD} = 1.8$ to $5.5$ V	1.8			$\mu\text{s}$
TIO input frequency	$f_{TI}$	$V_{DD} = 2.7$ to $5.5$ V	0		4	MHz
		$V_{DD} = 1.8$ to $5.5$ V	0		275	kHz
Interrupt input high-/low-level width	$t_{INTH},$ $t_{INTL}$	INTP0 to INTP2	10			$\mu\text{s}$
$\overline{\text{RESET}}$ low-level width	$t_{RSL}$		10			$\mu\text{s}$

 $T_{CY}$  vs  $V_{DD}$  (system clock)

(2) Serial interface 00 ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $V_{DD} = 1.8$  to  $5.5$  V)

(i) 3-wire serial I/O mode ( $\overline{\text{SCK0}}$ ...Internal clock output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{\text{SCK0}}$ cycle time	$t_{\text{KCY1}}$	$V_{DD} = 2.7$ to $5.5$ V	800			ns
		$V_{DD} = 1.8$ to $5.5$ V	3200			ns
$\overline{\text{SCK0}}$ high-/low-level width	$t_{\text{KH1}}, t_{\text{KL1}}$	$V_{DD} = 2.7$ to $5.5$ V	$t_{\text{KCY1}}/2 - 50$			ns
		$V_{DD} = 1.8$ to $5.5$ V	$t_{\text{KCY1}}/2 - 150$			ns
SI0 setup time (to $\overline{\text{SCK0}}\uparrow$ )	$t_{\text{SIK1}}$	$V_{DD} = 2.7$ to $5.5$ V	150			ns
		$V_{DD} = 1.8$ to $5.5$ V	500			ns
SI0 hold time (from $\overline{\text{SCK0}}\uparrow$ )	$t_{\text{KSI1}}$	$V_{DD} = 2.7$ to $5.5$ V	400			ns
		$V_{DD} = 1.8$ to $5.5$ V	600			ns
SO0 output delay time from $\overline{\text{SCK0}}\downarrow$	$t_{\text{KSO1}}$	$R = 1\text{ k}\Omega$ , $C = 100\text{ pF}^{\text{Note}}$	$V_{DD} = 2.7$ to $5.5$ V	0	250	ns
			$V_{DD} = 1.8$ to $5.5$ V	0	1000	ns

**Note** R and C are the load resistance and load capacitance of the SO0 output line, respectively.

(ii) 3-wire serial I/O mode ( $\overline{\text{SCK0}}$ ...External clock input)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{\text{SCK0}}$ cycle time	$t_{\text{KCY2}}$	$V_{DD} = 2.7$ to $5.5$ V	900			ns
		$V_{DD} = 1.8$ to $5.5$ V	3500			ns
$\overline{\text{SCK0}}$ high-/low-level width	$t_{\text{KH2}}, t_{\text{KL2}}$	$V_{DD} = 2.7$ to $5.5$ V	400			ns
		$V_{DD} = 1.8$ to $5.5$ V	1600			ns
SI0 setup time (to $\overline{\text{SCK0}}\uparrow$ )	$t_{\text{SIK2}}$	$V_{DD} = 2.7$ to $5.5$ V	100			ns
		$V_{DD} = 1.8$ to $5.5$ V	150			ns
SI0 hold time (from $\overline{\text{SCK0}}\uparrow$ )	$t_{\text{KSI2}}$	$V_{DD} = 2.7$ to $5.5$ V	400			ns
		$V_{DD} = 1.8$ to $5.5$ V	600			ns
SO0 output delay time from $\overline{\text{SCK0}}\downarrow$	$t_{\text{KSO2}}$	$R = 1\text{ k}\Omega$ , $C = 100\text{ pF}^{\text{Note}}$	$V_{DD} = 2.7$ to $5.5$ V	0	300	ns
			$V_{DD} = 1.8$ to $5.5$ V	0	1000	ns

**Note** R and C are the load resistance and load capacitance of the SO0 output line, respectively.

(iii) UART mode (dedicated baud rate generator output)

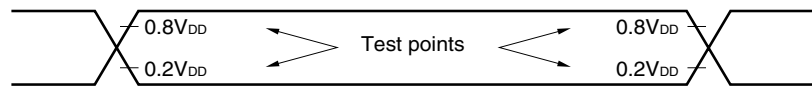
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate		$V_{DD} = 2.7$ to $5.5$ V			78125	bps
		$V_{DD} = 1.8$ to $5.5$ V			19531	bps

**(iv) UART mode (external clock input)**

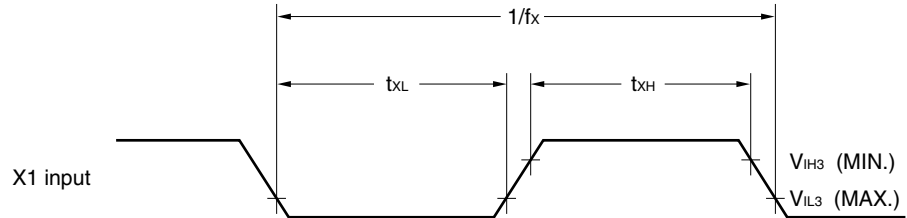
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
ASCK cycle time	$t_{KCY3}$	$V_{DD} = 2.7 \text{ to } 5.5 \text{ V}$	900			ns
		$V_{DD} = 1.8 \text{ to } 5.5 \text{ V}$	3500			ns
ASCK high-/low-level width	$t_{KH3}, t_{KL3}$	$V_{DD} = 2.7 \text{ to } 5.5 \text{ V}$	400			ns
		$V_{DD} = 1.8 \text{ to } 5.5 \text{ V}$	1600			ns
Transfer rate		$V_{DD} = 2.7 \text{ to } 5.5 \text{ V}$			39063	bps
		$V_{DD} = 1.8 \text{ to } 5.5 \text{ V}$			9766	bps
ASCK rise/fall time	$t_R, t_F$				1	$\mu\text{s}$



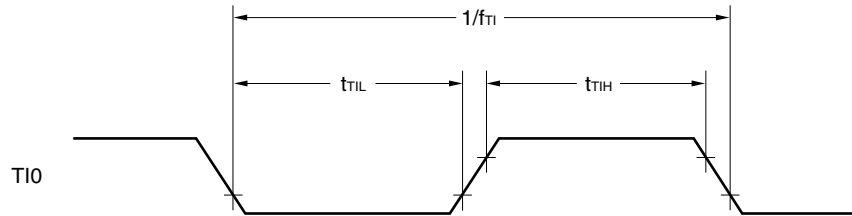
### AC Timing Test Points (Except X1 Input)



