

## **$\mu$ PD789104A, 789114A, 789124A, 789134A Subseries**

### **8-bit Single-chip Microcontrollers**

---

<b><math>\mu</math>PD789101A</b>	<b><math>\mu</math>PD789121A</b>
<b><math>\mu</math>PD789102A</b>	<b><math>\mu</math>PD789122A</b>
<b><math>\mu</math>PD789104A</b>	<b><math>\mu</math>PD789124A</b>
<b><math>\mu</math>PD789111A</b>	<b><math>\mu</math>PD789131A</b>
<b><math>\mu</math>PD789112A</b>	<b><math>\mu</math>PD789132A</b>
<b><math>\mu</math>PD789114A</b>	<b><math>\mu</math>PD789134A</b>
<b><math>\mu</math>PD78F9116A</b>	<b><math>\mu</math>PD78F9136A</b>
<b><math>\mu</math>PD789101A(A)</b>	<b><math>\mu</math>PD789121A(A)</b>
<b><math>\mu</math>PD789102A(A)</b>	<b><math>\mu</math>PD789122A(A)</b>
<b><math>\mu</math>PD789104A(A)</b>	<b><math>\mu</math>PD789124A(A)</b>
<b><math>\mu</math>PD789111A(A)</b>	<b><math>\mu</math>PD789131A(A)</b>
<b><math>\mu</math>PD789112A(A)</b>	<b><math>\mu</math>PD789132A(A)</b>
<b><math>\mu</math>PD789114A(A)</b>	<b><math>\mu</math>PD789134A(A)</b>

**[MEMO]**

**① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS**

Note:

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

**② HANDLING OF UNUSED INPUT PINS FOR CMOS**

Note:

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

**③ STATUS BEFORE INITIALIZATION OF MOS DEVICES**

Note:

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

**EEPROM is a trademark of NEC 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.**

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

**NEWS and NEWS-OS are trademarks of Sony Corporation.**

**TRON is an abbreviation of The Realtime Operating System Nucleus.**

**ITRON is an abbreviation of Industrial TRON.**

The export of these products from Japan is regulated by the Japanese government. The export of some or all of these products may be prohibited without governmental license. To export or re-export some or all of these products from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

Licence not needed:  $\mu$ PD78F9116A, 78F9136A

The customer must judge the need for license:

$\mu$ PD789101A, 789102A, 789104A, 789101A(A), 789102A(A), 789104A(A)

$\mu$ PD789111A, 789112A, 789114A, 789111A(A), 789112A(A), 789114A(A)

$\mu$ PD789121A, 789122A, 789124A, 789121A(A), 789122A(A), 789124A(A)

$\mu$ PD789131A, 789132A, 789134A, 789131A(A), 789132A(A), 789134A(A)

- **The information in this document is current as of June, 2000. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC's data sheets or data books, etc., for the most up-to-date specifications of NEC semiconductor products. Not all products and/or types are available in every country. Please check with an NEC sales representative for availability and additional information.**

- No part of this document may be copied or reproduced in any form or by any means without prior written consent of NEC. NEC assumes no responsibility for any errors that may appear in this document.
- NEC does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC semiconductor products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of customer's equipment shall be done under the full responsibility of customer. NEC assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC endeavours to enhance the quality, reliability and safety of NEC semiconductor products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC semiconductor products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment, and anti-failure features.
- NEC semiconductor products are classified into the following three quality grades:  
"Standard", "Special" and "Specific". The "Specific" quality grade applies only to semiconductor products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of a semiconductor product depend on its quality grade, as indicated below. Customers must check the quality grade of each semiconductor product before using it in a particular application.

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

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

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

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

(Note)

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

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

M8E 00.4

# Regional Information

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

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

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

**NEC Electronics Inc. (U.S.)**

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

**NEC Electronics (Germany) GmbH**

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

**NEC Electronics (UK) Ltd.**

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

**NEC Electronics Italiana s.r.l.**

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

**NEC Electronics (Germany) GmbH**

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

**NEC Electronics (France) S.A.**

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

**NEC Electronics (France) S.A.**

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

**NEC Electronics (Germany) GmbH**

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

**NEC Electronics Hong Kong Ltd.**

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

**NEC Electronics Hong Kong Ltd.**

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

**NEC Electronics Singapore Pte. Ltd.**

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

**NEC Electronics Taiwan Ltd.**

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

**NEC do Brasil S.A.**

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

J99.1

**[MEMO]**

## INTRODUCTION

### Target Readers

This manual is intended for users who wish to understand the functions of the  $\mu$ PD789104A, 789114A, 789124A, 789134A Subseries and to design and develop application systems and programs using these microcontrollers.

The target devices are shown as follows:

- $\mu$ PD789104A Subseries:  $\mu$ PD789101A, 789102A, 789104A,  $\mu$ PD789101A(A), 789102A(A), 789104A(A)
- $\mu$ PD789114A Subseries:  $\mu$ PD789111A, 789112A, 789114A, 78F9116A,  $\mu$ PD789111A(A), 789112A(A), 789114A(A)
- $\mu$ PD789124A Subseries:  $\mu$ PD789121A, 789122A, 789124A,  $\mu$ PD789121A(A), 789122A(A), 789124A(A)
- $\mu$ PD789134A Subseries:  $\mu$ PD789131A, 789132A, 789134A, 78F9136A,  $\mu$ PD789131A(A), 789132A(A), 789134A(A)

The  $\mu$ PD789104A/114A/124A/134A Subseries is a generic term for all the target devices in this manual.

The oscillation frequency of the system clock is regarded as  $f_x$  for ceramic/crystal oscillation ( $\mu$ PD789104A and 789114A Subseries), and regarded as  $f_{cc}$  for an RC oscillation ( $\mu$ PD789124A and 789134A Subseries).

### Purpose

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

### Organization

The  $\mu$ PD789104A, 789114A, 789124A, 789134A Subseries User's Manual is divided into two parts: this manual and instructions (common to the 78K/0S Series).

$\mu$ PD789104A, 789114A, 789124A, 789134A Subseries User's Manual (This manual)	78K/0S Series User's Manual Instructions
<ul style="list-style-type: none"><li>• Pin functions</li><li>• Internal block functions</li><li>• Interrupt</li><li>• Other internal peripheral functions</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 in the fields of electrical engineering, logic circuits, and microcontrollers.

- For users who use this document as the manual for the  $\mu$ PD789101A(A), 789102A(A), 789104A(A), 789111A(A), 789112A(A), 789114A(A), 789121A(A), 789122A(A), 789124A(A), 789131A(A), 789132A(A), and 789134A(A)

→ The only differences between standard products and (A) products are the quality grades and the electrical specifications. (refer to **1.9 Differences between Standard Quality Grade Products and (A) Products** and **2.9 Differences between Standard Quality Grade Products and (A) Products**). For the (a) products, read the part numbers in the following manner.

$\mu$ PD789101A	→ $\mu$ PD789101A(A)	$\mu$ PD789121A	→ $\mu$ PD789121A(A)
$\mu$ PD789102A	→ $\mu$ PD789102A(A)	$\mu$ PD789122A	→ $\mu$ PD789122A(A)
$\mu$ PD789104A	→ $\mu$ PD789104A(A)	$\mu$ PD789124A	→ $\mu$ PD789124A(A)
$\mu$ PD789111A	→ $\mu$ PD789111A(A)	$\mu$ PD789131A	→ $\mu$ PD789131A(A)
$\mu$ PD789112A	→ $\mu$ PD789112A(A)	$\mu$ PD789132A	→ $\mu$ PD789132A(A)
$\mu$ PD789114A	→ $\mu$ PD789114A(A)	$\mu$ PD789134A	→ $\mu$ PD789134A(A)

- To understand the overall functions in general
  - Read this manual in the order of the **CONTENTS**.
- How to interpret register formats
  - The name  $iV$  a bit whose number is encircled is reserved for the assembler and is defined for the C compiler by the header file **sfrbit.h**.
- To learn the detailed functions of a register whose register name is known
  - Refer to **APPENDIX C REGISTER INDEX**.
- To learn the details of the instruction functions of the 78K/0S Series
  - Refer to **78K/0S Series User's Manual Instructions (U11047E)**.
- To know the electrical specifications of the  $\mu$ PD789104A/114A/124A/134A Subseries
  - Refer to separately available Data Sheet.

**Caution** The application examples in this manual are created for "Standard" quality grade products for general electric equipment. When using the application examples in this manual for purposes which require "Special" quality grades, thoroughly examine the quality grade of each part and circuit actually used.

## Conventions

Data significance:	Higher digits on the left and lower digits on the right
Active low representation:	$\overline{xxx}$ (overscore over pin or signal name)
<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 ... xxx
	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.	
	English	Japanese
μPD789101A,102A,104A,111A,112A,114A,101A(A),102A(A),104A(A),111A(A), 112A(A),114A(A) Data Sheet	U14590E	U14590J
μPD789121A,122A,124A,131A,132A,134A,121A(A),122A(A),124A(A),131A(A), 132A(A),134A(A) Data Sheet	U14678E	U14678J
μPD78F9116A Data Sheet	U14667E	U14667J
μPD78F9136A Data Sheet	U14690E	U14690J
μPD789104A,789114A,789124A,789134A Subseries User's Manual	This manual	U14643J
78K/0S Series User's Manual Instruction	U11047E	U11047J
78K/0, 78K/0S Series Application Note Flash Memory Write	U14458E	U14458J

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

Document Name		Document No.	
		English	Japanese
RA78K0S Assembler Package	Operation	U11622E	U11622J
	Assembly Language	U11599E	U11599J
	Structured Assembly Language	U11623E	U11623J
CC78K0S C Compiler	Operation	U11816E	U11816J
	Language	U11817E	U11817J
SM78K0S System Simulator Windows™ Based	Reference	U11489E	U11489J
SM78K Series System Simulator	External components user open interface specification	U10092E	U10092J
ID78K0S-NS Integrated Debugger Windows Based	Reference	U12901E	U12901J
IE-78K0S-NS In-Circuit Emulator		U13549E	U13549J
IE-789136-NS-EM1 Emulation Board		U14363E	U14363J

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

**Document Related to Embedded Software (User's Manual)**

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

**Other Documents**

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

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

## CONTENTS

<b>CHAPTER 1 GENERAL (<math>\mu</math>PD789104A, 789114A SUBSERIES) .....</b>	<b>23</b>
1.1 Features .....	23
1.2 Applications .....	23
1.3 Ordering Information .....	24
1.4 Quality Grade .....	24
1.5 Pin Configuration (Top View) .....	25
1.6 78K/0S Series Lineup .....	26
1.7 Block Diagram .....	28
1.8 Outline of Functions .....	29
1.9 Differences between Standard Quality Grade Products and (A) Products .....	30
 <b>CHAPTER 2 GENERAL (<math>\mu</math>PD789124A, 789134A SUBSERIES) .....</b>	 <b>31</b>
2.1 Features .....	31
2.2 Applications .....	31
2.3 Ordering Information .....	32
2.4 Quality Grade .....	32
2.5 Pin Configuration (Top View) .....	33
2.6 78K/0S Series Lineup .....	34
2.7 Block Diagram .....	36
2.8 Outline of Functions .....	37
2.9 Differences between Standard Quality Grade Products and (A) Products .....	38
 <b>CHAPTER 3 PIN FUNCTIONS .....</b>	 <b>39</b>
3.1 Pin Function List .....	39
3.2 Description of Pin Functions .....	41
3.2.1 P00 to P03 (Port 0) .....	41
3.2.2 P10, P11 (Port 1) .....	41
3.2.3 P20 to P25 (Port 2) .....	41
3.2.4 P50 to P53 (Port 5) .....	42
3.2.5 P60 to P63 (Port 6) .....	42
3.2.6 $\overline{\text{RESET}}$ .....	42
3.2.7 X1, X2 ( $\mu$ PD789104A, 789114A Subseries) .....	42
3.2.8 CL1, CL2 ( $\mu$ PD789124A, 789134A Subseries) .....	42
3.2.9 $\text{AV}_{\text{DD}}$ .....	42
3.2.10 $\text{AV}_{\text{SS}}$ .....	42
3.2.11 $\text{V}_{\text{DD}}$ .....	42
3.2.12 $\text{V}_{\text{SS}}$ .....	42
3.2.13 $\text{V}_{\text{PP}}$ ( $\mu$ PD78F9116A, 78F9136A only) .....	43
3.2.14 IC0 (pin No.20) (mask ROM version only) .....	43
3.2.15 IC0 (pins No.10 and No.21) .....	43
3.3 Pin Input/Output Circuits and Recommended Connection of Unused Pins .....	44
 <b>CHAPTER 4 CPU ARCHITECTURE .....</b>	 <b>47</b>
4.1 Memory Space .....	47
4.1.1 Internal program memory space .....	51

4.1.2 Internal data memory (internal high-speed RAM) space .....	52
4.1.3 Special function register (SFR) area .....	52
4.1.4 Data memory addressing .....	52
<b>4.2 Processor Registers .....</b>	<b>56</b>
4.2.1 Control registers .....	56
4.2.2 General registers .....	59
4.2.3 Special function registers (SFRs) .....	60
<b>4.3 Instruction Address Addressing .....</b>	<b>63</b>
4.3.1 Relative addressing .....	63
4.3.2 Immediate addressing .....	64
4.3.3 Table indirect addressing .....	65
4.3.4 Register addressing .....	65
<b>4.4 Operand Address Addressing .....</b>	<b>66</b>
4.4.1 Direct addressing .....	66
4.4.2 Short direct addressing .....	67
4.4.3 Special function register (SFR) addressing .....	68
4.4.4 Register addressing .....	69
4.4.5 Register indirect addressing .....	70
4.4.6 Based addressing .....	71
4.4.7 Stack addressing .....	71
<b>CHAPTER 5 PORT FUNCTIONS .....</b>	<b>73</b>
<b>5.1 Functions of Ports .....</b>	<b>73</b>
<b>5.2 Port Configuration .....</b>	<b>75</b>
5.2.1 Port 0 .....	75
5.2.2 Port 1 .....	76
5.2.3 Port 2 .....	77
5.2.4 Port 5 .....	81
5.2.5 Port 6 .....	82
<b>5.3 Port Function Control Registers .....</b>	<b>83</b>
<b>5.4 Operation of Port Functions .....</b>	<b>85</b>
5.4.1 Writing to I/O port .....	85
5.4.2 Reading from I/O port .....	85
5.4.3 Arithmetic operation of I/O port .....	86
<b>CHAPTER 6 CLOCK GENERATOR (<math>\mu</math>PD789104A, 789114A SUBSERIES) .....</b>	<b>87</b>
<b>6.1 Function of Clock Generator .....</b>	<b>87</b>
<b>6.2 Configuration of Clock Generator .....</b>	<b>87</b>
<b>6.3 Register Controlling Clock Generator .....</b>	<b>88</b>
<b>6.4 System Clock Oscillator .....</b>	<b>89</b>
6.4.1 System clock oscillator .....	89
6.4.2 Divider .....	92
<b>6.5 Operation of Clock Generator .....</b>	<b>92</b>
<b>6.6 Changing Setting of CPU Clock .....</b>	<b>93</b>
6.6.1 Time required for switching CPU clock .....	93
6.6.2 Switching CPU clock .....	93

<b>CHAPTER 7 CLOCK GENERATOR (<math>\mu</math>PD789124A, 789134A SUBSERIES)</b>	<b>95</b>
<b>7.1 Function of Clock Generator</b>	<b>95</b>
<b>7.2 Configuration of Clock Generator</b>	<b>95</b>
<b>7.3 Register Controlling Clock Generator</b>	<b>96</b>
<b>7.4 System Clock Oscillator</b>	<b>97</b>
7.4.1 System clock oscillator	97
7.4.2 Divider	100
<b>7.5 Operation of Clock Generator</b>	<b>100</b>
<b>7.6 Changing Setting of CPU Clock</b>	<b>101</b>
7.6.1 Time required for switching CPU clock	101
7.6.2 Switching CPU clock	101
<b>CHAPTER 8 16-BIT TIMER 20</b>	<b>103</b>
<b>8.1 16-Bit Timer 20 Functions</b>	<b>103</b>
<b>8.2 16-Bit Timer 20 Configuration</b>	<b>104</b>
<b>8.3 Registers Controlling 16-Bit Timer 20</b>	<b>106</b>
<b>8.4 16-Bit Timer 20 Operation</b>	<b>109</b>
8.4.1 Operation as timer interrupt	109
8.4.2 Operation as timer output	111
8.4.3 Capture operation	112
8.4.4 16-bit timer counter 20 readout	113
<b>CHAPTER 9 8-BIT TIMER/EVENT COUNTER 80</b>	<b>115</b>
<b>9.1 Functions of 8-Bit Timer/Event Counter 80</b>	<b>115</b>
<b>9.2 8-Bit Timer/Event Counter 80 Configuration</b>	<b>116</b>
<b>9.3 8-Bit Timer/Event Counter 80 Control Registers</b>	<b>118</b>
<b>9.4 Operation of 8-Bit Timer/Event Counter 80</b>	<b>120</b>
9.4.1 Operation as interval timer	120
9.4.2 Operation as external event counter	122
9.4.3 Operation as square wave output	123
9.4.4 Operation as PWM output	125
<b>9.5 Notes on Using 8-Bit Timer/Event Counter 80</b>	<b>127</b>
<b>CHAPTER 10 WATCHDOG TIMER</b>	<b>129</b>
<b>10.1 Functions of Watchdog Timer</b>	<b>129</b>
<b>10.2 Configuration of Watchdog Timer</b>	<b>130</b>
<b>10.3 Watchdog Timer Control Register</b>	<b>131</b>
<b>10.4 Operation of Watchdog Timer</b>	<b>133</b>
10.4.1 Operation as watchdog timer	133
10.4.2 Operation as interval timer	134
<b>CHAPTER 11 8-BIT A/D CONVERTER (<math>\mu</math>PD789104A, 789124A SUBSERIES)</b>	<b>135</b>
<b>11.1 8-Bit A/D Converter Functions</b>	<b>135</b>
<b>11.2 8-Bit A/D Converter Configuration</b>	<b>135</b>
<b>11.3 Registers Controlling 8-Bit A/D Converter</b>	<b>138</b>
<b>11.4 8-Bit A/D Converter Operation</b>	<b>140</b>
11.4.1 Basic operation of 8-bit A/D converter	140