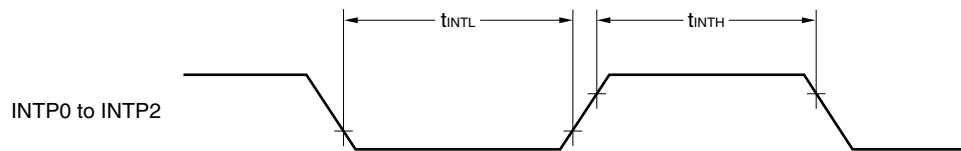
### Clock Timing



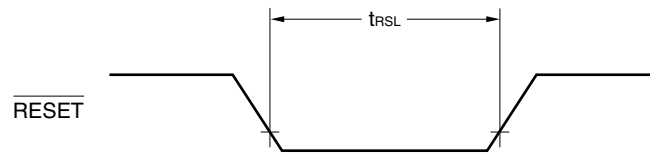
### TI Timing



### Interrupt Input Timing

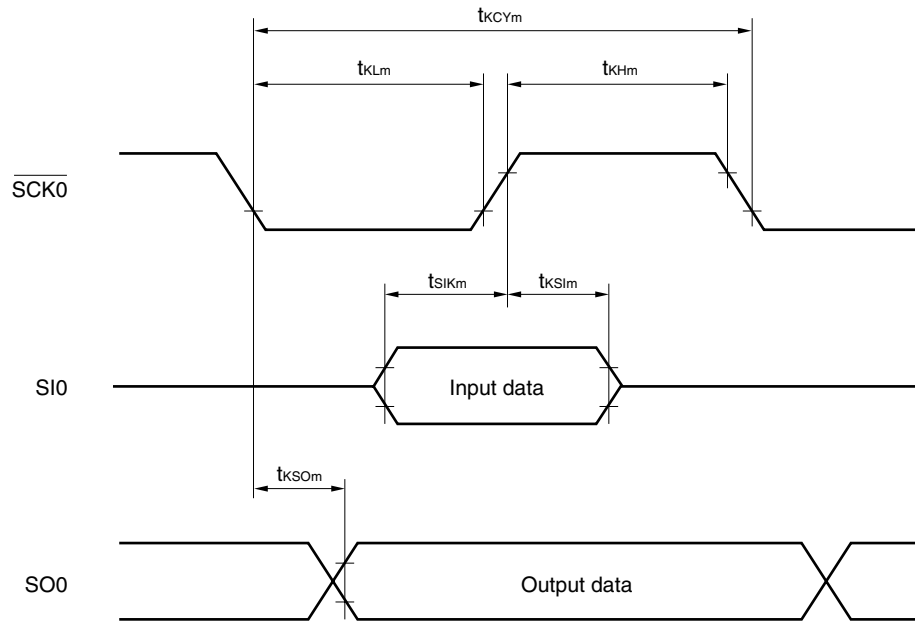


### RESET Input Timing



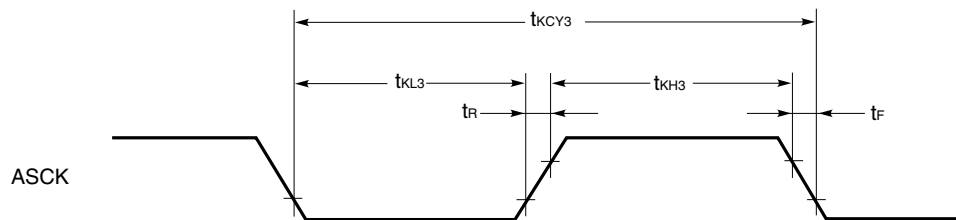
## Serial Transfer Timing

### 3-wire serial I/O mode:



$m = 1, 2$

### UART mode (external clock input):



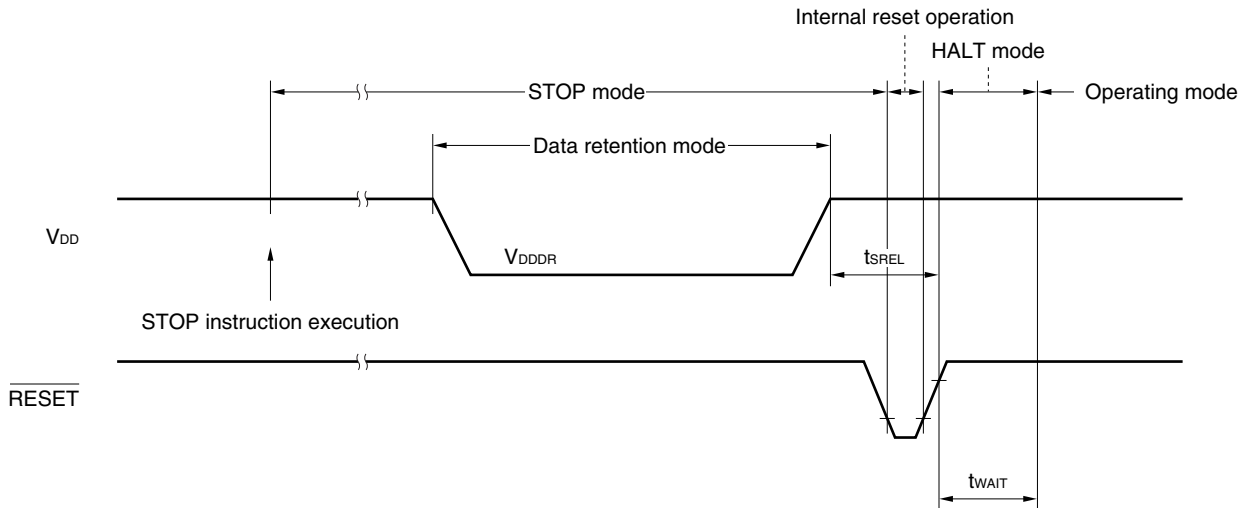
**Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics ( $T_A = -40$  to  $+85^\circ\text{C}$ )**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention supply voltage	$V_{DDDR}$		1.8		5.5	V
Release signal set time	$t_{SREL}$		0			$\mu\text{s}$
Oscillation stabilization wait time	$t_{WAIT}$	Release by $\overline{\text{RESET}}$		$2^{15}/f_x$		ms
		Release by interrupt		<b>Note</b>		ms