11.4.2	Input voltage and conversion result .....	141
11.4.3	Operation mode of 8-bit A/D converter .....	143
<b>11.5</b>	<b>Cautions Related to 8-Bit A/D Converter .....</b>	<b>144</b>
<b>CHAPTER 12</b>	<b>10-BIT A/D CONVERTER (<math>\mu</math>PD789114A, 789134A SUBSERIES) .....</b>	<b>149</b>
12.1	10-Bit A/D Converter Functions .....	149
12.2	10-Bit A/D Converter Configuration .....	149
12.3	Registers Controlling 10-Bit A/D Converter .....	152
12.4	10-Bit A/D Converter Operation .....	154
12.4.1	Basic operation of 10-bit A/D converter .....	154
12.4.2	Input voltage and conversion result .....	155
12.4.3	Operation mode of 10-bit A/D converter .....	157
12.5	Cautions Related to 10-Bit A/D Converter .....	158
<b>CHAPTER 13</b>	<b>SERIAL INTERFACE 20 .....</b>	<b>163</b>
13.1	Functions of Serial Interface 20 .....	163
13.2	Serial Interface 20 Configuration .....	163
13.3	Serial Interface 20 Control Registers .....	167
13.4	Serial Interface 20 Operation .....	174
13.4.1	Operation stop mode .....	174
13.4.2	Asynchronous serial interface (UART) mode .....	176
13.4.3	3-wire serial I/O mode .....	189
<b>CHAPTER 14</b>	<b>MULTIPLIER .....</b>	<b>199</b>
14.1	Multiplier Function .....	199
14.2	Multiplier Configuration .....	199
14.3	Multiplier Control Register .....	201
14.4	Multiplier Operation .....	202
<b>CHAPTER 15</b>	<b>INTERRUPT FUNCTIONS .....</b>	<b>203</b>
15.1	Interrupt Function Types .....	203
15.2	Interrupt Sources and Configuration .....	203
15.3	Interrupt Function Control Registers .....	206
15.4	Interrupt Processing Operation .....	211
15.4.1	Non-maskable interrupt request acknowledgement operation .....	211
15.4.2	Maskable interrupt request acknowledgement operation .....	213
15.4.3	Multiple interrupt processing .....	215
15.4.4	Interrupt request reserve .....	217
<b>CHAPTER 16</b>	<b>STANDBY FUNCTION .....</b>	<b>219</b>
16.1	Standby Function and Configuration .....	219
16.1.1	Standby function .....	219
16.1.2	Standby function control register ( $\mu$ PD789104A, 789114A Subseries) .....	220
16.2	Operation of Standby Function .....	221
16.2.1	HALT mode .....	221
16.2.2	STOP mode .....	224
<b>CHAPTER 17</b>	<b>RESET FUNCTION .....</b>	<b>227</b>

<b>CHAPTER 18</b>	<b><math>\mu</math>PD78F9116A, 78F9136A .....</b>	<b>231</b>
<b>18.1</b>	<b>Flash Memory Programming .....</b>	<b>232</b>
18.1.1	Selecting communication mode .....	232
18.1.2	Function of flash memory programming .....	233
18.1.3	Flashpro III connection example .....	233
18.1.4	Example of settings for Flashpro III (PG-FP3) .....	237
<b>CHAPTER 19</b>	<b>MASK OPTION (MASK ROM VERSION) .....</b>	<b>239</b>
<b>CHAPTER 20</b>	<b>INSTRUCTION SET .....</b>	<b>241</b>
<b>20.1</b>	<b>Operation .....</b>	<b>241</b>
20.1.1	Operand identifiers and description methods .....	241
20.1.2	Description of “operation” column .....	242
20.1.3	Description of “flag operation” column .....	242
<b>20.2</b>	<b>Operation List .....</b>	<b>243</b>
<b>20.3</b>	<b>Instructions Listed by Addressing Type .....</b>	<b>248</b>
<b>APPENDIX A</b>	<b>DEVELOPMENT TOOLS .....</b>	<b>251</b>
<b>A.1</b>	<b>Language Processing Software .....</b>	<b>253</b>
<b>A.2</b>	<b>Flash Memory Writing Tools .....</b>	<b>254</b>
<b>A.3</b>	<b>Debugging Tools .....</b>	<b>254</b>
A.3.1	Hardware .....	254
A.3.2	Software .....	255
<b>APPENDIX B</b>	<b>EMBEDDED SOFTWARE .....</b>	<b>257</b>
<b>APPENDIX C</b>	<b>REGISTER INDEX .....</b>	<b>259</b>
<b>C.1</b>	<b>Register Name Index (Alphabetic Order) .....</b>	<b>259</b>
<b>C.2</b>	<b>Register Symbol Index (Alphabetic Order) .....</b>	<b>261</b>

**[MEMO]**



## LIST OF FIGURES (1/4)

Figure No.	Title	Page
3-1.	Pin Input/Output Circuits .....	45
4-1.	Memory Map ( $\mu$ PD789101A, 789111A, 789121A, 789131A) .....	47
4-2.	Memory Map ( $\mu$ PD789102A, 789112A, 789122A, 789132A) .....	48
4-3.	Memory Map ( $\mu$ PD789104A, 789114A, 789124A, 789134A) .....	49
4-4.	Memory Map ( $\mu$ PD78F9116A, 78F9136A) .....	50
4-5.	Data Memory Addressing ( $\mu$ PD789101A, 789111A, 789121A, 789131A) .....	52
4-6.	Data Memory Addressing ( $\mu$ PD789102A, 789112A, 789122A, 789132A) .....	53
4-7.	Data Memory Addressing ( $\mu$ PD789104A, 789114A, 789124A, 789134A) .....	54
4-8.	Data Memory Addressing ( $\mu$ PD78F9116A, 78F9136A) .....	55
4-9.	Program Counter Configuration .....	56
4-10.	Program Status Word Configuration .....	56
4-11.	Stack Pointer Configuration .....	58
4-12.	Data to be Saved to Stack Memory .....	58
4-13.	Data to be Restored from Stack Memory .....	58
4-14.	General Register Configuration .....	59
5-1.	Port Types .....	73
5-2.	Block Diagram of P00 to P03 .....	75
5-3.	Block Diagram of P10 and P11 .....	76
5-4.	Block Diagram of P20 .....	77
5-5.	Block Diagram of P21 .....	78
5-6.	Block Diagram of P22, P23, and P25 .....	79
5-7.	Block Diagram of P24 .....	80
5-8.	Block Diagram of P50 to P53 .....	81
5-9.	Block Diagram of P60 to P63 .....	82
5-10.	Port Mode Register Format .....	84
5-11.	Pull-Up Resistor Option Register 0 Format .....	84
5-12.	Pull-Up Resistor Option Register B2 Format .....	84
6-1.	Block Diagram of Clock Generator .....	87
6-2.	Processor Clock Control Register Format .....	88
6-3.	External Circuit of System Clock Oscillator .....	89
6-4.	Examples of Incorrect Resonator Connection .....	90
6-5.	Switching CPU Clock .....	93
7-1.	Block Diagram of Clock Generator .....	95
7-2.	Processor Clock Control Register Format .....	96
7-3.	External Circuit of System Clock Oscillator .....	97
7-4.	Examples of Incorrect Resonator Connection .....	98
7-5.	Switching CPU Clock .....	101

## LIST OF FIGURES (2/4)

Figure No.	Title	Page
8-1.	Block Diagram of 16-Bit Timer 20 .....	104
8-2.	16-Bit Timer Mode Control Register 20 Format .....	107
8-3.	Port Mode Register 2 Format .....	108
8-4.	Settings of 16-Bit Timer Mode Control Register 20 at Timer Interrupt Operation .....	109
8-5.	Timing of Timer Interrupt Operation .....	110
8-6.	Settings of 16-Bit Timer Mode Control Register 20 at Timer Output Operation .....	111
8-7.	Timer Output Timing .....	111
8-8.	Settings of 16-Bit Timer Mode Control Register 20 at Capture Operation .....	112
8-9.	Capture Operation Timing (Both Edges of CPT20 Pin Are Specified) .....	112
8-10.	16-Bit Timer Counter 20 Readout Timing .....	113
9-1.	Block Diagram of 8-Bit Timer/Event Counter 80 .....	117
9-2.	8-Bit Timer Mode Control Register 80 Format .....	118
9-3.	Port Mode Register 2 Format .....	119
9-4.	Interval Timer Operation Timing .....	121
9-5.	External Event Counter Operation Timing (with Rising Edge Specified) .....	122
9-6.	Square Wave Output Timing .....	124
9-7.	PWM Output Timing .....	126
9-8.	Start Timing of 8-Bit Timer Counter .....	127
9-9.	External Event Counter Operation Timing .....	127
10-1	Block Diagram of Watchdog Timer .....	130
10-2	Timer Clock Select Register 2 Format .....	131
10-3	Format of Watchdog Timer Mode Register .....	132
11-1.	Block Diagram of 8-Bit A/D Converter .....	136
11-2.	Format of A/D Converter Mode Register 0 .....	138
11-3.	Format of Analog Input Channel Specification Register 0 .....	139
11-4.	Basic Operation of 8-Bit A/D Converter .....	141
11-5.	Relationships between Analog Input Voltage and A/D Conversion Result .....	142
11-6.	Software-Started A/D Conversion .....	143
11-7.	How to Reduce Current Consumption in Standby Mode .....	144
11-8.	Conversion Result Readout Timing (When Conversion Result Is Undefined Value) .....	145
11-9.	Conversion Result Readout Timing (When Conversion Result Is Normal Value) .....	145
11-10.	Analog Input Pin Treatment .....	146
11-11.	A/D Conversion End Interrupt Request Generation Timing .....	147
11-12.	AV <sub>DD</sub> Pin Treatment .....	147
12-1.	Block Diagram of 10-Bit A/D Converter .....	150
12-2.	Format of A/D Converter Mode Register 0 .....	152
12-3.	Format of Analog Input Channel Specification Register 0 .....	153
12-4.	Basic Operation of 10-Bit A/D Converter .....	155
12-5.	Relationships between Analog Input Voltage and A/D Conversion Result .....	156

## LIST OF FIGURES (3/4)

Figure No.	Title	Page
12-6.	Software-Started A/D Conversion .....	157
12-7.	How to Reduce Current Consumption in Standby Mode .....	158
12-8.	Conversion Result Readout Timing (When Conversion Result Is Undefined Value) .....	159
12-9.	Conversion Result Readout Timing (When Conversion Result Is Normal Value) .....	159
12-10.	Analog Input Pin Treatment .....	160
12-11.	A/D Conversion End Interrupt Request Generation Timing .....	161
12-12.	AV <sub>DD</sub> Pin Treatment .....	161
13-1.	Serial Interface 20 Block Diagram .....	164
13-2.	Baud Rate Generator Block Diagram .....	165
13-3.	Serial Operating Mode Register 20 Format .....	167
13-4.	Asynchronous Serial Interface Mode Register 20 Format .....	168
13-5.	Asynchronous Serial Interface Status Register 20 Format .....	170
13-6.	Baud Rate Generator Control Register 20 Format .....	171
13-7.	Asynchronous Serial Interface Transmit/Receive Data Format .....	183
13-8.	Asynchronous Serial Interface Transmission Completion Interrupt Timing .....	185
13-9.	Asynchronous Serial Interface Reception Completion Interrupt Timing .....	186
13-10.	Receive Error Timing .....	187
13-11.	3-Wire Serial I/O Mode Timing .....	192
14-1.	Block Diagram of Multiplier .....	200
14-2.	Multiplier Control Register 0 Format .....	201
14-3.	Multiplier Operation Timing .....	202
15-1.	Basic Configuration of Interrupt Function .....	205
15-2.	Interrupt Request Flag Register Format .....	207
15-3.	Interrupt Mask Flag Register Format .....	208
15-4.	External Interrupt Mode Register 0 Format .....	209
15-5.	Program Status Word Configuration .....	210
15-6.	Flowchart from Non-Maskable Interrupt Request Generation to Acknowledgement .....	212
15-7.	Timing of Non-Maskable Interrupt Request Acknowledgement .....	212
15-8.	Acknowledging Non-Maskable Interrupt Request .....	212
15-9.	Interrupt Acknowledgement Program Algorithm .....	214
15-10.	Interrupt Request Acknowledgement Timing (Example of MOV A,r) .....	215
15-11.	Interrupt Request Acknowledgement Timing (When Interrupt Request Flag Is Generated at the Last Clock During Instruction Execution) .....	215
15-12.	Example of Multiple Interrupts .....	216
16-1.	Oscillation Stabilization Time Select Register Format .....	220
16-2.	Releasing HALT Mode by Interrupt .....	222
16-3.	Releasing HALT Mode by $\overline{\text{RESET}}$ Input .....	223
16-4.	Releasing STOP Mode by Interrupt .....	225
16-5.	Releasing STOP Mode by $\overline{\text{RESET}}$ Input .....	226

## LIST OF FIGURES (4/4)

Figure No.	Title	Page
17-1.	Block Diagram of Reset Function .....	227
17-2.	Reset Timing by $\overline{\text{RESET}}$ Input .....	228
17-3.	Reset Timing by Overflow in Watchdog Timer .....	228
17-4.	Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode .....	228
18-1.	Communication Mode Selection Format .....	232
18-2.	Flashpro III Connection in 3-Wire Serial I/O Mode .....	233
18-3.	Flashpro III Connection in UART Mode .....	234
18-4.	Flashpro III Connection in Pseudo 3-Wire Mode (When P0 Is Used) .....	234
18-5.	Flashpro III Connection in 3-Wire Serial I/O Mode .....	235
18-6.	Flashpro III Connection in UART Mode .....	235
18-7.	Flashpro III Connection in Pseudo 3-Wire Mode (When P0 Is Used) .....	236
A-1.	Development Tool Configuration .....	252

## LIST OF TABLES (1/2)

Table No.	Title	Page
1-1.	Differences between Standard Quality Grade Products and (A) Products .....	30
2-1.	Differences between Standard Quality Grade Products and (A) Products .....	38
3-1.	Types of Pin Input/Output Circuits and Recommended Connection of Unused Pins .....	44
4-1.	Internal ROM Capacity .....	51
4-2.	Vector Table .....	51
4-3.	Special Function Register List .....	61
5-1.	Port Functions .....	74
5-2.	Configuration of Port .....	75
5-3.	Port Mode Register and Output Latch Settings When Using Alternate Functions .....	83
6-1.	Configuration of Clock Generator .....	87
6-2.	Maximum Time Required for Switching CPU Clock .....	93
7-1.	Configuration of Clock Generator .....	95
7-2.	Maximum Time Required for Switching CPU Clock .....	101
8-1.	Configuration of 16-Bit Timer 20 .....	104
8-2.	Interval Time of 16-Bit Timer 20 .....	109
8-3.	Settings of Capture Edge .....	112
9-1.	Interval Time of 8-Bit Timer/Event Counter 80 .....	115
9-2.	Square Wave Output Range of 8-Bit Timer/Event Counter 80 .....	115
9-3.	8-Bit Timer/Event Counter 80 Configuration .....	116
9-4.	Interval Time of 8-Bit Timer/Event Counter 80 (At $f_x = 5.0\text{-MHz}$ Operation) .....	120
9-5.	Interval Time of 8-Bit Timer/Event Counter 80 (At $f_{cc} = 4.0\text{-MHz}$ Operation) .....	120
9-6.	Square Wave Output Range of 8-Bit Timer/Event Counter 80 (At $f_x = 5.0\text{-MHz}$ Operation) .....	123
9-7.	Square Wave Output Range of 8-Bit Timer/Event Counter 80 (At $f_{cc} = 4.0\text{-MHz}$ Operation) .....	123
10-1.	Runaway Detection Time of Watchdog Timer .....	129
10-2.	Interval Time .....	129
10-3.	Configuration of Watchdog Timer .....	130
10-4.	Runaway Detection Time of Watchdog Timer .....	133
10-5.	Interval Time of Interval Timer .....	134
11-1.	Configuration of 8-Bit A/D Converter .....	135
12-1.	Configuration of 10-Bit A/D Converter .....	149

## LIST OF TABLES (2/2)

Table No.	Title	Page
13-1.	Serial Interface 20 Configuration .....	163
13-2.	Serial Interface 20 Operating Mode Settings .....	169
13-3.	Example of Relationships between System Clock and Baud Rate .....	172
13-4.	Relationship between ASCK20 Pin Input Frequency and Baud Rate (When BRGC20 Is Set to 80H) .....	173
13-5.	Example of Relationship between System Clock and Baud Rate .....	181
13-6.	Relationship between ASCK20 Pin Input Frequency and Baud Rate (When BRGC20 Is Set to 80H) .....	182
13-7.	Receive Error Causes .....	187
15-1.	Interrupt Source List .....	204
15-2.	Flags Corresponding to Interrupt Request Signal .....	206
15-3.	Time from Generation of Maskable Interrupt Request to Processing .....	213
16-1.	HALT Mode Operating Status .....	221
16-2.	Operation after Release of HALT Mode .....	223
16-3.	STOP Mode Operating Status .....	224
16-4.	Operation after Release of STOP Mode .....	226
17-1.	Hardware Status after Reset .....	229
18-1.	Differences between Flash Memory and Mask ROM Versions .....	231
18-2.	Communication Mode .....	232
18-3.	Functions of Flash Memory Programming .....	233
18-4.	Example of Settings for PG-FP3 .....	237
19-1.	Selection of Mask Option for Pins .....	239
20-1.	Operand Identifiers and Description Methods .....	241

## CHAPTER 1 GENERAL ( $\mu$ PD789104A, 789114A SUBSERIES)

### 1.1 Features

- ROM and RAM capacities

Part Number \ Item	Program Memory		Data Memory (Internal High-Speed RAM)
$\mu$ PD789101A, 789111A	ROM	2 Kbytes	256 bytes
$\mu$ PD789102A, 789112A		4 Kbytes	
$\mu$ PD789104A, 789114A		8 Kbytes	
$\mu$ PD78F9116A	Flash memory	16 Kbytes	

- System clock: Crystal/ceramic oscillation
- Two minimum instruction execution times selectable: high speed (0.4  $\mu$ s) and low speed (1.6  $\mu$ s) (system clock: 5.0 MHz)
- 20 I/O ports
- Serial interface: 1 channel  
3-wire serial I/O mode/UART mode selectable
- 8-bit resolution A/D converter: 4 channels ( $\mu$ PD789104A Subseries)
- 10-bit resolution A/D converter: 4 channels ( $\mu$ PD789114A Subseries)
- 3 timers
  - 16-bit timer: 1 channel
  - 8-bit timer/event counter: 1 channel
  - Watchdog timer: 1 channel
- Multiplier: 8 bits  $\times$  8 bits = 16 bits
- Vectored interrupt source: 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

Vacuum cleaners, washing machines, refrigerators, battery chargers, etc.