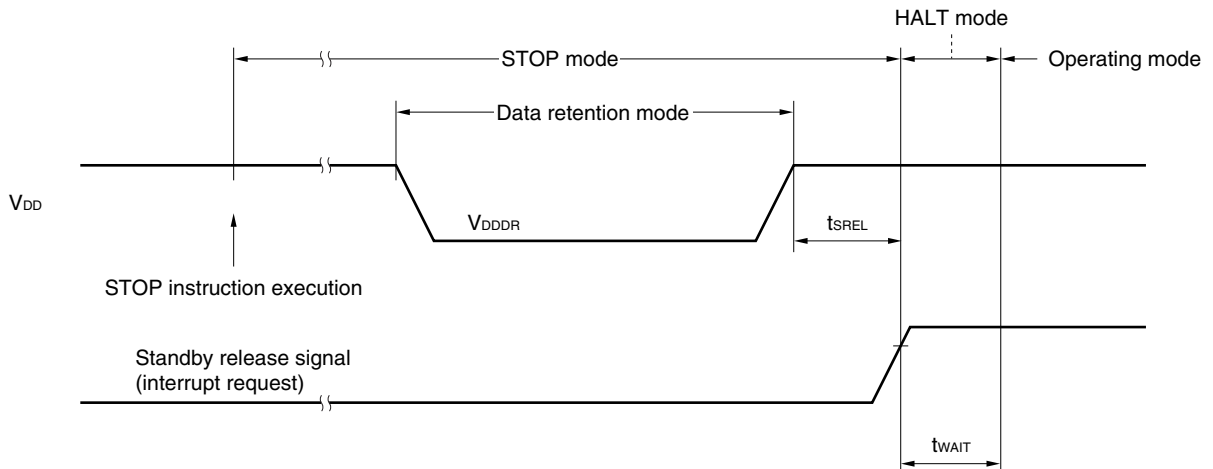
**Note**  $2^{12}/f_x$ ,  $2^{15}/f_x$ , or  $2^{17}/f_x$  can be selected according to the setting of bits 0 to 2 (OSTS0 to OSTS2) of the oscillation stabilization time select register.

**Remark**  $f_x$ : System clock oscillation frequency

**Data Retention Timing (STOP Mode Release by  $\overline{\text{RESET}}$ )**

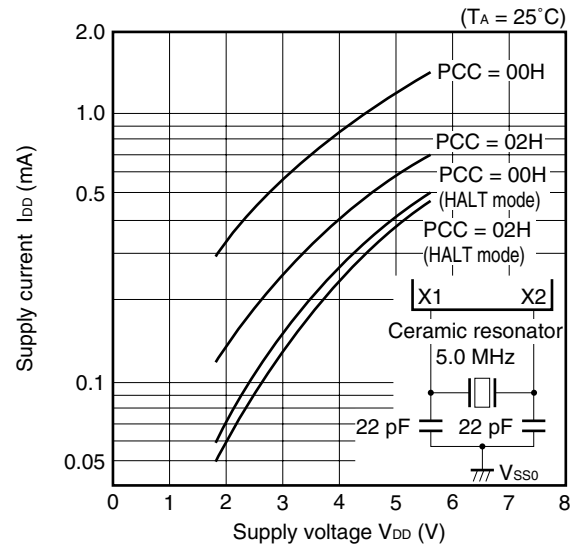


**Data Retention Timing (Standby Release Signal: STOP Mode Release by Interrupt Signal)**

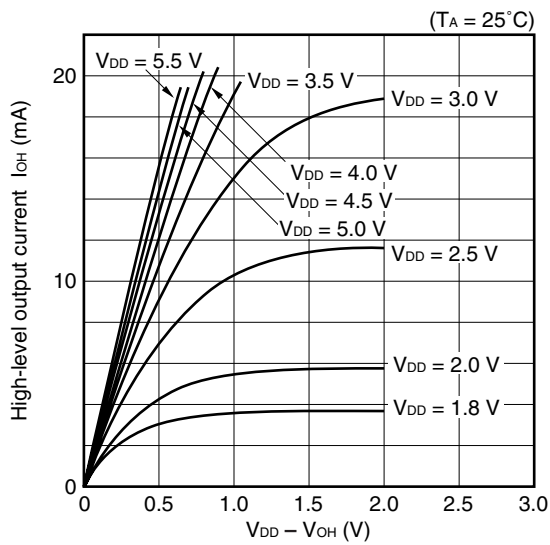


# CHAPTER 16 CHARACTERISTICS CURVES (MASK ROM VERSION)

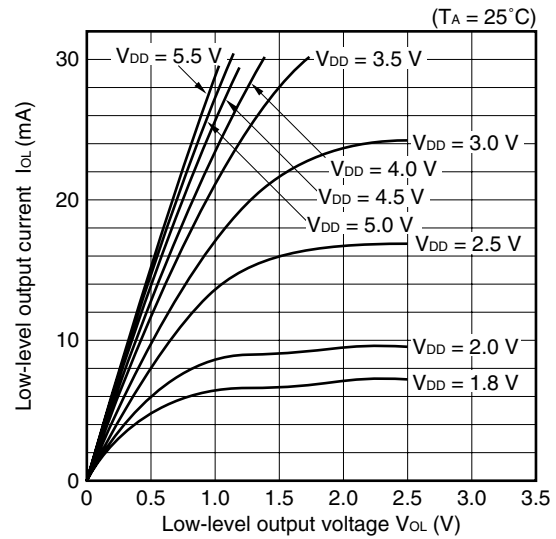
**$I_{DD}$  vs  $V_{DD}$  (system clock: 5.0 MHz, crystal resonator)**



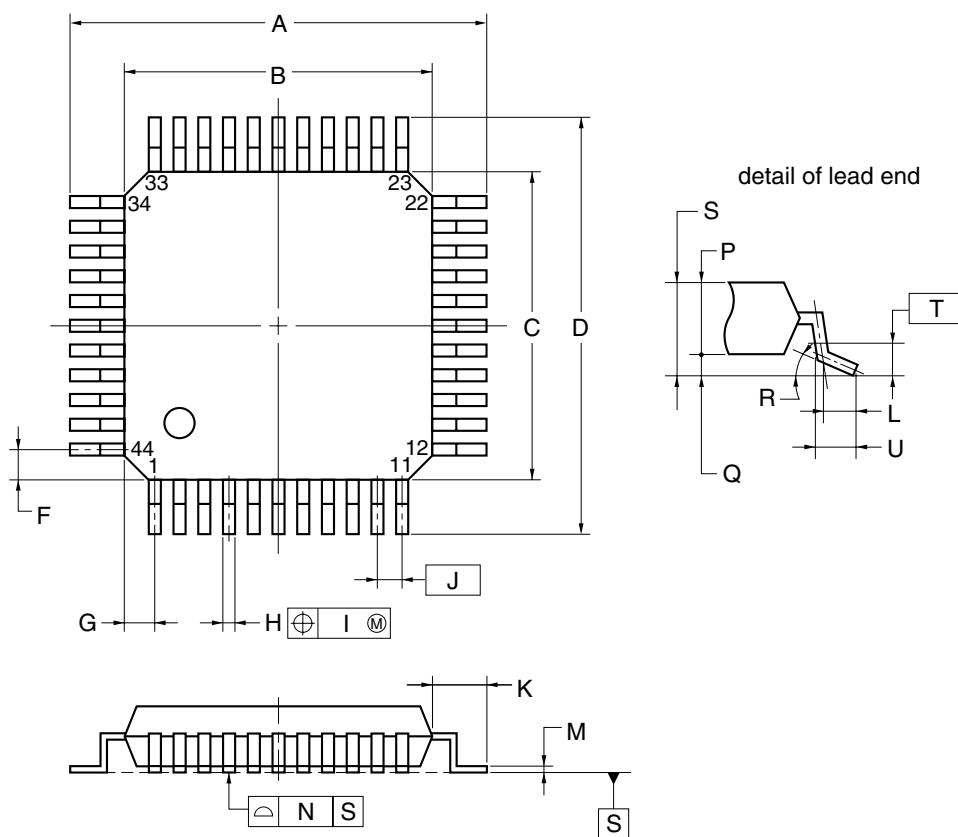
**$I_{OH}$  vs  $V_{DD} - V_{OH}$**



**$I_{OL}$  vs  $V_{OL}$**



## 44 PIN PLASTIC LQFP (10x10)

**NOTE**

Each lead centerline is located within 0.20 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	12.0±0.2
B	10.0±0.2
C	10.0±0.2
D	12.0±0.2
F	1.0
G	1.0
H	0.37 <sup>+0.08</sup> <sub>-0.07</sub>
I	0.20
J	0.8 (T.P.)
K	1.0±0.2
L	0.5
M	0.17 <sup>+0.03</sup> <sub>-0.06</sub>
N	0.10
P	1.4±0.05
Q	0.1±0.05
R	3° <sup>+4°</sup> <sub>-3°</sub>
S	1.6 MAX.
T	0.25 (T.P.)
U	0.6±0.15

S44GB-80-8ES-2

## CHAPTER 18 RECOMMENDED SOLDERING CONDITIONS

The  $\mu$ PD789026 Subseries should be soldered and mounted under the following recommended conditions.

For details of the recommended soldering conditions, refer to the document **Semiconductor Device Mounting Technology Manual (C10535E)**.

For soldering methods and conditions other than those recommended below, contact an NEC Electronics sales representative.

**Table 18-1. Surface Mounting Type Soldering Conditions**

$\mu$ PD789022GB-xxx-8ES: 44-pin plastic LQFP (10 × 10)

$\mu$ PD789024GB-xxx-8ES: 44-pin plastic LQFP (10 × 10)

$\mu$ PD789025GB-xxx-8ES: 44-pin plastic LQFP (10 × 10)

$\mu$ PD789026GB-xxx-8ES: 44-pin plastic LQFP (10 × 10)

$\mu$ PD78F9026AGB-8ES: 44-pin plastic LQFP (10 × 10)

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C, Time: 30 seconds max. (at 210°C or higher), Count: Two times or less	IR35-00-2
VPS	Package peak temperature: 215°C, Time: 40 seconds max. (at 200°C or higher), Count: Two times or less	VP15-00-2
Wave soldering	Solder bath temperature: 260°C max., Time: 10 seconds max., Count: Once, Preheating temperature: 120°C max. (package surface temperature)	WS60-00-1
Partial heating	Pin temperature: 300°C max., Time: 3 seconds max. (per pin row)	—

**Caution** Do not use different soldering methods together (except for partial heating).

## APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for development of systems using the  $\mu$ PD789026 Subseries. Figure A-1 shows development tools.

- Support of PC98-NX Series

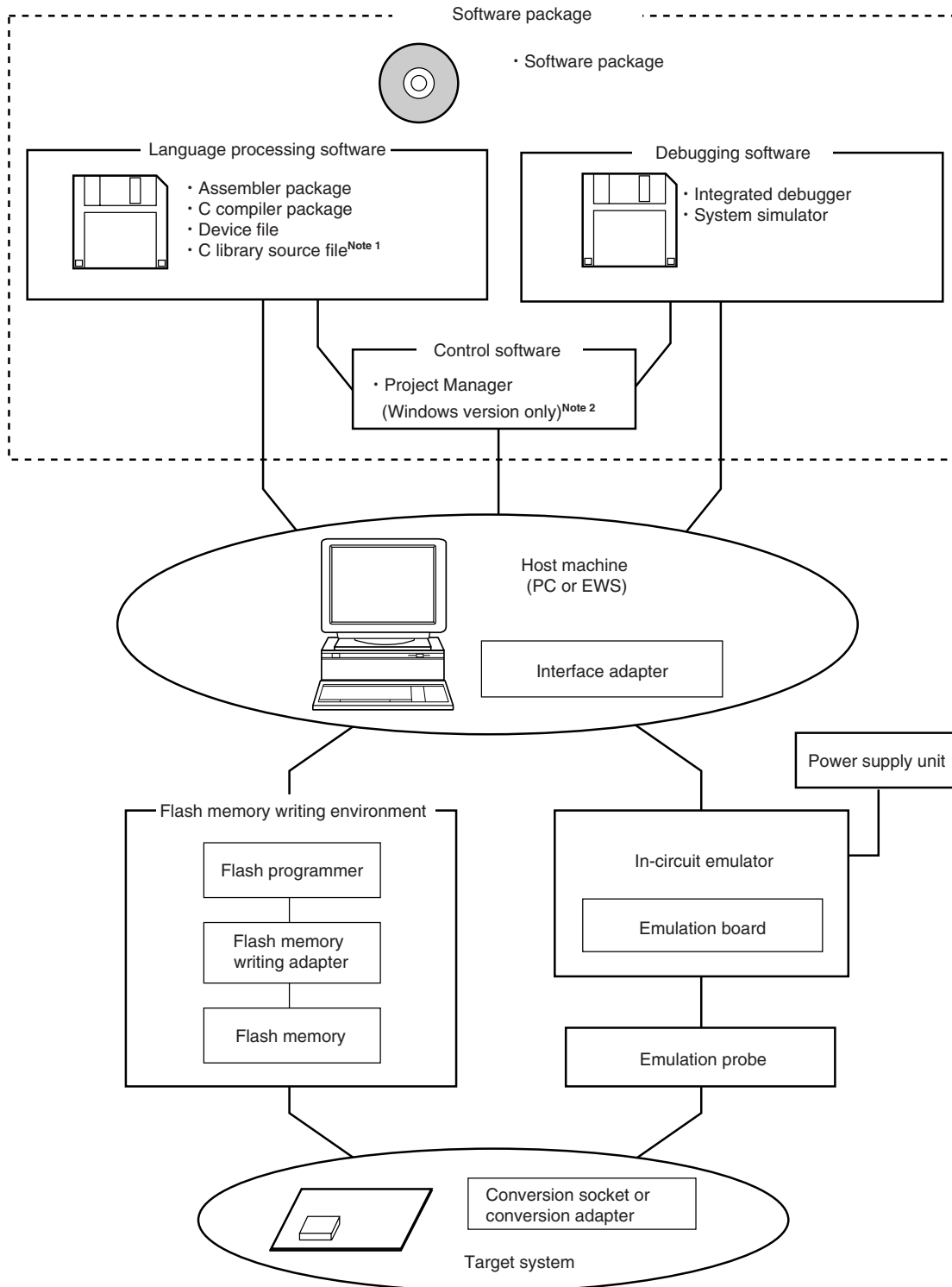
Unless specified otherwise, the products supported by IBM PC/AT™ compatibles can be used in PC98-NX Series. When using the PC98-NX Series, refer to the explanation of IBM PC/AT compatibles.

- Windows

Unless specified otherwise, “Windows” indicates the following operating systems.