### 1.3 Ordering Information

Part Number	Package	Internal ROM
$\mu$ PD789101AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789102AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789104AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789111AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789112AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789114AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD78F9116AMC-5A4	30-pin plastic SSOP (7.62 mm (300) )	Flash memory
$\mu$ PD789101AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789102AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789104AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789111AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789112AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789114AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM

**Remark** xxx indicates ROM code suffix.

### 1.4 Quality Grade

Part Number	Package	Quality Grade
$\mu$ PD789101AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789102AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789104AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789111AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789112AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789114AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD78F9116AMC-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789101AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789102AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789104AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789111AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789112AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789114AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special

**Remark** xxx indicates ROM code suffix.

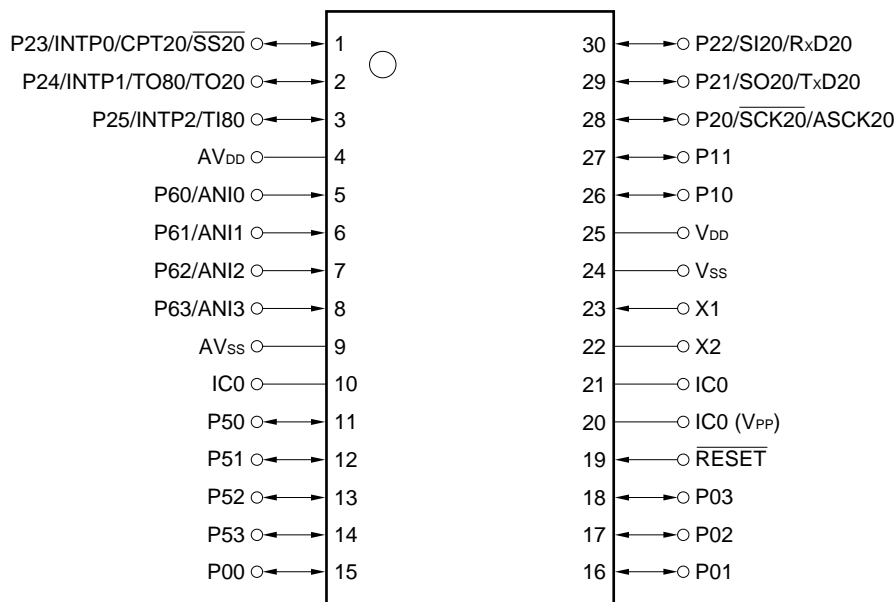
Please refer to "Quality Grades on NEC Semiconductor Devices" (Document No. C11531E) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.



## 1.5 Pin Configuration (Top View)

- 30-pin plastic SSOP (7.62 mm (300) )

$\mu$ PD789101AMC-xxx-5A4     $\mu$ PD789102AMC-xxx-5A4     $\mu$ PD789104AMC-xxx-5A4  
 $\mu$ PD789111AMC-xxx-5A4     $\mu$ PD789112AMC-xxx-5A4     $\mu$ PD789114AMC-xxx-5A4  
 $\mu$ PD78F9116AMC-5A4  
 $\mu$ PD789101AMC(A)-xxx-5A4     $\mu$ PD789102AMC(A)-xxx-5A4     $\mu$ PD789104AMC(A)-xxx-5A4  
 $\mu$ PD789111AMC(A)-xxx-5A4     $\mu$ PD789112AMC(A)-xxx-5A4     $\mu$ PD789114AMC(A)-xxx-5A4



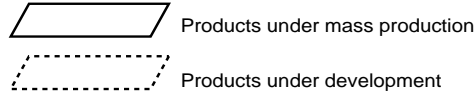
- Cautions**
1. Connect the IC0 (internally connected) pin directly to the V<sub>SS</sub> pin.
  2. Connect the AV<sub>DD</sub> pin to the V<sub>DD</sub> pin.
  3. Connect the AV<sub>SS</sub> pin to the V<sub>SS</sub> pin.

**Remark** Pin connection in parentheses is intended for the  $\mu$ PD78F9116A.

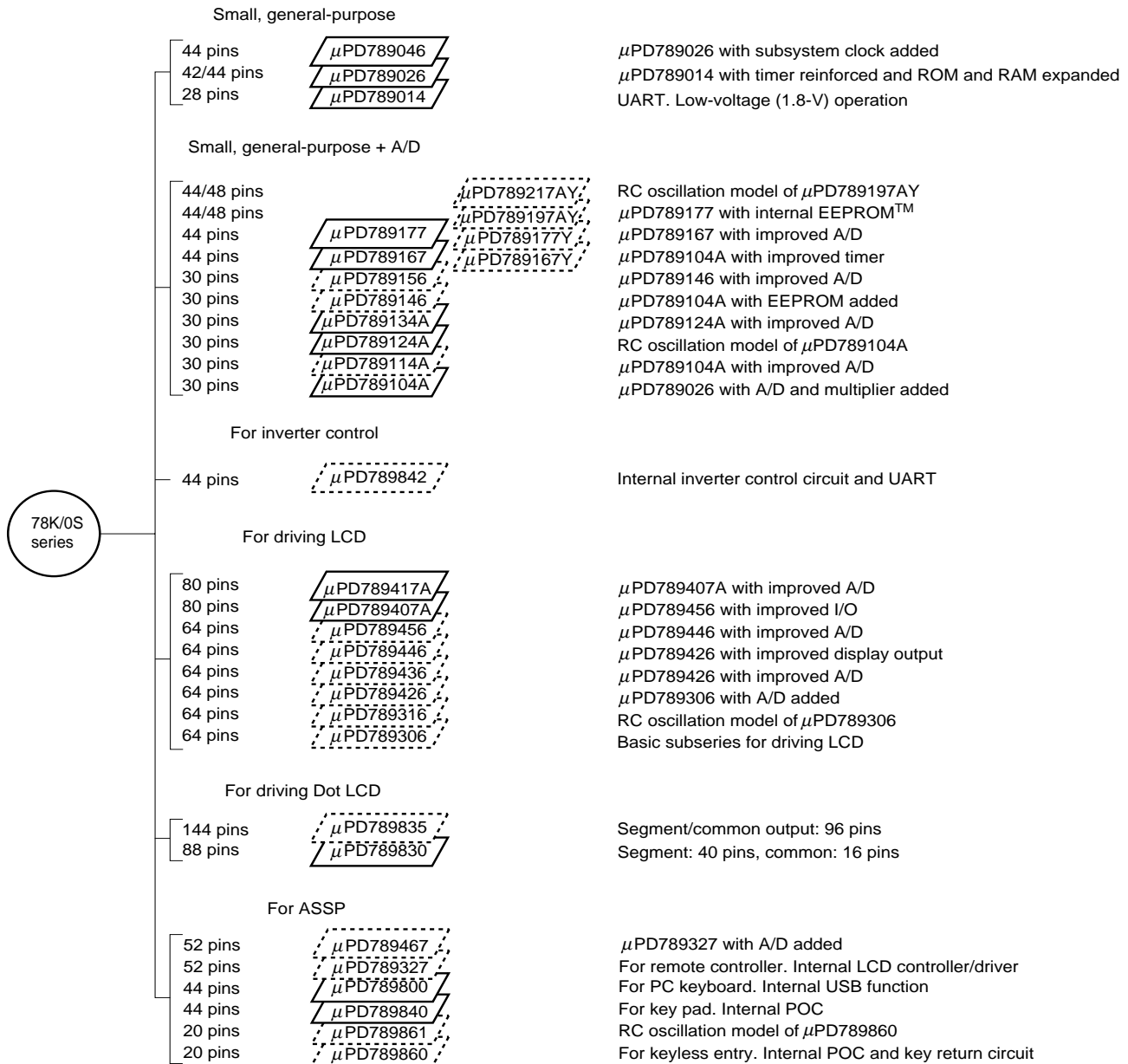
ANI0 to ANI3:	Analog Input	RxD20:	Receive Data
ASCK20:	Asynchronous Serial Input	$\overline{SCK}20$ :	Serial Clock
AV <sub>DD</sub> :	Analog Power Supply	SI20:	Serial Input
AV <sub>SS</sub> :	Analog Ground	SO20:	Serial Output
CPT20:	Capture Trigger Input	$\overline{SS}20$ :	Chip Select Input
IC0:	Internally Connected	TI80:	Timer Input
INTP0 to INTP2:	Interrupt from Peripherals	TO20, TO80:	Timer Output
P00 to P03:	Port 0	TxD20:	Transmit Data
P10, P11:	Port 1	V <sub>DD</sub> :	Power Supply
P20 to P25:	Port 2	V <sub>PP</sub> :	Programming Power Supply
P50 to P53:	Port 5	V <sub>SS</sub> :	Ground
P60 to P63:	Port 6	X1, X2:	Crystal 1, 2
$\overline{RESET}$ :	Reset		

## 1.6 78K/0S Series Lineup

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



Y subseries supports SMB.

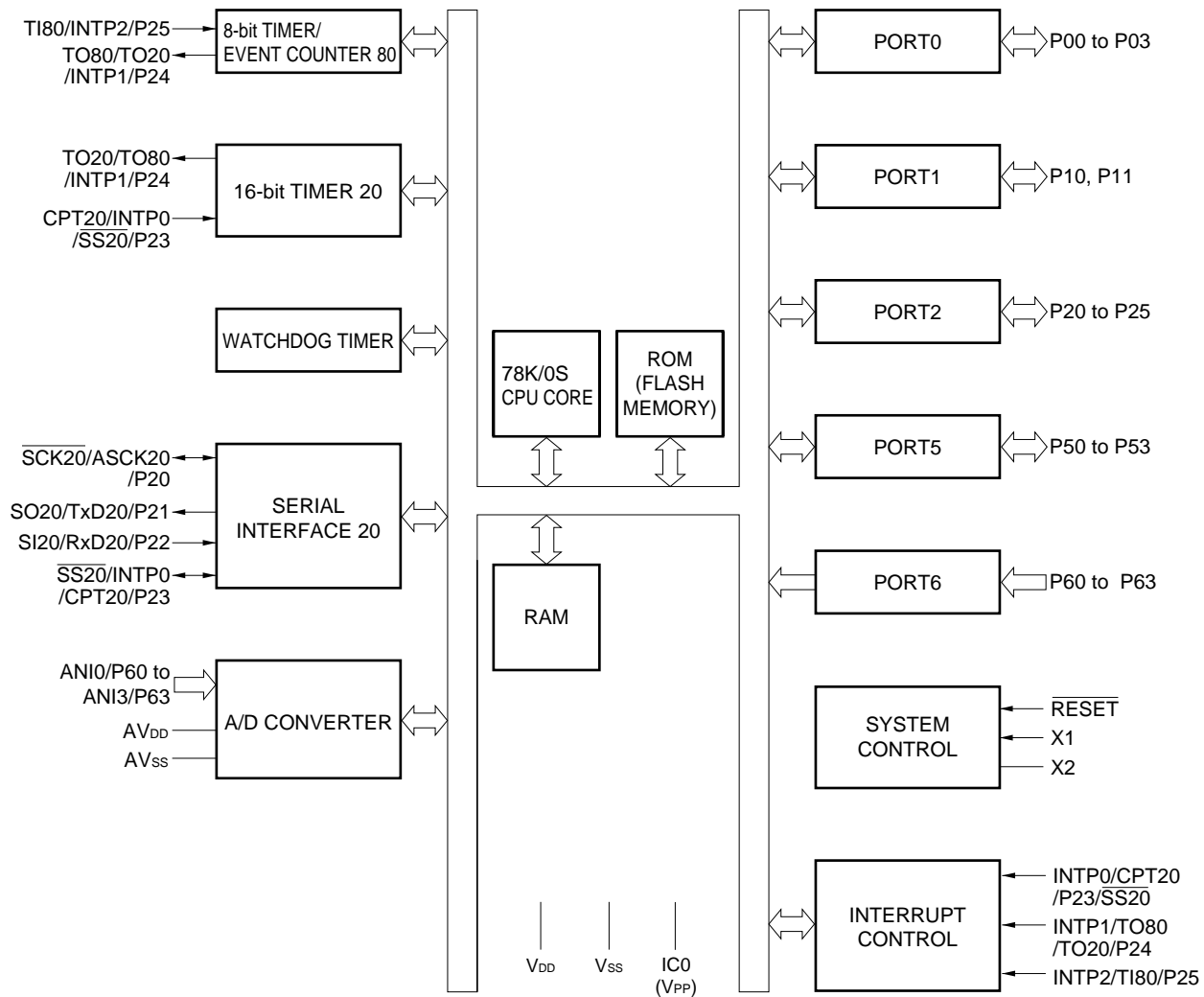


The major differences between subseries are shown below.

Function Subseries Name		ROM Capacity	Timer				8-bit A/D	10-bit A/D	Serial Interface	I/O	V <sub>DD</sub> MIN Value	Remark
			8-bit	16-bit	Watch	WDT						
Small, general- purpose	μPD789046	16 K	1 ch	1 ch	1 ch	1 ch	—	—	1 ch (UART: 1 ch)	34 pins	1.8 V	—
	μPD789026	4 K-16 K			—							
	μPD789014	2 K-4 K	2 ch	—						22 pins		
Small, general- purpose + A/D	μPD789177	16 K-24 K	3 ch	1 ch	1 ch	1 ch	—	8 ch	1 ch (UART: 1 ch)	31 pins	1.8 V	—
	μPD789167						8 ch	—				
	μPD789156	8 K-16 K	1 ch		—		—	4 ch		20 pins		Internal EEPROM
	μPD789146						4 ch	—				
	μPD789134A	2 K-8 K						4 ch				RC oscillation version
	μPD789124A						4 ch	—				
	μPD789114A						—	4 ch				—
	μPD789104A						4 ch	—				
For inverter control	μPD789842	8 K-16 K	3 ch	<b>Note</b>	1 ch	1 ch	8 ch	—	1 ch (UART: 1 ch)	30 pins	4.0 V	—
For LCD driving	μPD789417A	12 K-24 K	3 ch	1 ch	1 ch	1 ch		7 ch	1 ch (UART: 1 ch)	43 pins	1.8 V	—
	μPD789407A						7 ch	—				
	μPD789456	12 K-16 K	2 ch				—	6 ch		30 pins		
	μPD789446						6 ch	—				
	μPD789436						—	6 ch		40 pins		
	μPD789426						6 ch	—				
	μPD789316	8 K to 16K					—		2 ch (UART: 1 ch)	23 pins		RC oscillation version
	μPD789306											—
For Dot LCD driving	μPD789835	24 K-60 K	6 ch	—	1 ch	1 ch	2 ch	—	1 ch	27 pins	1.8 V	—
	μPD789830	24 K	1 ch	1 ch			—		1 ch (UART: 1 ch)	30 pins	2.7 V	
ASSP	μPD789467	4 K-24 K	2 ch	—	1 ch	1 ch	1 ch	—	—	18 pins	1.8 V	Internal LCD
	μPD789327						—		1 ch	21 pins		
	μPD789800	8 K	2 ch	1 ch	—	1 ch	—		2 ch (USB: 1 ch)	31 pins	4.0 V	—
	μPD789840						4 ch		1 ch	29 pins	2.8 V	
	μPD789861	4 K		—			—		—	14 pins	1.8 V	RC oscillation version, Internal EEPROM
	μPD789860											Internal EEPROM

**Note** 10-bit timer: 1 channel

## 1.7 Block Diagram



- Remarks**
1. The size of the internal ROM varies depending on the product.
  2. Items in parentheses apply to the μPD78F9116A.

## 1.8 Outline of Functions

Item		$\mu$ PD789101A $\mu$ PD789111A	$\mu$ PD789102A $\mu$ PD789112A	$\mu$ PD789104A $\mu$ PD789114A	$\mu$ PD78F9116A
Mask ROM		Mask ROM			Flash memory
Internal memory	ROM	2 Kbytes	4 Kbytes	8 Kbytes	16 Kbytes
	High-speed RAM	256 bytes			
System clock		Crystal/ceramic oscillation			
Minimum instruction execution time		0.4/1.6 $\mu$ s (@5.0-MHz operation with system clock)			
General registers		8 bits $\times$ 8 registers			
Instruction set		<ul style="list-style-type: none"> <li>16-bit operations</li> <li>Bit manipulations (such as set, reset, and test)</li> </ul>			
Multiplier		8 bits $\times$ 8 bits = 16 bits			
I/O ports		Total: 20 <ul style="list-style-type: none"> <li>CMOS input: 4</li> <li>CMOS I/O: 12</li> <li>N-ch open-drain (12-V withstand voltage): 4</li> </ul>			
A/D converter		8-bit resolution $\times$ 4 channels ( $\mu$ PD789104A Subseries) 10-bit resolution $\times$ 4 channels ( $\mu$ PD789114A Subseries)			
Serial interface		3-wire serial I/O mode/UART mode selectable: 1 channel			
Timer		16-bit timer: 1 channel 8-bit timer/event counter: 1 channel Watchdog timer: 1 channel			
Timer outputs		One output			
Vectored interrupt	Maskable	Internal: 6, External: 3			
	Non-maskable	Internal: 1			
Power supply voltage		$V_{DD} = 1.8$ to $5.5$ V			
Operating ambient temperature		$T_A = -40$ to $+85^\circ\text{C}$			
Packages		30-pin plastic SSOP (7.62 mm (300) )			

An outline of the timers is shown below.

		16-Bit Timer 20	8-Bit Timer/Event Counter 80	Watchdog Timer
Operating Mode	Interval timer	—	1 channel	1 channel <sup>Note</sup>
	External event timer	—	1 channel	—
Function	Timer output	1 output	1 output	—
	PWM output	—	1 output	—
	Square wave output	—	1 output	—
	Capture	1 input	—	—
	Interrupt source	1	1	1

**Note** Since the watchdog timer provides the watchdog timer function and interval timer function, select the one out of two functions.

## 1.9 Differences between Standard quality Grade Products and (A) Products

Table 1-1 shows the differences between the standard quality grade products ( $\mu$ PD789101A, 789102A, 789104A, 789111A, 789112A, 789114A) and (A) products ( $\mu$ PD789101A(A), 789102A(A), 789104A(A), 789111A(A), 789112A(A), 789114A(A)).