- Windows 3.1
- Windows 95, 98, 2000
- Windows NT™ Ver.4.0

Figure A-1. Development Tools



**Notes 1.** C library source file is not included in the software package.

**2.** Project Manager is included in the assembler package.

Project Manager is used only in the Windows environment.



## A.1 Software Package

SP78K0S Software package	Software tools for development of the 78K/0S Series are combined in this package. The following tools are included. RA78K0S, CC78K0S, ID78K0S-NS, SM78K0S, and device files
	Part number: $\mu$ SxxxxSP78K0S

**Remark** xxxx in the part number differs depending on the operating system to be used.

$\mu$ SxxxxSP78K0S

xxxx	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Japanese Windows	CD-ROM
BB17		English Windows	

## A.2 Language Processing Software

RA78K0S Assembler package	Program that converts program written in mnemonic into object codes that can be executed by microcontroller. In addition, automatic functions to generate a symbol table and optimize branch instructions are also provided. Used in combination with a device file (DF789026) (sold separately). <b>&lt;Caution when used in PC environment&gt;</b> The assembler package is a DOS-based application but may be used in the Windows environment by using the Project Manager of Windows (included in the assembler package).
	Part number: $\mu$ SxxxxRA78K0S
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 an assembler package (RA78K0S) and device file (DF789026) (both sold separately). <b>&lt;Caution when used in PC environment&gt;</b> The C compiler package is a DOS-based application but may be used in the Windows environment by using the Project Manager of Windows (included in the assembler package).
	Part number: $\mu$ SxxxxCC78K0S
DF789026 <sup>Note 1</sup> Device file	File containing the information inherent to the device. Used in combination with the RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S (all sold separately).
	Part number: $\mu$ SxxxxDF789026
CC78K0S-L <sup>Note 2</sup> C library source file	Source file of functions for generating object library included in C compiler package. Necessary for changing object library included in C compiler package according to customer's specifications. Since this is a source file, its working environment does not depend on any particular operating system.
	Part number: $\mu$ SxxxxCC78K0S-L

**Notes** 1. DF789026 is a common file that can be used with RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S.  
2. CC78K0S-L is not included in the software package (SP78K0S).

**Remark** xxxx in the part number differs depending on the host machine and operating system to be used.

μSxxxxRA78K0S

μSxxxxCC78K0S

xxxx	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT compatibles	Japanese Windows	3.5-inch 2HD FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	
3P17	HP9000 series 700™	HP-UX™ (Rel. 10.10)	
3K17	SPARCstation™	SunOS™ (Rel. 4.1.4), Solaris™ (Rel. 2.5.1)	

μSxxxxDF789026

μSxxxxCC78K0S-L

xxxx	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT compatibles	Japanese Windows	3.5-inch 2HD FD
BB13		English Windows	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT
3K13	SPARCstation	SunOS (Rel. 4.1.4),	3.5-inch 2HD FD
3K15		Solaris (Rel. 2.5.1)	1/4-inch CGMT

### A.3 Control Software

Project Manager	Control software created for efficient development of the user program in the Windows environment. User program development operations such as editor startup, build, and debugger startup can be performed from the Project Manager. <Caution> The Project Manager is included in the assembler package (RA78K0S). The Project Manager is used only in the Windows environment.
-----------------	---

### A.4 Flash Memory Writing Tools

Flashpro III (FL-PR3, PG-FP3) Flashpro IV (FL-PR4, PG-FP4) Flash programmer	Dedicated flash programmer for microcontrollers incorporating flash memory
FA-44GB-8ES Flash memory writing adapter	Adapter for writing to flash memory and connected to Flashpro III or Flashpro IV. FA-44GB-8ES: For 44-pin plastic LQFP (GB-8ES type)

**Remark** The FL-PR3, FL-PR4, and FA-44GB-8ES are products made by Naito Densai Machida Mfg. Co., Ltd. (TEL +81-45-475-4191).

## A.5 Debugging Tools (Hardware)

IE-78K0S-NS In-circuit emulator	In-circuit emulator for debugging hardware and software of application system using 78K/0S Series. Supports integrated debugger (ID78K0S-NS). Used in combination with AC adapter, emulation probe, and interface adapter for connecting the host machine.
IE-78K0S-NS-A In-circuit emulator	The IE-78K0S-NS-A provides a coverage function in addition to the IE-78K0S-NS functions, thus enhancing the debug functions, including the tracer and timer functions.
IE-70000-MC-PS-B AC adapter	Adapter for supplying power from AC 100 to 240 V outlet.
IE-70000-98-IF-C Interface adapter	Adapter necessary when using PC-9800 series PC (except notebook type) as host machine (C bus supported)
IE-70000-CD-IF-A PC card interface	PC card and interface cable necessary when using notebook PC as host machine (PCMCIA socket supported)
IE-70000-PC-IF-C Interface adapter	Adapter necessary when using IBM PC/AT compatible as host machine (ISA bus supported)
IE-70000-PCI-IF-A Interface adapter	Adapter necessary when using personal computer incorporating PCI bus as host machine
IE-789026-NS-EM1 Emulation board	Board for emulating the peripheral hardware inherent to the device. Used in combination with in-circuit emulator.
NP-44GB-TQ NP-H44GB-TQ Emulation probe	Cable to connect the in-circuit emulator and target system. Used in combination with the TGB-044SAP.
TGB-044SAP Conversion adapter	Conversion adapter to connect the NP-44GB-TQ or NP-H44GB-TQ and a target system board on which a 44-pin plastic LQFP (GB-8ES type) can be mounted

- Remarks**
1. The NP-44GB-TQ and NP-H44GB-TQ are products made by Naito Densai Machida Mfg. Co., Ltd. (TEL +81-45-475-4191).
  2. The TGB-044SAP is a product made by TOKYO ELETECH CORPORATION.  
For further information, contact: Daimaru Kogyo, Ltd.  
Tokyo Electronics Department (TEL +81-3-3820-7112)  
Osaka Electronics Department (TEL +81-6-6244-6672)

## A.6 Debugging Tools (Software)

ID78K0S-NS Integrated debugger	This debugger supports the in-circuit emulators IE-78K0S-NS and IE-78K0S-NS-A for the 78K/0S Series. The ID78K0S-NS is Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. Used in combination with a device file (DF789026) (sold separately).
	Part number: $\mu$ SxxxxID78K0S-NS
SM78K0S System simulator	This is a system simulator for the 78K/0S Series. The SM78K0S is Windows-based software. It can be used to debug the target system at C source level or assembler level while simulating the operation of the target system on the host machine. Using SM78K0S, the logic and performance of the application can be verified independently of hardware development. Therefore, the development efficiency can be enhanced and the software quality can be improved. Used in combination with a device file (DF789026) (sold separately).
	Part number: $\mu$ SxxxxSM78K0S
DF789026 <sup>Note</sup> Device file	File containing the information inherent to the device. Used in combination with the RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S (all sold separately).
	Part number: $\mu$ SxxxxDF789026

**Note** The DF789026 is a common file that can be used with RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S.

**Remark** xxxx in the part number differs depending on the operating system and supply medium to be used.

$\mu$ SxxxxID78K0S-NS

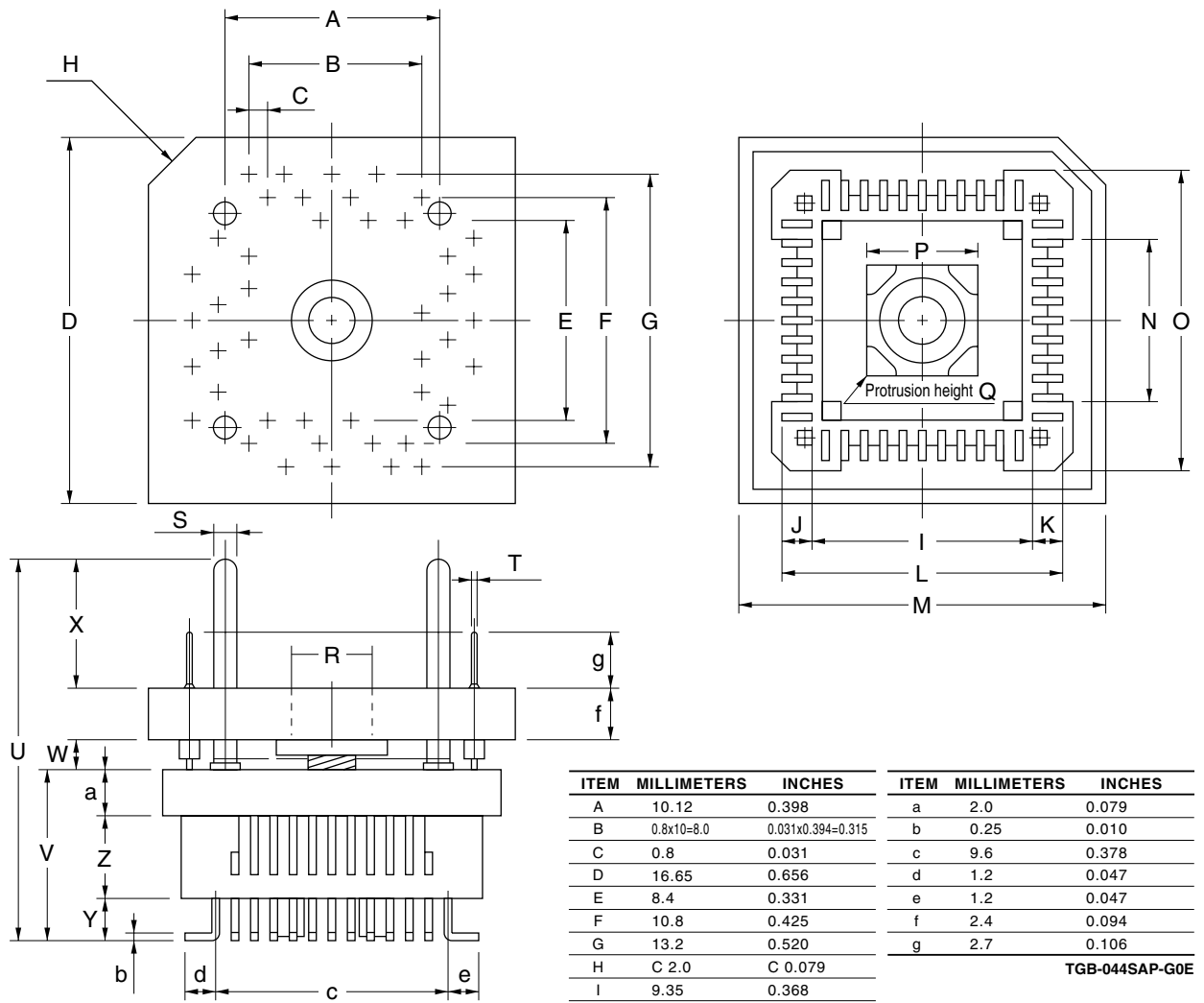
$\mu$ SxxxxSM78K0S

xxxx	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT compatibles	Japanese Windows	3.5-inch 2HD FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	

## A.7 Conversion Adapter (TGB-044SAP) Drawing

Figure A-2. TGB-044SAP Package Drawing (Reference)

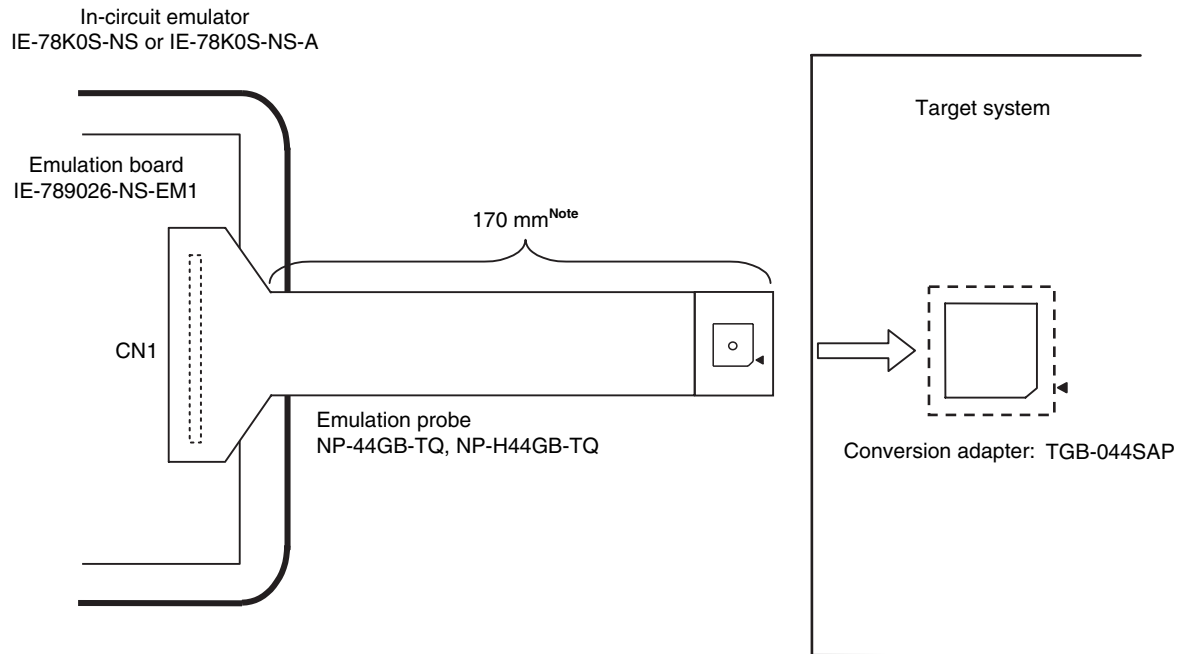
Reference diagram: TGB-044SAP (TQPACK044SA+TQSOCKET044SAP)  
Package dimension (unit: mm)