**Table 1-1. Differences between Standard Quality Grade Products and (A) Products**

Item \ Part Number	Standard Quality Grade Products	(A) Products
	Standard	Special
Quality grade	Standard	Special
Electrical Specifications	Refer to separate Data Sheets	

## CHAPTER 2 GENERAL ( $\mu$ PD789124A, 789134A SUBSERIES)

### 2.1 Features

- ROM and RAM capacities

Part Number \ Item	Program Memory		Data Memory (Internal High-Speed RAM)
$\mu$ PD789121A, 789131A	ROM	2 Kbytes	256 bytes
$\mu$ PD789122A, 789132A		4 Kbytes	
$\mu$ PD789124A, 789134A		8 Kbytes	
$\mu$ PD78F9136A	Flash memory	16 Kbytes	

- System clock: RC oscillation
- Two minimum instruction execution times selectable: high speed ( $0.5\ \mu\text{s}$ ) and low speed ( $2.0\ \mu\text{s}$ ) (system clock: 4.0 MHz)
- 20 I/O ports
- Serial interface: 1 channel
  - 3-wire serial I/O mode/UART mode selectable
- 8-bit resolution A/D converter: 4 channels ( $\mu$ PD789124A Subseries)
- 10-bit resolution A/D converter: 4 channels ( $\mu$ PD789134A Subseries)
- 3 timers
  - 16-bit timer: 1 channel
  - 8-bit timer/event counter: 1 channel
  - Watchdog timer: 1 channel
- Multiplier: 8 bits  $\times$  8 bits = 16 bits
- Vectored interrupt source: 10
- Supply voltage:  $V_{DD} = 1.8$  to 5.5 V
- Operating ambient temperature:  $T_A = -40$  to  $+85^\circ\text{C}$

### 2.2 Applications

Vacuum cleaners, washing machines, refrigerators, battery chargers, etc.

## 2.3 Ordering Information

Part Number	Package	Internal ROM
$\mu$ PD789121AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789122AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789124AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789131AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789132AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789134AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD78F9136AMC-5A4	30-pin plastic SSOP (7.62 mm (300) )	Flash memory
$\mu$ PD789121AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789122AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789124AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789131AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789132AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM
$\mu$ PD789134AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Mask ROM

**Remark** xxx indicates ROM code suffix.

## 2.4 Quality Grade

Part Number	Package	Quality Grade
$\mu$ PD789121AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789122AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789124AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789131AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789132AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789134AMC-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD78F9136AMC-5A4	30-pin plastic SSOP (7.62 mm (300) )	Standard
$\mu$ PD789121AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789122AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789124AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789131AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789132AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special
$\mu$ PD789134AMC(A)-xxx-5A4	30-pin plastic SSOP (7.62 mm (300) )	Special

**Remark** xxx indicates ROM code suffix.

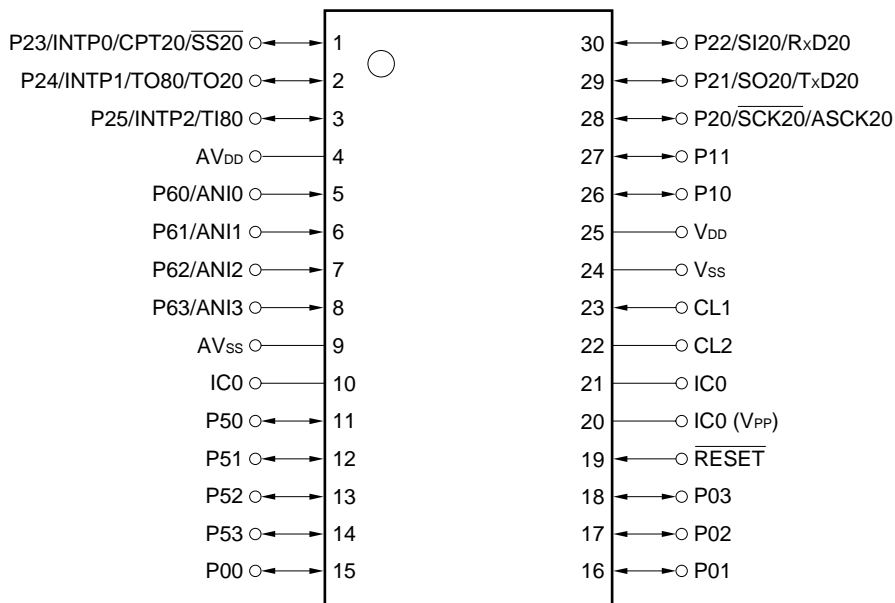
Please refer to "Quality Grades on NEC Semiconductor Devices" (Document No. C11531E) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.



## 2.5 Pin Configuration (Top View)

- 30-pin plastic SSOP (7.62 mm (300) )

μPD789121AMC-xxx-5A4    μPD789122AMC-xxx-5A4    μPD789124AMC-xxx-5A4  
 μPD789131AMC-xxx-5A4    μPD789132AMC-xxx-5A4    μPD789134AMC-xxx-5A4  
 μPD78F9136AMC-5A4  
 μPD789121AMC(A)-xxx-5A4    μPD789122AMC(A)-xxx-5A4    μPD789124AMC(A)-xxx-5A4  
 μPD789131AMC(A)-xxx-5A4    μPD789132AMC(A)-xxx-5A4    μPD789134AMC(A)-xxx-5A4




- Cautions**
1. Connect the IC0 (internally connected) pin directly to the Vss pin.
  2. Connect the AVDD pin to the VDD pin.
  3. Connect the AVss pin to the Vss pin.

**Remark** Pin connection in parentheses is intended for the μPD78F9136A.

ANI0 to ANI3:	Analog Input	RESET:	Reset
ASCK20:	Asynchronous Serial Input	RxD20:	Receive Data
AVDD:	Analog Power Supply	SCK20:	Serial Clock
AVss:	Analog Ground	SI20:	Serial Input
CL1, CL2:	RC oscillator	SO20:	Serial Output
CPT20:	Capture Trigger Input	SS20:	Chip Select Input
IC0:	Internally Connected	TI80:	Timer Input
INTP0 to INTP2:	Interrupt from Peripherals	TO20, TO80:	Timer Output
P00 to P03:	Port 0	TxD20:	Transmit Data
P10, P11:	Port 1	VDD:	Power Supply
P20 to P25:	Port 2	VPP:	Programming Power Supply
P50 to P53:	Port 5	Vss:	Ground
P60 to P63:	Port 6		

## 2.6 78K/0S Series Lineup

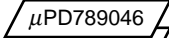
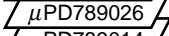
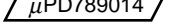
The products in the 78K/0S Series are listed below. The names enclosed in boxes are subseries names.

 Products under mass production


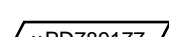
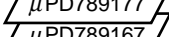
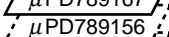
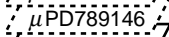
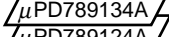
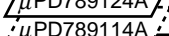
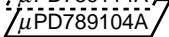

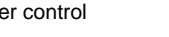
 Products under development

Y subseries supports SMB.

### Small, general-purpose

44 pins		μPD789026 with subsystem clock added
42/44 pins		μPD789014 with timer reinforced and ROM and RAM expanded
28 pins		UART. Low-voltage (1.8-V) operation

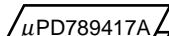
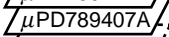
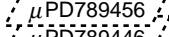
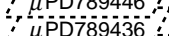
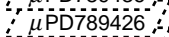
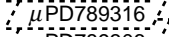
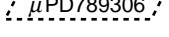
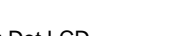
### Small, general-purpose + A/D

44/48 pins		RC oscillation model of μPD789197AY
44/48 pins		μPD789177 with internal EEPROM™
44 pins		μPD789167 with improved A/D
44 pins		μPD789104A with improved timer
30 pins		μPD789146 with improved A/D
30 pins		μPD789104A with EEPROM added
30 pins		μPD789124A with improved A/D
30 pins		RC oscillation model of μPD789104A
30 pins		μPD789104A with improved A/D
30 pins		μPD789026 with A/D and multiplier added

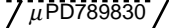
### For inverter control

44 pins		Internal inverter control circuit and UART
---------	---	--

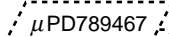
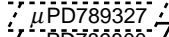
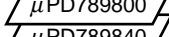
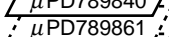
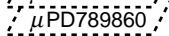

### For driving LCD

80 pins		μPD789407A with improved A/D
80 pins		μPD789456 with improved I/O
64 pins		μPD789446 with improved A/D
64 pins		μPD789426 with improved display output
64 pins		μPD789426 with improved A/D
64 pins		μPD789306 with A/D added
64 pins		RC oscillation model of μPD789306
64 pins		Basic subseries for driving LCD

### For driving Dot LCD

144 pins		Segment/common output: 96 pins
88 pins		Segment: 40 pins, common: 16 pins

### For ASSP

52 pins		μPD789327 with A/D added
52 pins		For remote controller. Internal LCD controller/driver
44 pins		For PC keyboard. Internal USB function
44 pins		For key pad. Internal POC
20 pins		RC oscillation model of μPD789860
20 pins		For keyless entry. Internal POC and key return circuit

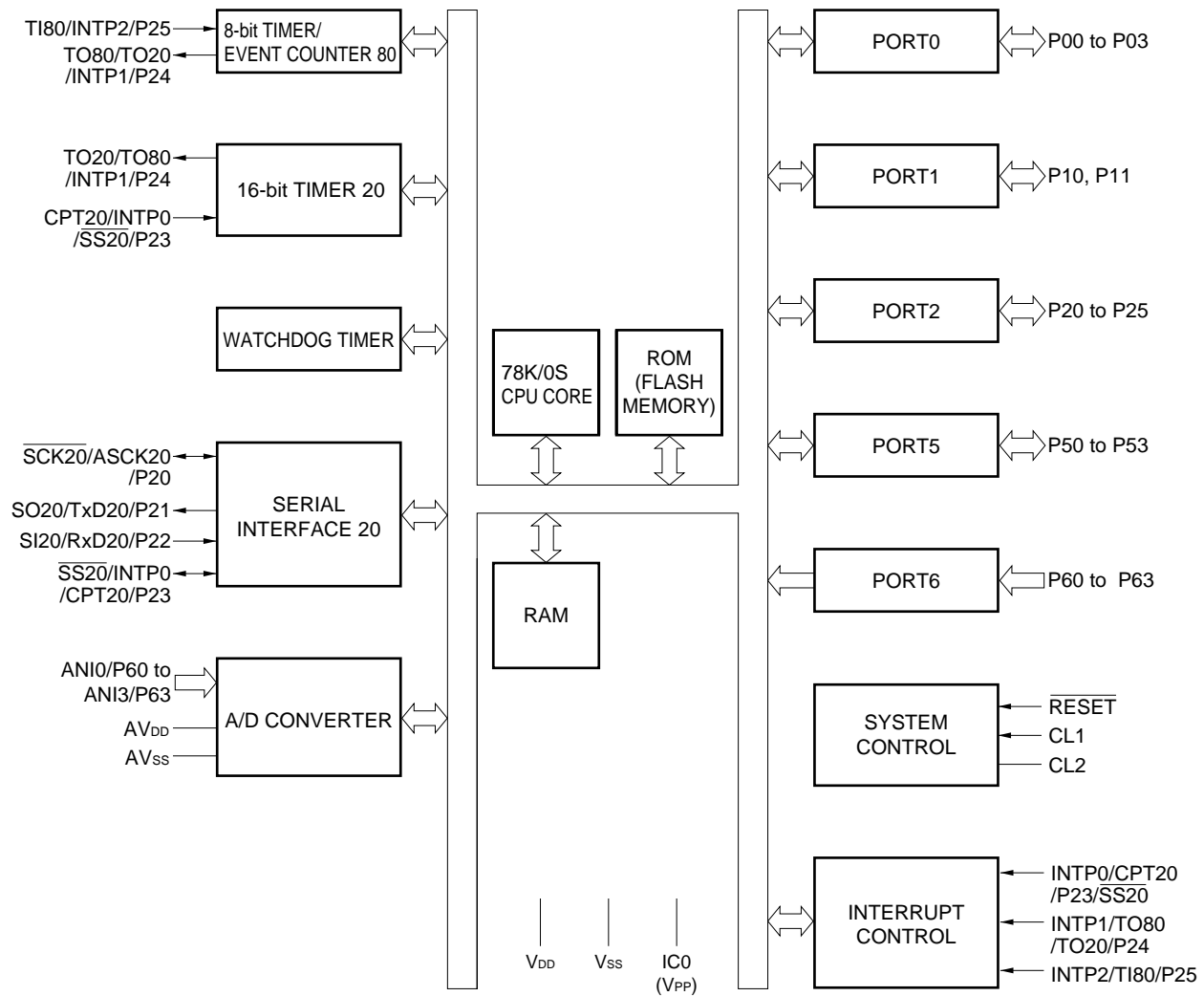
78K/0S  
series

The major differences between subseries are shown below.

Function Subseries Name		ROM Capacity	Timer				8-bit A/D	10-bit A/D	Serial Interface	I/O	V <sub>DD</sub> MIN Value	Remark
			8-bit	16-bit	Watch	WDT						
Small, general- purpose	μPD789046	16 K	1 ch	1 ch	1 ch	1 ch	—	—	1 ch (UART: 1 ch)	34 pins	1.8 V	—
	μPD789026	4 K-16 K			—							
	μPD789014	2 K-4 K	2 ch	—						22 pins		
Small, general- purpose + A/D	μPD789177	16 K-24 K	3 ch	1 ch	1 ch	1 ch	—	8 ch	1 ch (UART: 1 ch)	31 pins	1.8 V	—
	μPD789167						8 ch	—				
	μPD789156	8 K-16 K	1 ch		—		—	4 ch		20 pins		Internal EEPROM
	μPD789146						4 ch	—				
	μPD789134A	2 K-8 K						4 ch				RC oscillation version
	μPD789124A						4 ch	—				
	μPD789114A						—	4 ch				—
	μPD789104A						4 ch	—				
For inverter control	μPD789842	8 K-16 K	3 ch	<b>Note</b>	1 ch	1 ch	8 ch	—	1 ch (UART: 1 ch)	30 pins	4.0 V	—
For LCD driving	μPD789417A	12 K-24 K	3 ch	1 ch	1 ch	1 ch		7 ch	1 ch (UART: 1 ch)	43 pins	1.8 V	—
	μPD789407A						7 ch	—				
	μPD789456	12 K-16 K	2 ch				—	6 ch		30 pins		
	μPD789446						6 ch	—				
	μPD789436						—	6 ch		40 pins		
	μPD789426						6 ch	—				
	μPD789316	8 K to 16K					—		2 ch (UART: 1 ch)	23 pins		RC oscillation version
	μPD789306											—
For Dot LCD driving	μPD789835	24 K-60 K	6 ch	—	1 ch	1 ch	2 ch	—	1 ch	27 pins	1.8 V	—
	μPD789830	24 K	1 ch	1 ch			—		1 ch (UART: 1 ch)	30 pins	2.7 V	
ASSP	μPD789467	4 K-24 K	2 ch	—	1 ch	1 ch	1 ch	—	—	18 pins	1.8 V	Internal LCD
	μPD789327						—		1 ch	21 pins		
	μPD789800	8 K	2 ch	1 ch	—	1 ch	—		2 ch (USB: 1 ch)	31 pins	4.0 V	—
	μPD789840						4 ch		1 ch	29 pins	2.8 V	
	μPD789861	4 K		—			—		—	14 pins	1.8 V	RC oscillation version, Internal EEPROM
	μPD789860											Internal EEPROM

**Note** 10-bit timer: 1 channel

## 2.7 Block Diagram



- Remarks**
1. The size of the internal ROM varies depending on the product.
  2. Items in parentheses apply to the μPD78F9136A.

## 2.8 Outline of Functions

Item		μPD789121A μPD789131A	μPD789122A μPD789132A	μPD789124A μPD789134A	μPD78F9136A
Internal memory	ROM	Mask ROM			Flash memory
		2 Kbytes	4 Kbytes	8 Kbytes	16 Kbytes
	High-speed RAM	256 bytes			
System clock		Crystal/ceramic oscillation			
Minimum instruction execution time		0.5/2.0 μs (@4.0-MHz operation with system clock)			
General registers		8 bits × 8 registers			
Instruction set		<ul style="list-style-type: none"> <li>16-bit operations</li> <li>Bit manipulations (such as set, reset, and test)</li> </ul>			
Multiplier		8 bits × 8 bits = 16 bits			
I/O ports		Total: 20 • CMOS input: 4 • CMOS I/O: 12 • N-ch open-drain (12-V withstand voltage ): 4			
A/D converter		8-bit resolution × 4 channels (μPD789124A Subseries) 10-bit resolution × 4 channels (μPD789134A Subseries)			
Serial interface		3-wire serial I/O mode/UART mode selectable: 1 channel			
Timer		16-bit timer: 1 channel 8-bit timer/event counter: 1 channel Watchdog timer: 1 channel			
Timer outputs		One output			
Vectored interrupt	Maskable	Internal: 6, External: 3			
	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			
Packages		30-pin plastic SSOP (7.62 mm (300) )			

An outline of the timers is shown below.

		16-Bit Timer 20	8-Bit Timer/Event Counter 80	Watchdog Timer
Operating Mode	Interval timer	—	1 channel	1 channel <sup>Note</sup>
	External event timer	—	1 channel	—
Function	Timer output	1 output	1 output	—
	PWM output	—	1 output	—
	Square wave output	—	1 output	—
	Capture	1 input	—	—
	Interrupt source	1	1	1

**Note** Since the watchdog timer provides the watchdog timer function and interval timer function, select the one out of two functions.

## 2.9 Differences between Standard quality Grade Products and (A) Products

Table 2-1 shows the differences between the standard quality grade products ( $\mu$ PD789121A, 789122A, 789124A, 789131A, 789132A, 789134A) and (A) products ( $\mu$ PD789121A(A), 789122A(A), 789124A(A), 789131A(A), 789132A(A), 789134A(A)).

**Table 2-1. Differences between Standard Quality Grade Products and (A) Products**

Item \ Part Number	Standard Quality Grade Products	(A) Products
	Standard	Special
Quality grade	Standard	Special
Electrical Specifications	Refer to separate Data Sheets	

## CHAPTER 3 PIN FUNCTIONS

### 3.1 Pin Function List

#### (1) Port pins

Pin Name	Input/Output	Function	After Reset	Alternate Function
P00 to P03	Input/output	Port 0 4-bit input/output port Input/output can be specified in 1-bit units When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	—
P10, P11	Input/output	Port 1 2-bit input/output port Input/output can be specified in 1-bit units When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	—
P20	Input/output	Port 2 6-bit input/output port Input/output can be specified in 1-bit units An on-chip pull-up resistor can be specified by means of pull-up resistor option register B2 (PUB2)	Input	$\overline{\text{SCK20}}/\text{ASCK20}$
P21				SO20/TxD20
P22				SI20/RxD20
P23				$\overline{\text{INTP0/CPT20/SS20}}$
P24				$\overline{\text{INTP1/TO80/TO20}}$
P25				$\overline{\text{INTP2/TI80}}$
P50 to P53	Input/output	Port 5 4-bit N-channel open-drain input/output port Input/output can be specified in 1-bit units For a mask ROM version, an on-chip pull-up resistor can be specified by the mask option.	Input	—
P60 to P63	Input	Port 6 4-bit input port	Input	ANI0 to ANI3

## (2) Non-port pins

Pin Name	Input/Output	Function	After Reset	Alternate Function
INTP0	Input	External interrupt input for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.	Input	P23/CPT20/ $\overline{SS20}$
INTP1				P24/TO80/TO20
INTP2				P25/TI80
SI20	Input	Serial data input to serial interface	Input	P22/RxD20
SO20	Output	Serial data output from serial interface	Input	P21/TxD20
$\overline{SCK20}$	Input/output	Serial clock input/output for serial interface	Input	P20/ASCK20
ASCK20	Input	Serial clock input to asynchronous serial interface	Input	P20/ $\overline{SCK20}$
$\overline{SS20}$	Input	Chip select input to serial interface	Input	P23/CPT20/INTP0
RxD20	Input	Serial data input to asynchronous serial interface	Input	P22/SI20
TxD20	Output	Serial data output from asynchronous serial interface	Input	P21/SO20
TI80	Input	External count clock input to 8-bit timer (TM80)	Input	P25/INTP2
TO80	Output	8-bit timer (TM80) output	Input	P24/INTP1/TO20
TO20	Output	16-bit timer (TM20) output	Input	P24/INTP1/TO80
CPT20	Input	Capture edge input	Input	P23/INTP0/ $\overline{SS20}$
ANI0 to ANI3	Input	A/D converter analog input	Input	P60 to P63
AV <sub>SS</sub>	—	A/D converter ground potential	—	—
AV <sub>DD</sub>	—	A/D converter analog power supply	—	—
X1	Input	Connecting crystal resonator for system clock oscillation ( $\mu$ PD789104A, 789114A Subseries)	—	—
X2	—		—	—
CL1	Input	Connecting resistor (R) and capacitor (C) for system clock oscillation ( $\mu$ PD789124A and 789134A Subseries)	—	—
CL2	—		—	—
$\overline{RESET}$	Input	System reset input	Input	—
V <sub>DD</sub>	—	Positive power supply	—	—
V <sub>SS</sub>	—	Ground potential	—	—
IC0	—	Internally connected. Directly connect to the V <sub>SS</sub> pin.	—	—
V <sub>PP</sub>	—	Sets flash memory programming mode. Applies high voltage when a program is written or verified. Connect directly to V <sub>SS</sub> in normal operation mode.	—	—



## 3.2 Description of Pin Functions

### 3.2.1 P00 to P03 (Port 0)

These pins constitute a 4-bit I/O port and can be set in 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, use of an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).

### 3.2.2 P10, P11 (Port 1)

These pins constitute a 2-bit I/O port and can be set in 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, use of an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).

### 3.2.3 P20 to P25 (Port 2)

These pins constitute a 6-bit I/O port. In addition, they function as timer input/outputs, external interrupt inputs, and serial interface data and clock input/outputs.

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

#### (1) Port mode

In this mode, P20 to P25 function as a 6-bit I/O port. Port 2 can be specified as input or output mode in 1-bit units by using port mode register 2 (PM2). Use of an on-chip pull-up resistor can be specified in 1-bit units by using pull-up resistor option register B2 (PUB2), regardless of the setting of port mode register 2 (PM2).

#### (2) Control mode

In this mode, P20 to P25 function as the timer input/output, the external interrupt input, and the clock input/output of the serial interface and the data input/output.

##### (a) TI80

This is the external clock input pin for 8-bit timer/event counter 80.

##### (b) TO20, TO80

TO20 is the output pin of the 16-bit timer. TO80 is the output pin of the 8-bit timer.

##### (c) CPT20

This is the input pin of the capture edge.

##### (d) INTP0 to INTP2

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

##### (e) SI20, SO20

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

##### (f) $\overline{\text{SCK20}}$

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

##### (g) $\overline{\text{SS20}}$

This is the chip select input pin of the serial interface.

**(h) RxD20, TxD20**

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

**(i) ASCK20**

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

**Caution** When using these pins as serial interface pins, the input/output mode and output latch must be set according to the functions to be used. For the details of the setting, refer to Table 13-2 Serial Interface 20 Operating Mode Settings.

**3.2.4 P50 to P53 (Port 5)**

This is a 4-bit N-ch open-drain I/O port. Port 5 can be specified in input or output mode in 1-bit units by using port mode register 5 (TM5). For a mask ROM version, use of an on-chip pull-up resistor can be specified by the mask option.

**3.2.5 P60 to P63 (Port 6)**

This is a 4-bit input-only port. In addition to general-purpose input ports, these pins function as the A/D converter input pins.

**(1) Port mode**

In the port mode, port 6 functions as a 4-bit input-only port.

**(2) Control mode**

In the control mode, the pins of port 6 can be used as A/D converter analog inputs (ANI0 to ANI3).

**3.2.6 RESET**

This pin inputs an active-low system reset signal.

**3.2.7 X1, X2 ( $\mu$ PD789104A, 789114A Subseries)**

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.

**3.2.8 CL1, CL2 ( $\mu$ PD789124A, 789134A Subseries)**

Resistor (R) and capacitor (C) connect pins for system clock oscillation.

**3.2.9 AV<sub>DD</sub>**

Analog power supply pin of the A/D converter. Always use the same potential as that of the V<sub>DD</sub> pin even when the A/D converter is not used.

**3.2.10 AV<sub>SS</sub>**

This is a ground potential pin of the A/D converter. Always use the same potential as that of the V<sub>SS</sub> pin even when the A/D converter is not used.

**3.2.11 V<sub>DD</sub>**

Positive power supply pin

**3.2.12 V<sub>SS</sub>**

Ground pin

**3.2.13 V<sub>PP</sub> ( $\mu$ PD78F9116A, 78F9136A only)**

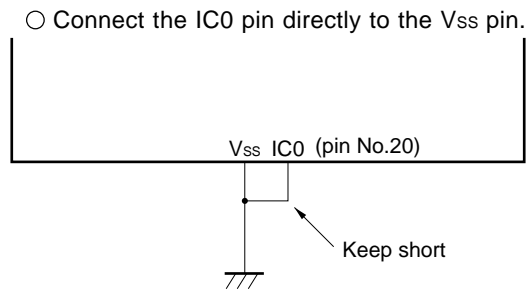
A high voltage should be applied to this pin when the flash memory programming mode is set and when the program is written or verified.

Directly connect this pin to V<sub>ss</sub> in the normal operation mode.

**3.2.14 IC0 (pin No.20) (mask ROM version only)**

The IC0 (Internally Connected) pin (No. 20) (refer to **1.5 Pin Configuration (Top View)**, **2.5 Pin Configuration (Top View)**) is used to set the  $\mu$ PD789104A/114A/124A/134A subseries in the test mode before shipment. In the normal operation mode, connect this pin directly to the V<sub>ss</sub> pin with as short a wiring length as possible.

If a potential difference is generated between the IC0 pin and V<sub>ss</sub> pin due to a long wiring length between the IC0 pin and V<sub>ss</sub> pin or external noise superimposed on the IC0 pin, the user program may not run correctly.

**3.2.15 IC0 (pins No.10 and No.21)**

The IC0 pins (No.10 and No.21) (refer to **1.5 Pin Configuration (Top View)**, **2.5 Pin Configuration (Top View)**) are internally connected.

Connect the IC0 pins directly to V<sub>ss</sub>.

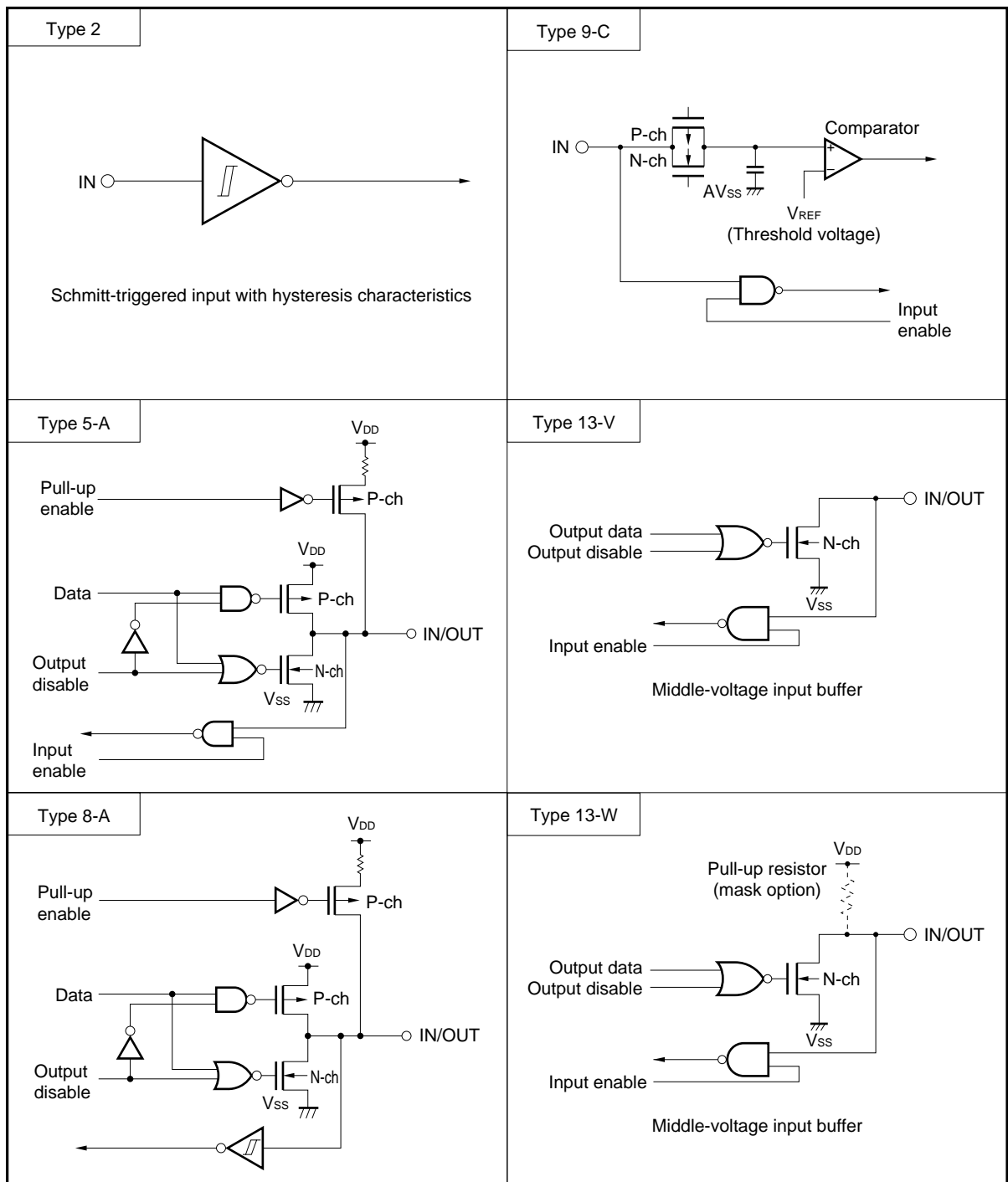
### 3.3 Pin Input/Output Circuits and Recommended Connection of Unused Pins

The input/output circuit type for each pin and the recommended connection of pins are shown in Table 3-1. For the input/output circuit configuration of each type, refer to **Figure 3-1**.

**Table 3-1. Types of Pin Input/Output Circuits and Recommended Connection of Unused Pins**

Pin Name	Input/Output Circuit Type	Input/Output	Recommended Connection of Unused Pins
P00 to P03	5-A	Input/output	Input: Independently connect these pins to V <sub>DD</sub> or V <sub>SS</sub> via a resistor. Output: Leave open
P10, P11			
P20/ $\overline{\text{SCK20}}$ /ASCK20	8-A		
P21/SO20/TxD20			
P22/SI20/RxD20			
P23/INTP0/CPT20/ $\overline{\text{SS20}}$			Input: Independently connect these pins to V <sub>SS</sub> via a resistor. Output: Leave open
P24/INTP1/TO80/TO20			
P25/INTP2/TI80			
P50 to P53 (Mask ROM version)	13-W	Input/output	Input: Independently connect these pins to V <sub>DD</sub> via a resistor. Output: Leave open
P50 to P53 ( $\mu$ PD78F9116A, 78F9136A)	13-V		
P60/ANI0 to P63/ANI3	9-C	Input	Connect to V <sub>DD</sub> or V <sub>SS</sub> .
AV <sub>DD</sub>	—	—	Connect to V <sub>DD</sub> .
AV <sub>SS</sub>			Connect to V <sub>SS</sub> .
$\overline{\text{RESET}}$	2	Input	—
IC0	—	—	Connect directly to V <sub>SS</sub> .

Figure 3-1. Pin Input/Output Circuits



**[MEMO]**

## CHAPTER 4 CPU ARCHITECTURE

### 4.1 Memory Space

The  $\mu$ PD789104A/114A/124A/134A Subseries can access 64 Kbytes of memory space. Figures 4-1 through 4-4 show the memory maps.

Figure 4-1. Memory Map ( $\mu$ PD789101A, 789111A, 789121A, 789131A)

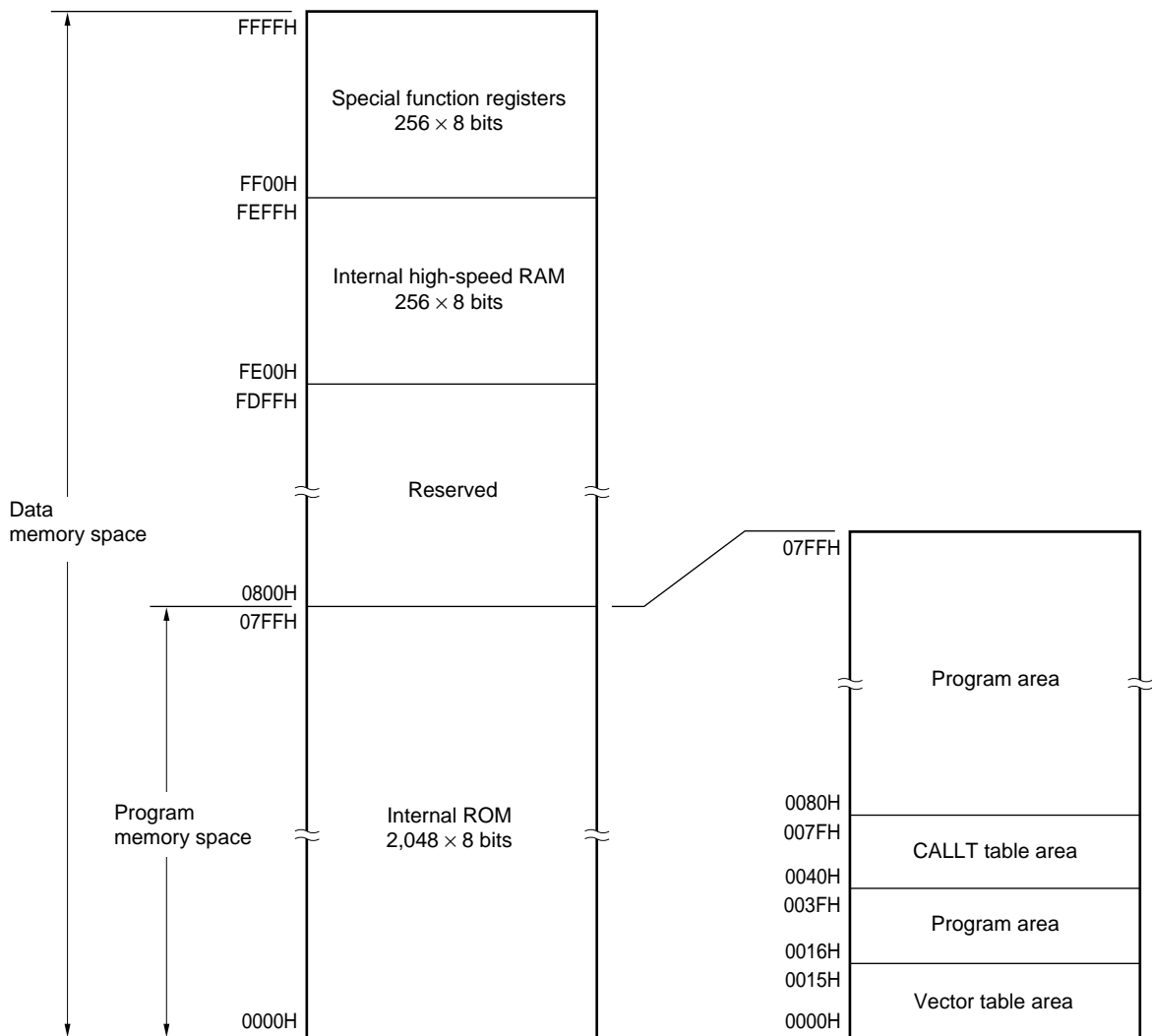


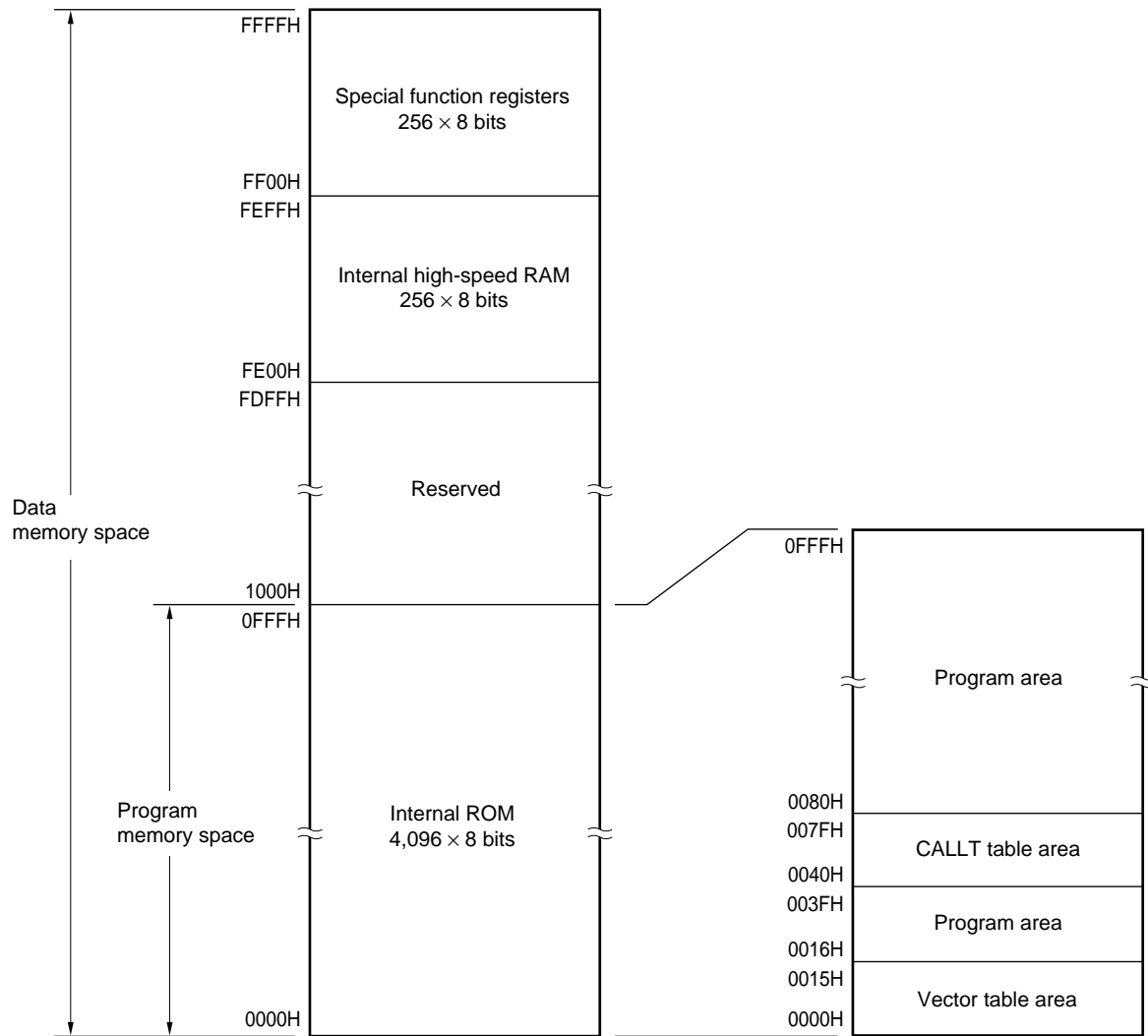
Figure 4-2. Memory Map ( $\mu$ PD789102A, 789112A, 789122A, 789132A)