**note:** Product by TOKYO ELETECH CORPORATION.

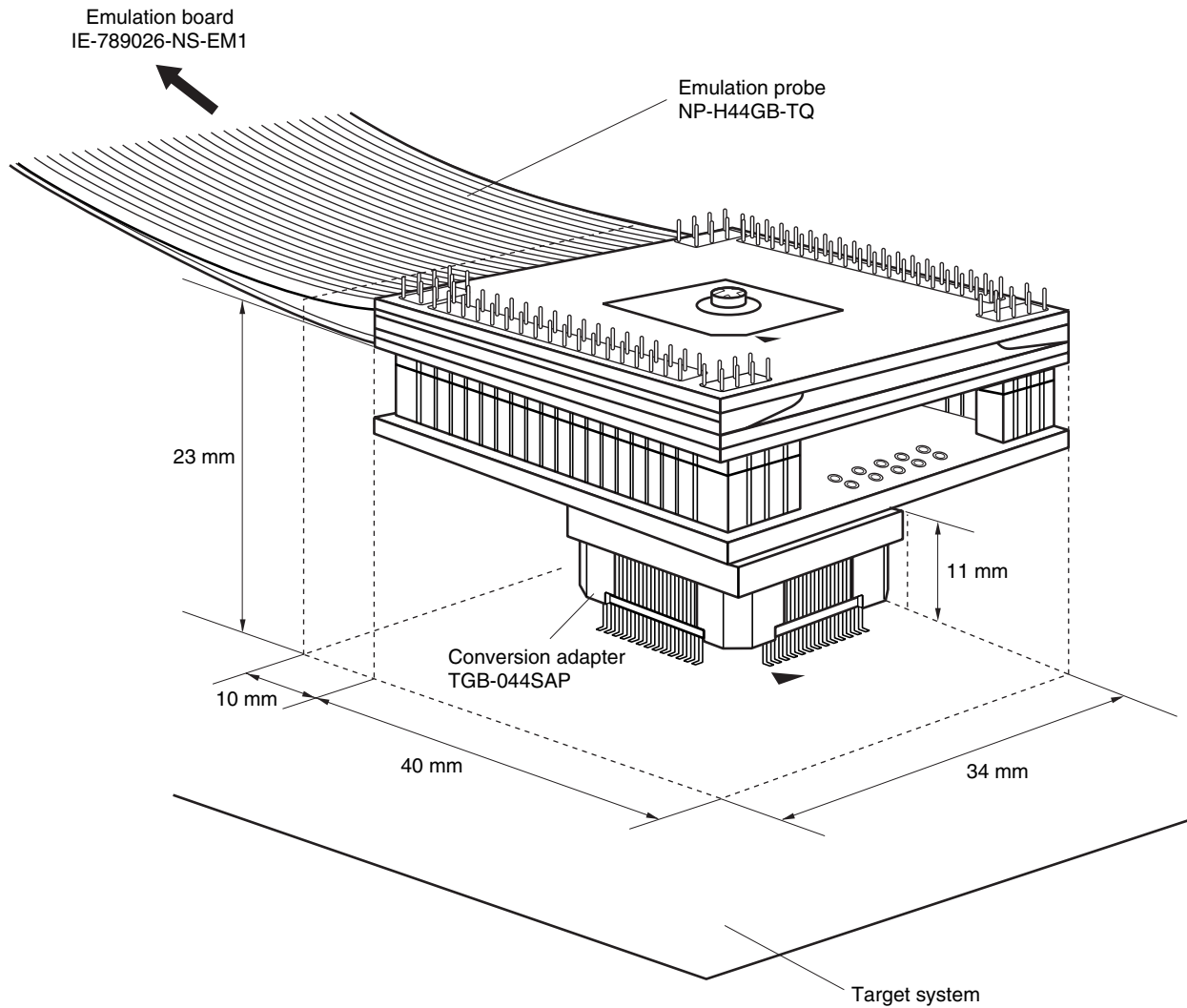
Figures B-1 and B-2 show the conditions when connecting the emulation probe to the conversion adapter. Follow the configuration below and consider the shape of parts to be mounted on the target system when designing a system.

**Figure B-1. Distance Between In-Circuit Emulator and Conversion Adapter**



**Note** Distance when NP-44GB-TQ is used. When NP-H44GB-TQ is used, the distance is 370 mm.

- Remarks**
1. NP-44GB-TQ and NP-H44GB-TQ are products of Naito Densai Machida Mfg. Co., Ltd.
  2. TGB-044SAP is a product of TOKYO ELETECH CORPORATION.

**Figure B-2. Connection Condition of Target System (NP-H44GB-TQ)**

- Remarks**
1. NP-H44GB-TQ is a product of Naito Densetsu Machida Mfg. Co., Ltd.
  2. TGB-044SAP is a product of TOKYO ELETECH CORPORATION.

## APPENDIX C REGISTER INDEX

### C.1 Register Name Index

16-bit capture register 20 (TCP20) .....	88
16-bit compare register 20 (CR20) .....	88
16-bit timer counter 20 (TM20) .....	88
16-bit timer mode control register 20 (TMC20) .....	89
8-bit compare register 00 (CR00) .....	100
8-bit timer counter 00 (TM00) .....	100
8-bit timer mode control register 00 (TMC00) .....	101

#### [A]

Asynchronous serial interface mode register 00 (ASIM00).....	120, 127, 129, 141
Asynchronous serial interface status register 00 (ASIS00).....	122, 130

#### [B]

Baud rate generator control register 00 (BRGC00) .....	123, 131, 142
--	---------------

#### [E]

External interrupt mode register 0 (INTM0) .....	150
--	-----

#### [I]

Interrupt mask flag register 0 (MK0) .....	149
Interrupt mask flag register 1 (MK1) .....	149
Interrupt request flag register 0 (IF0) .....	148
Interrupt request flag register 1 (IF1) .....	148

#### [K]

Key return mode register 00 (KRM00) .....	152
---	-----

#### [O]

Oscillation stabilization time select register (OSTS) .....	161
---	-----

#### [P]

Port 0 (P0).....	66
Port 1 (P1).....	67
Port 2 (P2).....	68
Port 3 (P3).....	71
Port 4 (P4).....	72
Port 5 (P5).....	73
Port mode register 0 (PM0).....	76
Port mode register 1 (PM1).....	76
Port mode register 2 (PM2).....	76



Port mode register 3 (PM3).....	76
Port mode register 4 (PM4).....	76
Port mode register 5 (PM5).....	76, 91, 102
Processor clock control register (PCC).....	80
Pull-up resistor option register (PUO) .....	77

**[R]**

Receive buffer register 00 (RXB00) .....	118
Receive shift register 00 (RXS00).....	118

**[S]**

Serial operation mode register 00 (CSIM00).....	119, 126, 128, 140
---	--------------------