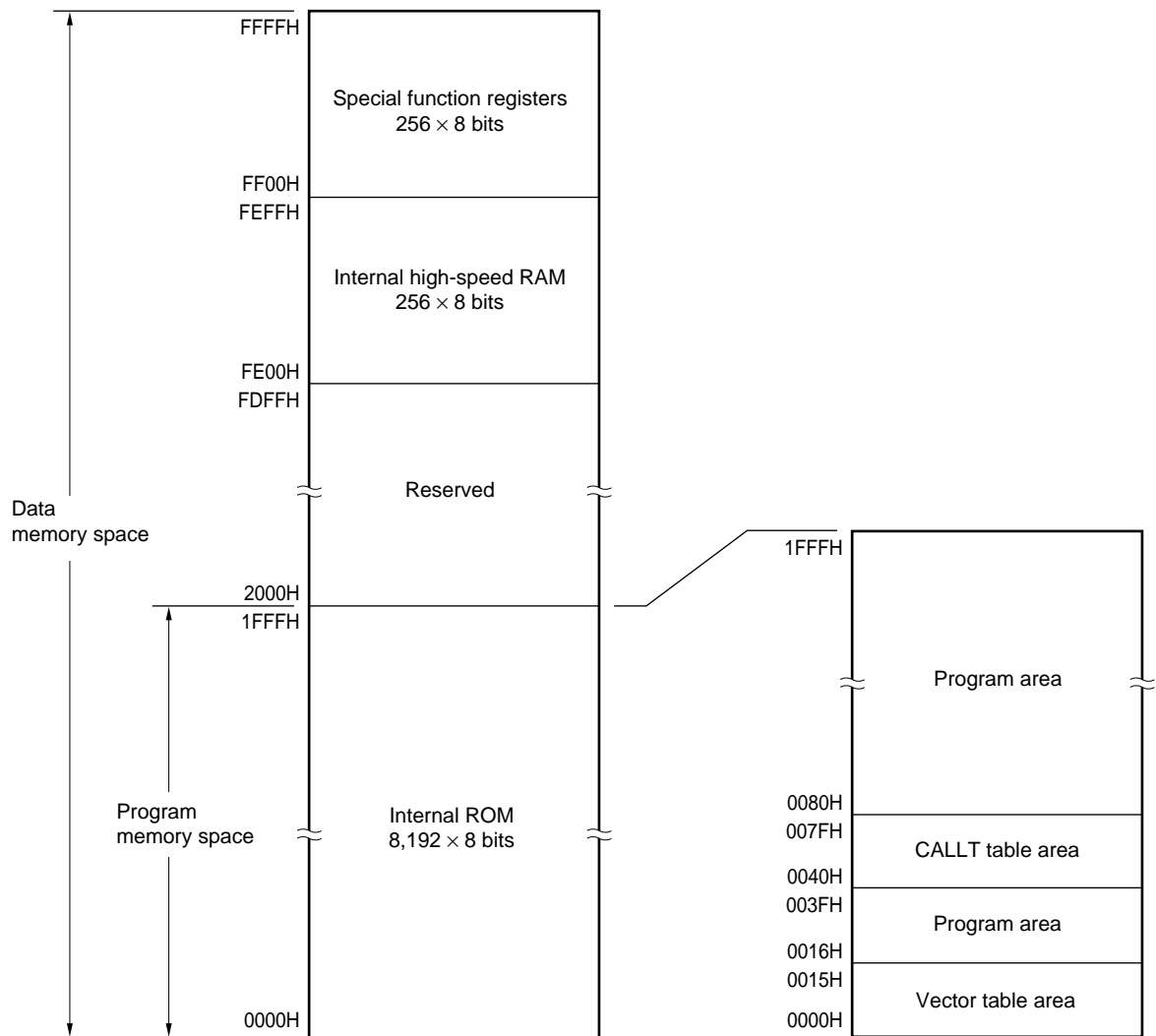
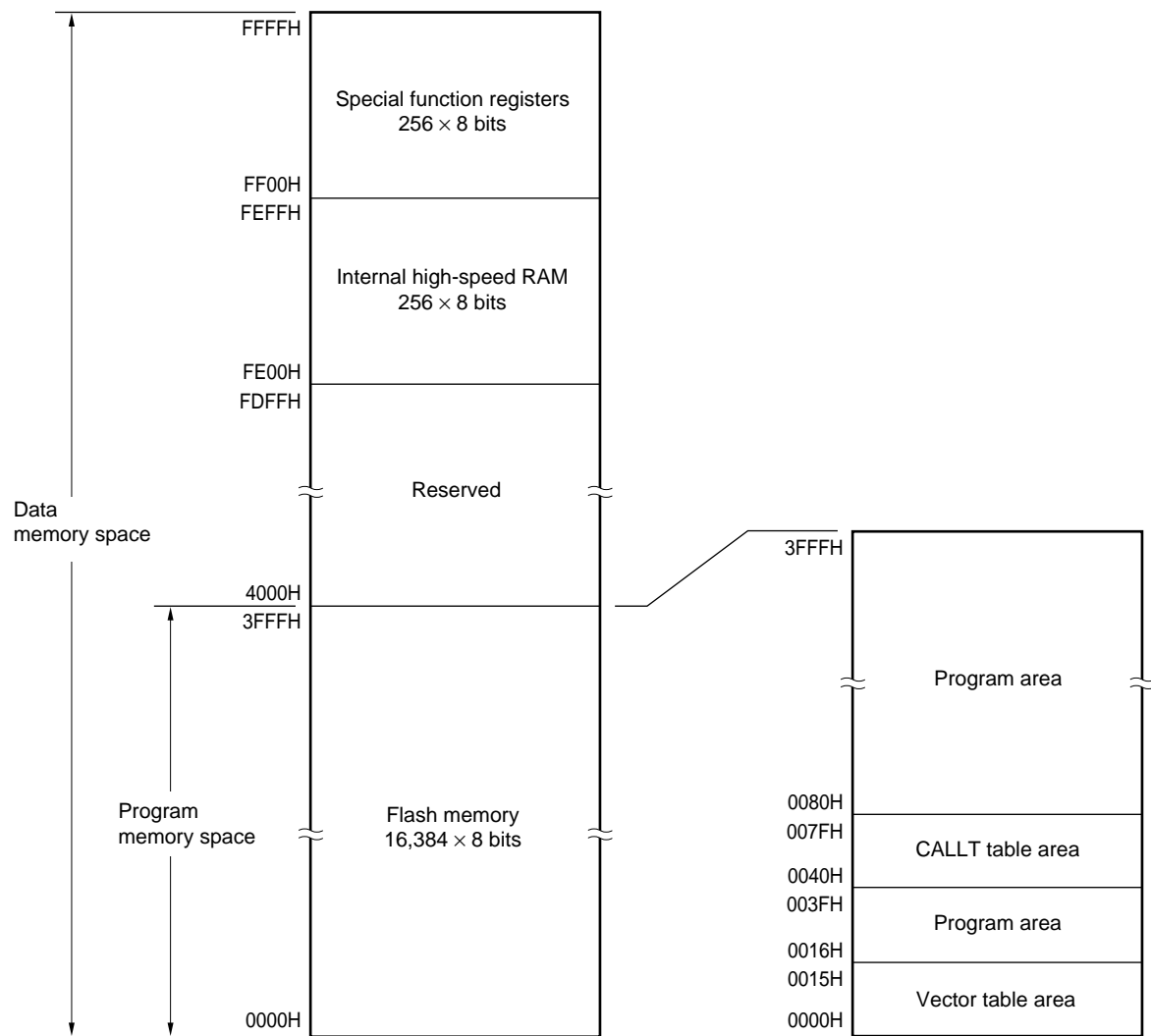
Figure 4-3. Memory Map ( $\mu$ PD789104A, 789114A, 789124A, 789134A)

Figure 4-4. Memory Map ( $\mu$ PD78F9116A, 78F9136A)

#### 4.1.1 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  $\mu$ PD789104A/114A/124A/134A Subseries provides the following internal ROMs (or flash memory) containing the following capacities.

**Table 4-1. Internal ROM Capacity**

Part Number	Internal ROM	
	Structure	Capacity
$\mu$ PD789101A, 789111A, 789121A, 789131A	Mask ROM	$2,048 \times 8$ bits
$\mu$ PD789102A, 789112A, 789122A, 789132A		$4,096 \times 8$ bits
$\mu$ PD789104A, 789114A, 789124A, 789134A		$8,192 \times 8$ bits
$\mu$ PD78F9116A, 78F9136A	Flash memory	$16,384 \times 8$ bits

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

##### (1) Vector table area

A 22-byte area of addresses 0000H to 0015H is reserved as a vector table area. This area stores program start addresses to be used when branching by the  $\overline{\text{RESET}}$  input or an interrupt request generation. Of a 16-bit program address, the lower 8 bits are stored in an even address, and the higher 8 bits are stored in an odd address.

**Table 4-2. Vector Table**

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	$\overline{\text{RESET}}$ input	000CH	INTSR20/INTCSI20
0004H	INTWDT	000EH	INTST20
0006H	INTP0	0010H	INTTM80
0008H	INTP1	0012H	INTTM20
000AH	INTP2	0014H	INTAD0

##### (2) CALLT instruction table area

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

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

The  $\mu$ PD789104A/114A/124A/134A Subseries provides a 256-byte internal high-speed RAM.

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

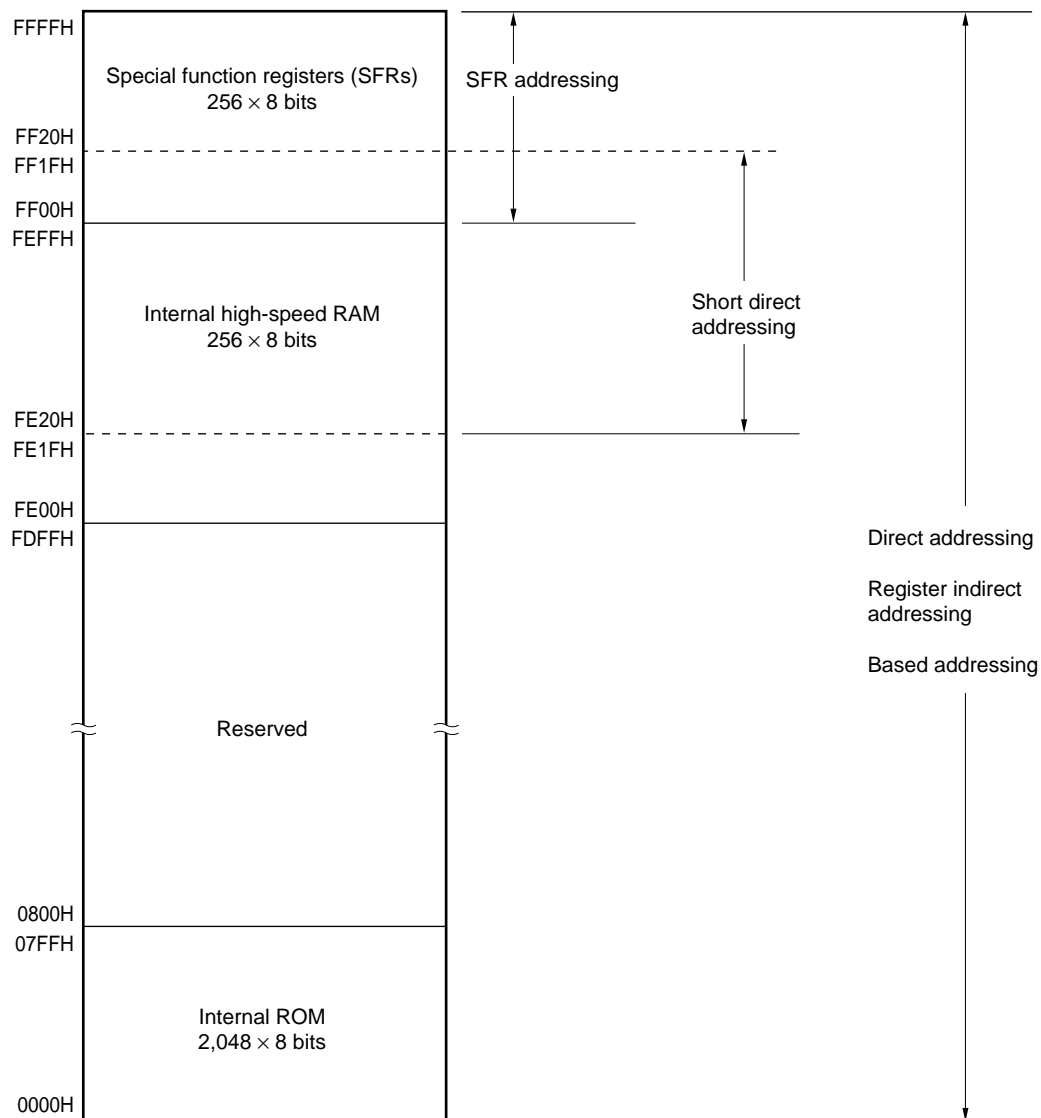
#### 4.1.3 Special function register (SFR) area

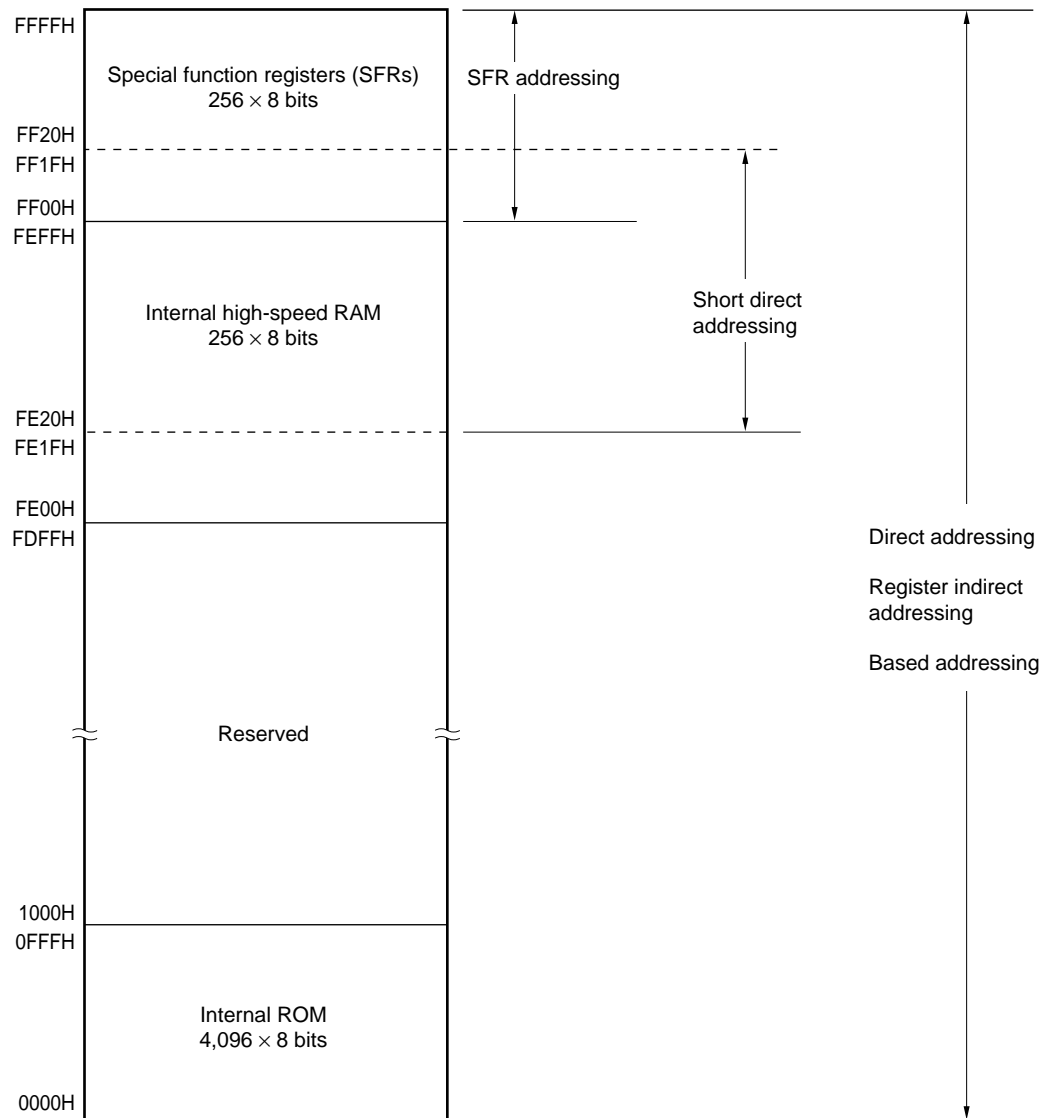
Special function registers (SFRs) of on-chip peripheral hardware are allocated to the area of FF00H to FFFFH (refer to Table 4-3).

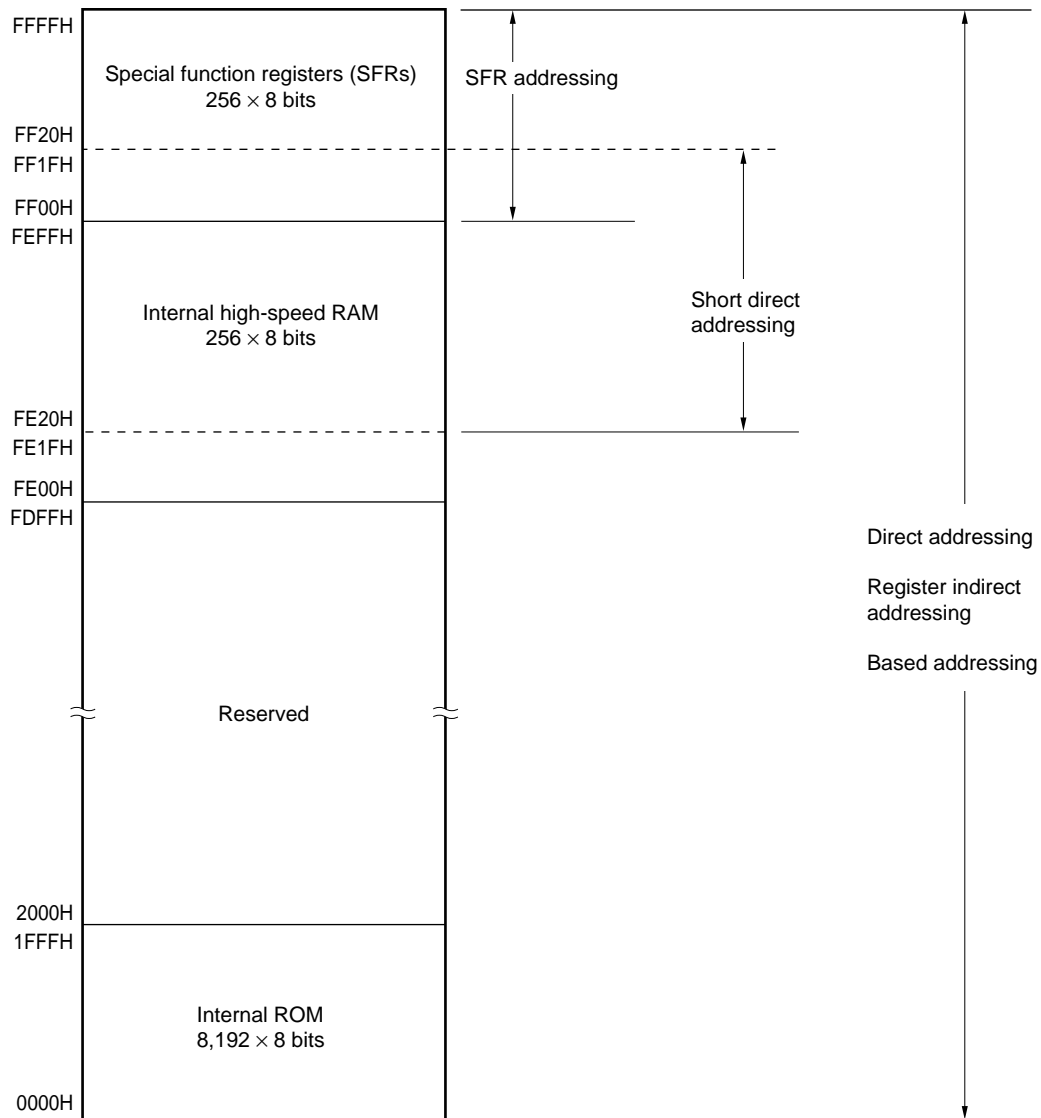
#### 4.1.4 Data memory addressing

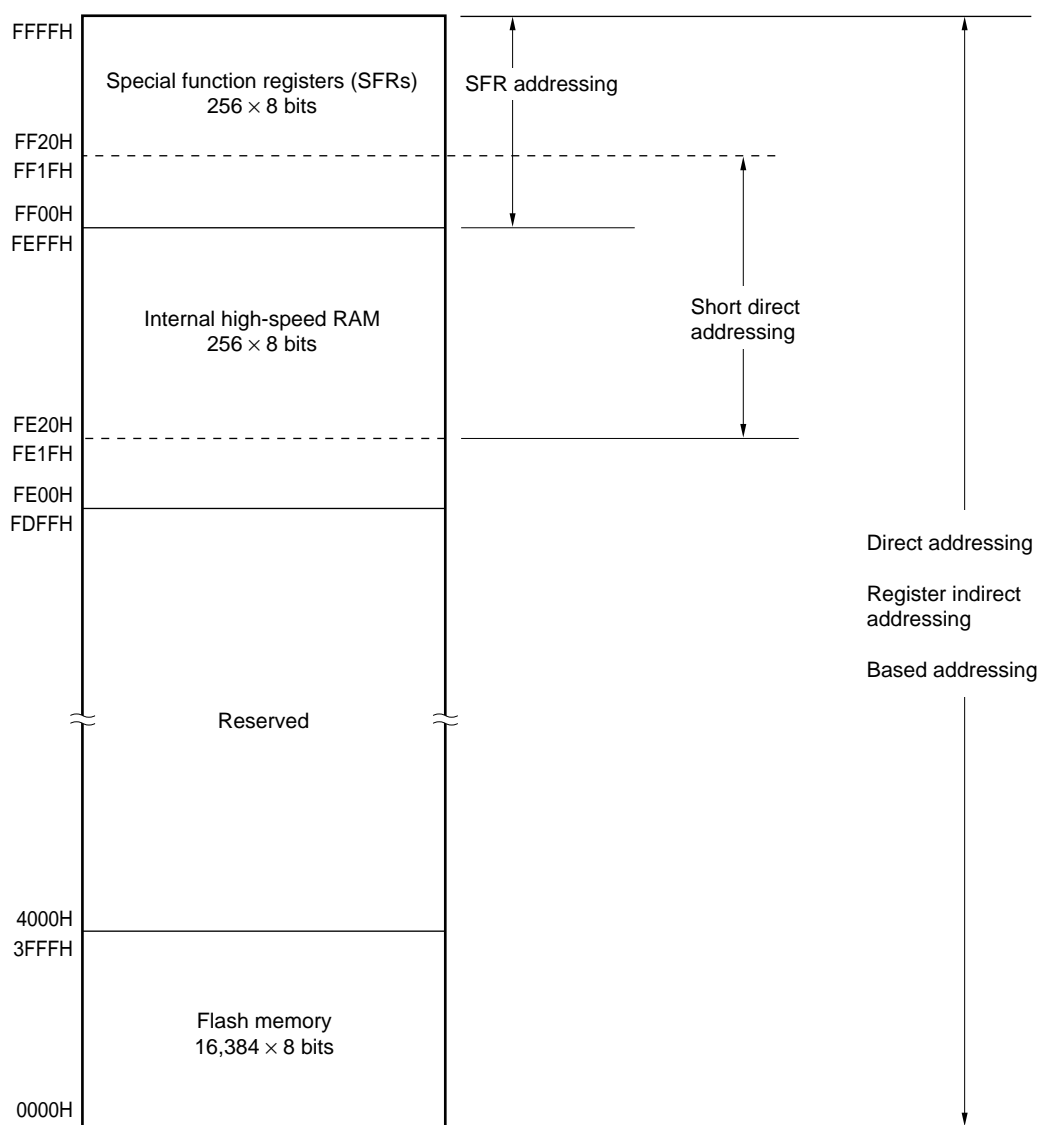
The  $\mu$ PD789104A/114A/124A/134A Subseries provides a variety of addressing modes which take account of memory manipulability, etc. Especially at addresses corresponding to data memory area (FE00H to FFFFH), particular addressing modes are possible to meet the functions of the special function registers (SFRs) and general registers. Figures 4-5 through 4-8 show the data memory addressing modes.

**Figure 4-5. Data Memory Addressing ( $\mu$ PD789101A, 789111A, 789121A, 789131A)**



**Figure 4-6. Data Memory Addressing ( $\mu$ PD789102A, 789112A, 789122A, 789132A)**

**Figure 4-7. Data Memory Addressing ( $\mu$ PD789104A, 789114A, 789124A, 789134A)**

**Figure 4-8. Data Memory Addressing ( $\mu$ PD78F9116A, 78F9136A)**

## 4.2 Processor Registers

The  $\mu$ PD789104A/114A/124A/134A Subseries provides the following on-chip processor registers:

### 4.2.1 Control registers

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

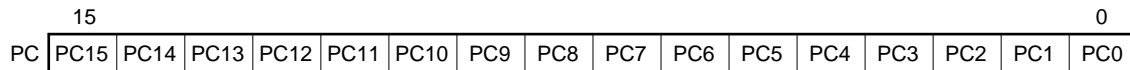
#### (1) Program counter (PC)

The program counter is a 16-bit register which holds the address information of the next program to be executed.

In normal operation, the PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data or register contents is set.

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

**Figure 4-9. Program Counter Configuration**

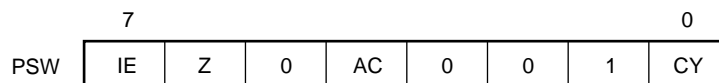


#### (2) Program status word (PSW)

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 4-10. Program Status Word Configuration**





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

This flag controls interrupt request acknowledge operations of CPU.

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

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

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

**(b) Zero flag (Z)**

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

**(c) 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 (0) in all other cases.

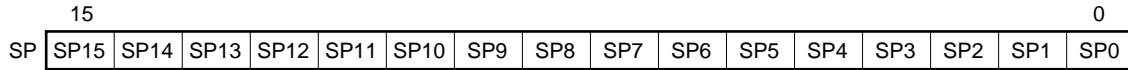
**(d) Carry flag (CY)**

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

### (3) Stack pointer (SP)

This is a 16-bit register to hold the start address of the memory stack area. Only the internal high-speed RAM area can be set as the stack area.

**Figure 4-11. Stack Pointer Configuration**

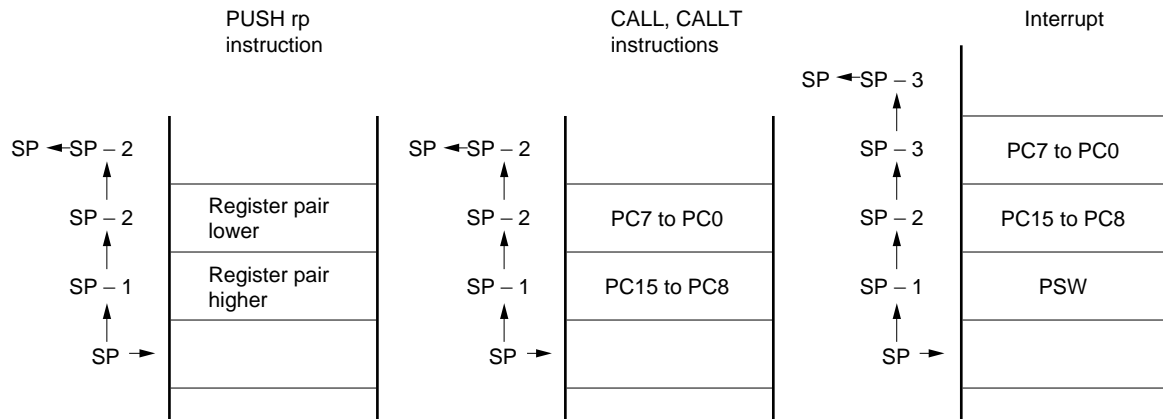


The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.

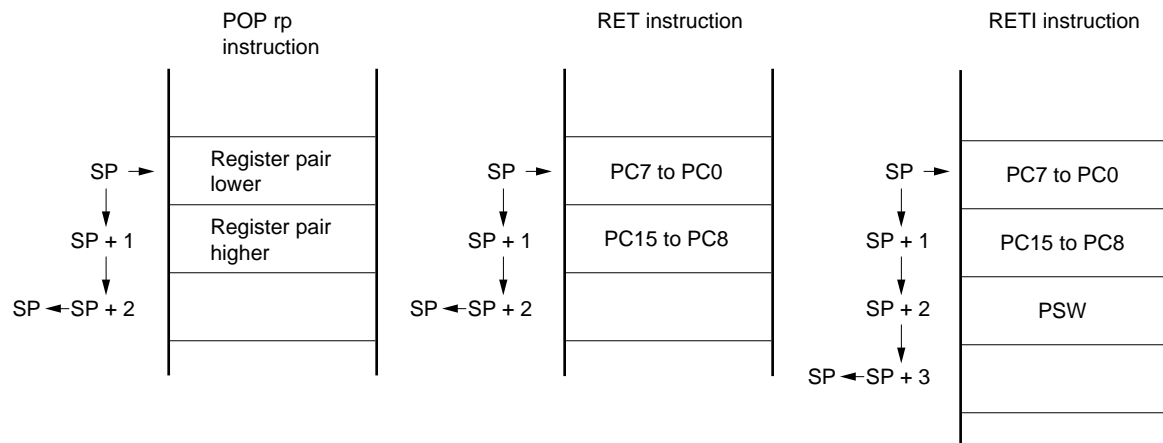
Each stack operation saves/restores data as shown in Figures 4-12 and 4-13.

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

**Figure 4-12. Data to be Saved to Stack Memory**



**Figure 4-13. Data to be Restored from Stack Memory**



### 4.2.2 General registers

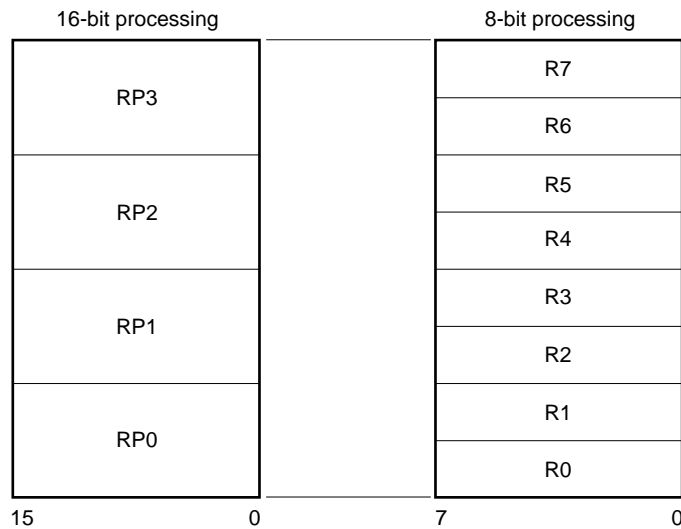
The general registers consist of eight 8-bit registers (X, A, C, B, E, D, L, and H).

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

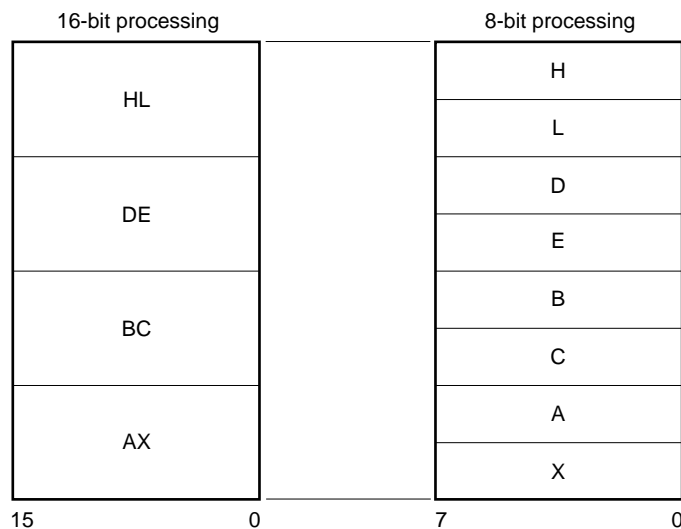
They can be described in terms of functional names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

**Figure 4-14. General Register Configuration**

**(a) Absolute names**



**(b) Functional names**



### 4.2.3 Special function registers (SFRs)

Unlike general registers, special function registers have their own functions and are allocated to the 256-byte area FF00H to FFFFH.

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

Each bit unit for manipulation can be specified as follows.

- 1-bit manipulation

A symbol reserved by assembler is described as the operand (sfr.bit) of a 1-bit manipulation instruction. This manipulation can also be specified with an address.

- 8-bit manipulation

A symbol reserved by assembler is described as the operand (sfr) of an 8-bit manipulation instruction. This manipulation can also be specified with an address.

- 16-bit manipulation

A symbol reserved by assembler is described as the operand of a 16-bit manipulation instruction. When specifying an address, describe an even address.