**[T]**

Timer clock select register 2 (TCL2) .....	111
Transmit shift register 00 (TXS00) .....	118

**[W]**

Watchdog timer mode register (WDTM) .....	112
---	-----

**C.2 Register Symbol Index****[A]**

ASIM00:	Asynchronous serial interface mode register 00 .....	120, 127, 129, 141
ASIS00:	Asynchronous serial interface status register 00 .....	122, 130

**[B]**

BRGC00:	Baud rate generator control register 00 .....	123, 131, 142
---------	---	---------------

**[C]**

CR00:	8-bit compare register 00 .....	100
CR20:	16-bit compare register 20 .....	88
CSIM00:	Serial operation mode register 00 .....	119, 126, 128, 140

**[I]**

IF0:	Interrupt request flag register 0 .....	148
IF1:	Interrupt request flag register 1 .....	148
INTM0:	External interrupt mode register 0 .....	150

**[K]**

KRM00:	Key return mode register 00 .....	152
--------	-----------------------------------	-----

**[M]**

MK0:	Interrupt mask flag register 0 .....	149
MK1:	Interrupt mask flag register 1 .....	149

**[O]**

OSTS:	Oscillation stabilization time select register .....	161
-------	--	-----

**[P]**

P0:	Port 0 .....	66
P1:	Port 1 .....	67
P2:	Port 2 .....	68
P3:	Port 3 .....	71
P4:	Port 4 .....	72
P5:	Port 5 .....	73
PCC:	Processor clock control register .....	80
PM0:	Port mode register 0 .....	76
PM1:	Port mode register 1 .....	76
PM2:	Port mode register 2 .....	76
PM3:	Port mode register 3 .....	76
PM4:	Port mode register 4 .....	76
PM5:	Port mode register 5 .....	76, 91, 102
PUO:	Pull-up resistor option register .....	77

**[R]**

RXB00:	Receive buffer register 00 .....	118
RXS00:	Receive shift register 00 .....	118

**[T]**

TCL2:	Timer clock select register 2 .....	111
TCP20:	16-bit capture register 20 .....	88
TM00:	8-bit timer counter 00 .....	100
TM20:	16-bit timer counter 20 .....	88
TMC00:	8-bit timer mode control register 00 .....	101
TMC20:	16-bit timer mode control register 20 .....	89
TXS00:	Transmit shift register 00.....	118

**[W]**

WDTM:	Watchdog timer mode register .....	112
-------	------------------------------------	-----

## APPENDIX D REVISION HISTORY

Here is the revision history of this manual. The “Applied to:” column indicates the chapters of each edition in which the revision was applied.

(1/2)

Edition	Revision from Previous Edition	Applied to:
Second edition	Change of $\mu$ PD789025 and $\mu$ PD789026 from “under development” to “developed”	Throughout
	Change of symbols in special function register list	CHAPTER 3 CPU ARCHITECTURE
	Change of asynchronous serial interface status register 00 so that it can be manipulated in 1-bit units	
	Change of block diagram of each port	CHAPTER 4 PORT FUNCTIONS
	Change of symbols and flag names of 16-bit timer mode control register 20	CHAPTER 6 16-BIT TIMER COUNTER
	Change of symbols and flag names of 8-bit timer mode control register 00	CHAPTER 7 8-BIT TIMER/ EVENT COUNTER
	Change of symbols and flag names of serial operation mode register 00	CHAPTER 9 SERIAL INTERFACE 00
	Change of symbols and flag names of asynchronous serial interface mode register 00	
	Change of symbols and flag names of asynchronous serial interface status register 00	
	Change of asynchronous serial interface status register 00 so that 1-bit memory manipulation instruction can be used	
	Change of symbols and flag names of baud rate generator control register 00	
	Change of flag names of interrupt request flag register	CHAPTER 10 INTERRUPT FUNCTIONS
	Change of flag names of interrupt mask flag register	
	Change of symbols and flag names of key return mode register 00	
	Addition of description on timing of maskable interrupt request acknowledgement	
	Addition of setting with Flashpro II	CHAPTER 13 $\mu$ PD78F9026
Third edition	Completion of development of $\mu$ PD789022 and $\mu$ PD789024	Throughout
	Change of part number from $\mu$ PD78F9026 to $\mu$ PD78F9026A	
	Deletion of following products: $\mu$ PD789022CU-xxx, $\mu$ PD789024CU-xxx	
	Addition of GB-8ES type package to all models	
	Change of circuit type and recommended connection of unused pins in processing of I/O circuit type of each pin and unused pins	CHAPTER 2 PIN FUNCTIONS
	Addition of cautions on rewriting CR20 to operation as timer interrupt	CHAPTER 6 16-BIT TIMER
	Addition of cautions on rewriting CR00 to 8-bit compare register 00 (CR00)	CHAPTER 7 8-BIT TIMER/ EVENT COUNTER
	Addition of description of operation to operation as interval timer	
	Addition of description of operation to operation as external event counter	
	Addition of description of operation to operation as square wave output	
	Change of flash programmer from Flashpro II to Flashpro III	CHAPTER 13 $\mu$ PD78F9026A
	Addition of part number of MX78K0S to embedded software	APPENDIX B EMBEDDED SOFTWARE

(2/2)

Edition	Revision from Previous Edition	Applied to:
Fourth edition	Deletion of the following packages • 42-pin plastic shrink DIP (CU type) • 44-pin plastic QFP (GB-3BS-MTX type)	Throughout
	Modification of pin handling of V <sub>PP</sub> pin	CHAPTER 2 PIN FUNCTIONS
	Modification of description in 6.4.1 Operation as timer interrupt and 6.4.2 Operation as timer output	CHAPTER 6 16-BIT TIMER 20
	Modification of Caution on rewriting CR20 in 6.4.1 Operation as timer interrupt	
	Addition of 6.5 Notes on Using 16-Bit Timer 20	
	Modification of description of PE00 flag in Figure 9-5 Format of Asynchronous Serial Interface Status Register 00	CHAPTER 9 SERIAL INTERFACE 00
	Addition of description on reading receive data of UART	
	Addition of Caution in Figure 10-2 Format of Interrupt Request Flag Register	CHAPTER 10 INTERRUPT FUNCTIONS
	Overall revision of contents related to flash memory programming as 13.1 Flash Memory Characteristics	CHAPTER 13 $\mu$ PD78F9026A
	Addition of electrical specifications	CHAPTER 15 ELECTRICAL SPECIFICATIONS
	Addition of characteristics curves for mask ROM version	CHAPTER 16 CHARACTERISTICS CURVES (MASK ROM VERSION)
	Addition of package drawing	CHAPTER 17 PACKAGE DRAWING
	Addition of recommended soldering conditions	CHAPTER 18 RECOMMENDED SOLDERING CONDITIONS
	Overall revision of contents of development tools Deletion of embedded software	APPENDIX A DEVELOPMENT TOOLS
	Addition of notes on target system design	APPENDIX B NOTES ON TARGET SYSTEM DESIGN