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

- Symbol

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

- R/W

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

R/W: Read/write

R: Read only

W: Write only

- Bit units for manipulation

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

- After reset

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

Table 4-3. Special Function Register List (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			√	√	—	
FF05H	Port 5	P5			√	√	—	
FF06H	Port 6	P6		R	√	√	—	Undefined
FF10H	16-bit multiplication result storage register 0	MUL0L	MUL0		—	√ <b>Note 1</b>	√ <b>Note 2</b>	
FF11H		MUL0H						
FF14H	A/D conversion result register <b>Note 3</b>	ADCR0			—	√	√ <b>Note 2</b>	
FF15H								
FF16H	16-bit compare register 20	CR20L	CR20	W	—	√ <b>Note 1</b>	√ <b>Note 2</b>	FFFFH
FF17H		CR20H						
FF18H	16-bit timer counter 20	TM20L	TM20	R	—	√ <b>Note 1</b>	√ <b>Note 2</b>	0000H
FF19H		TM20H						
FF1AH	16-bit capture register 20	TCP20L	TCP20		—	√ <b>Note 1</b>	√ <b>Note 2</b>	Undefined
FF1BH		TCP20H						
FF20H	Port mode register 0	PM0		R/W	√	√	—	FFH
FF21H	Port mode register 1	PM1			√	√	—	
FF22H	Port mode register 2	PM2			√	√	—	
FF25H	Port mode register 5	PM5			√	√	—	
FF32H	Pull-up resistor option register B2	PUB2			√	√	—	00H
FF42H	Time clock select register 2	TCL2			—	√	—	
FF48H	16-bit timer mode control register 20	TMC20			√	√	—	
FF50H	8-bit compare register 80	CR80		W	—	√	—	Undefined
FF51H	8-bit timer counter 80	TM80		R	—	√	—	
FF53H	8-bit timer mode control register 80	TMC80		R/W	√	√	—	

- Notes**
1. Although these registers are usually accessed in 16-bit units, they can also be accessed in 8-bit units. Access these registers in 8-bit units by means of direct addressing.
  2. These registers can be accessed in 16-bit units only by means of short direct addressing.
  3. When this register is used as an 8-bit A/D converter ( $\mu$ PD789104A and 789124A Subseries), it can be accessed only in 8-bit units. At this time, the register address is FF15H. When this register is used as a 10-bit A/D converter ( $\mu$ PD789114A and 789134A Subseries), it can be accessed only in 16-bit units. When using the  $\mu$ PD78F9116A as the flash memory version of the  $\mu$ PD789101A, 789102A, or 789104A, or when using the  $\mu$ PD78F9136A as the flash memory version of the  $\mu$ PD789121A, 789122A, or 789124A, this register can be accessed in 8-bit units. However, only the object file assembled with the  $\mu$ PD789101A, 789102A, or 789104A, or object file assembled with the  $\mu$ PD789121A, 789122A, or 789124A can be used.

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

Address	Special Function Register (SFR) Name	Symbol		R/W	Bit Units for Manipulation			After Reset
					1 bit	8 bits	16 bits	
FF70H	Asynchronous serial interface mode register 20	ASIM20		R/W	√	√	—	00H
FF71H	Asynchronous serial interface status register 20	ASIS20		R	√	√	—	
FF72H	Serial operating mode register 20	CSIM20		R/W	√	√	—	
FF73H	Baud rate generator control register 20	BRGC20			—	√	—	
FF74H	Transmit shift register 20	TXS20	SIO20	W	—	√	—	FFH
	Receive buffer register 20	RXB20		R	—	√	—	Undefined
FF80H	A/D converter mode register 0	ADM0		R/W	√	√	—	00H
FF84H	Analog input channel specification register 0	ADS0			√	√	—	
FFD0H	Multiplication data register A0	MRA0		W	√	√	—	Undefined
FFD1H	Multiplication data register B0	MRB0			√	√	—	
FFD2H	Multiplier control register 0	MULC0		R/W	√	√	—	00H
FFE0H	Interrupt request flag register 0	IF0			√	√	—	
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
FFF7H	Pull-up resistor option register 0	PU0			√	√	—	
FFF9H	Watchdog timer mode register	WDTM			√	√	—	
FFFAH	Oscillation stabilization time select register <b>Note</b>	OSTS			—	√	—	04H
FFFBH	Processor clock control register	PCC				√	√	—

**Note**  $\mu$ PD789104A, 789114A Subseries only

### 4.3 Instruction Address Addressing

An instruction address is determined by the program counter (PC) contents. The PC contents are normally incremented (+1 for each byte) automatically according to the number of bytes of an instruction to be fetched each time another instruction is executed. When a branch instruction is executed, the branch destination information is set to the PC and branched by the following addressing (For details of each instruction, refer to **78K/0S Series User's Manual Instruction (U11047E)**).

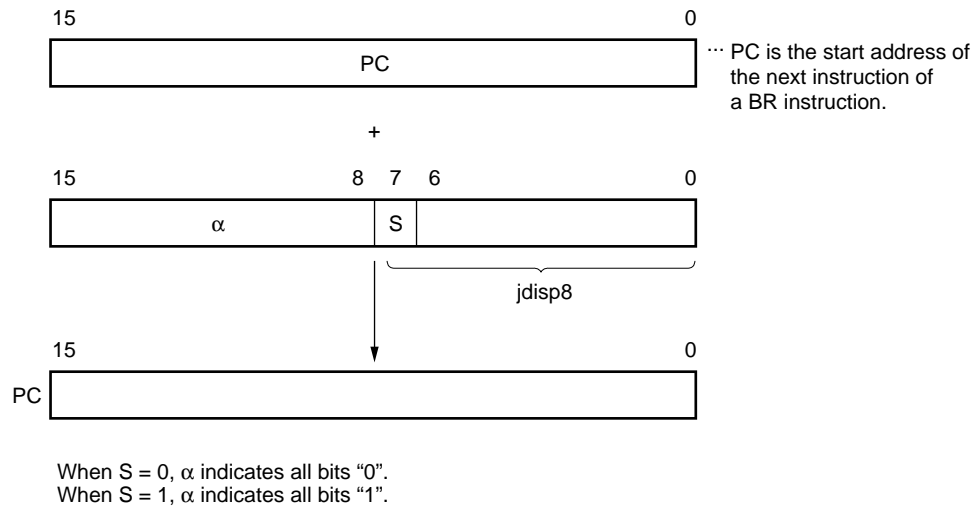
#### 4.3.1 Relative addressing

##### [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]



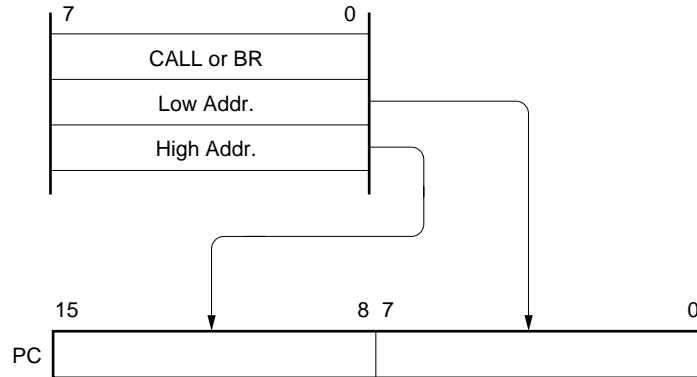
### 4.3.2 Immediate addressing

**[Function]**

Immediate data in the instruction word is transferred to the program counter (PC) and branched. This function is carried out when the “CALL !addr16 and BR !addr16” instructions are executed. CALL !addr16 and BR !addr16 instructions can branch to all the memory spaces.

**[Illustration]**

In case of CALL !addr16, BR !addr16 instruction





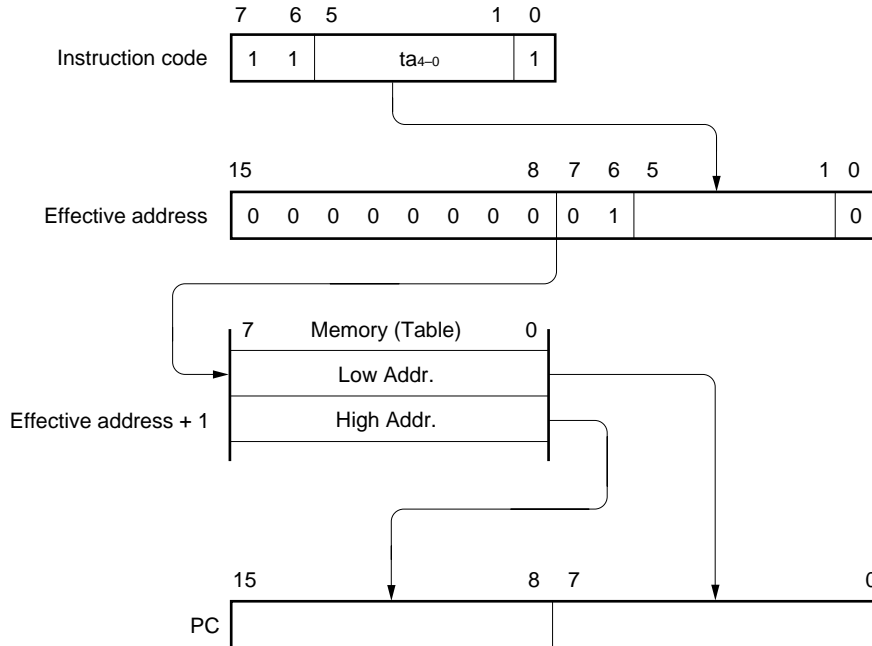
### 4.3.3 Table indirect addressing

#### [Function]

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

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

#### [Illustration]



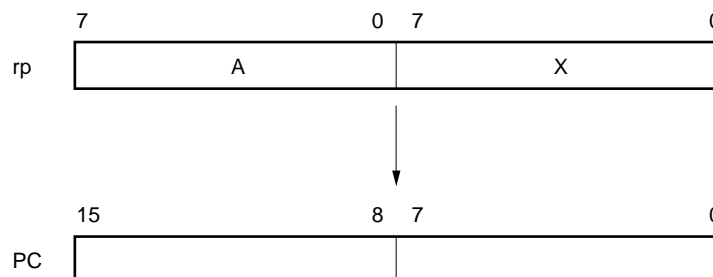
### 4.3.4 Register addressing

#### [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]



## 4.4 Operand Address Addressing

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

### 4.4.1 Direct addressing

#### [Function]

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

#### [Operand format]

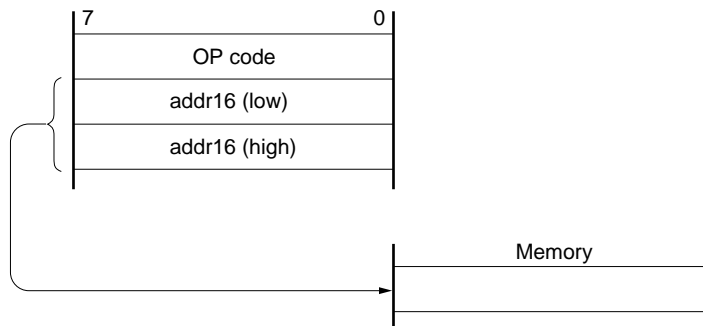
Identifier	Description
addr16	Label or 16-bit immediate data

#### [Description example]

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

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

#### [Illustration]



#### 4.4.2 Short direct addressing

##### [Function]

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

The SFR area (FF00H to FF1FH) where short direct addressing is applied is a part of all SFR areas. In this area, ports which are frequently accessed in a program and a compare register of the timer/event counter are mapped, and these SFRs can be manipulated with a small number of bytes and clocks.

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

##### [Operand format]

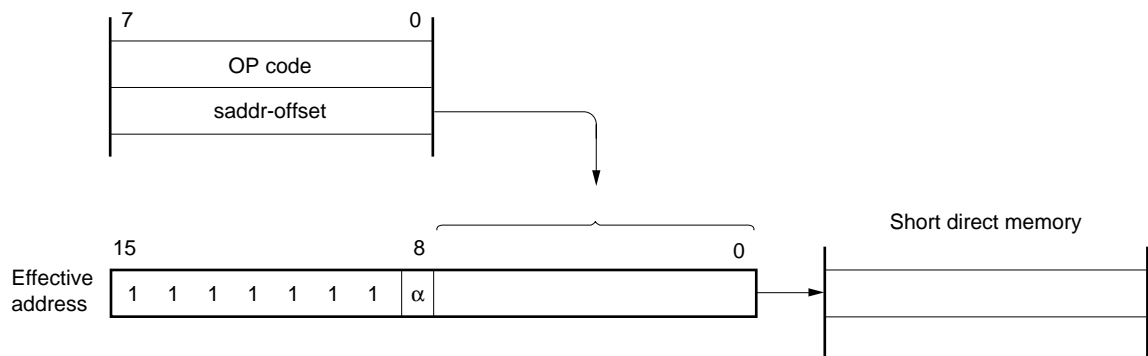
Identifier	Description
saddr	Label or FE20H to FF1FH immediate data
saddrp	Label or FE20H to FF1FH immediate data (even address only)

##### [Description example]

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

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

##### [Illustration]



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

### 4.4.3 Special function register (SFR) addressing

#### [Function]

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

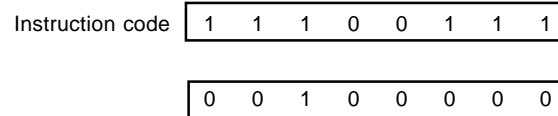
This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, SFRs mapped at FF00H to FF1FH can be accessed with short direct addressing.

#### [Operand format]

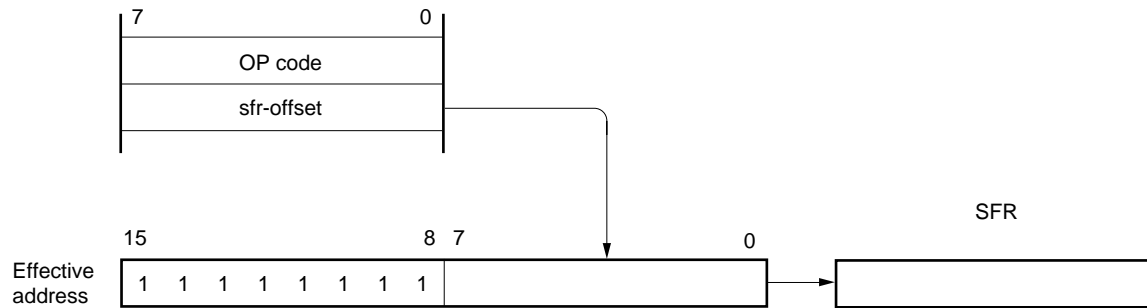
Identifier	Description
sfr	Special function register name

#### [Description example]

MOV PM0, A; When selecting PM0 for sfr



#### [Illustration]



#### 4.4.4 Register addressing

##### [Function]

General registers are accessed as operands. The general register to be accessed is specified with the register specify code and functional name in the instruction code.

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

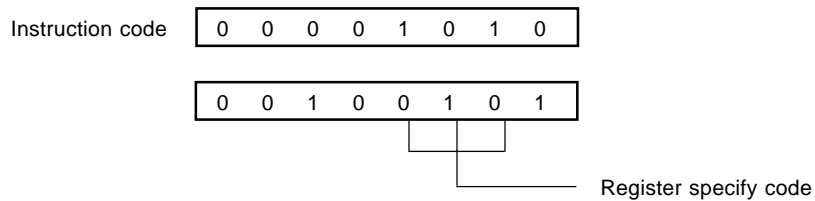
##### [Operand format]

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

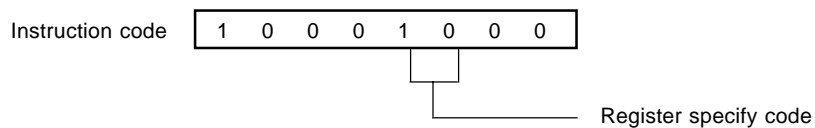
'r' and 'rp' can be described with absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

##### [Description example]

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



INCW DE; When selecting the DE register pair for rp



4.4.5 Register indirect addressing

[Function]

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

[Operand format]

Identifier	Description
—	[DE], [HL]

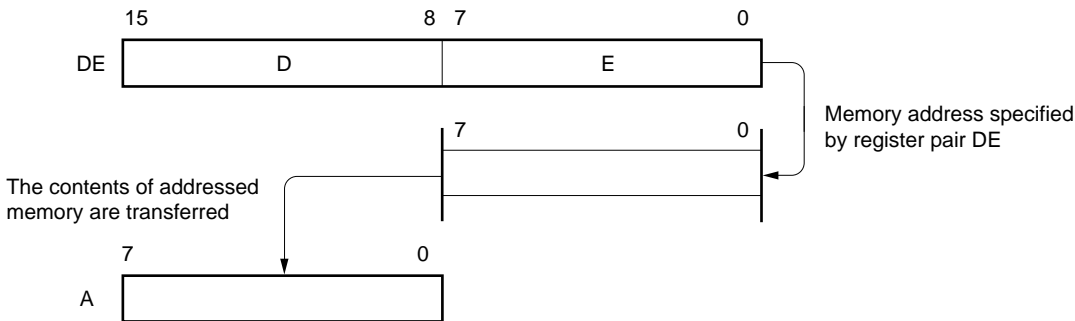
[Description example]

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

Instruction code 

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

[Illustration]



#### 4.4.6 Based addressing

##### [Function]

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

##### [Operand format]

Identifier	Description
—	[HL+byte]

##### [Description example]

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

Instruction code

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

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

#### 4.4.7 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/reset upon generation of an interrupt request.

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

##### [Description example]

In the case of PUSH DE

Instruction code

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

**[MEMO]**



## CHAPTER 5 PORT FUNCTIONS

### 5.1 Functions of Ports

The  $\mu$ PD789104A/114A/124A/134A Subseries provides the ports shown in Figure 5-1, enabling various methods of control.

Numerous other functions are provided that can be used in addition to the digital I/O port function. For more information on these additional functions, refer to **CHAPTER 3 PIN FUNCTIONS**.

**Figure 5-1. Port Types**

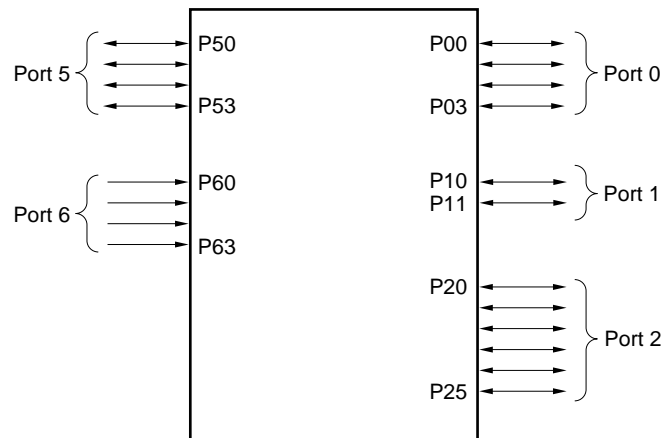


Table 5-1. Port Functions

Pin Name	Input/Output	Function	After Reset	Alternate Function
P00 to P03	Input/output	Port 0 4-bit I/O port Input/output can be specified in 1-bit units When used as input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	—
P10, P11	Input/output	Port 1 2-bit I/O port Input/output can be specified in 1-bit units When used as input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	—
P20	Input/output	Port 2 6-bit I/O port Input/output can be specified in 1-bit units An on-chip pull-up resistor can be specified by means of pull-up resistor option register B2 (PUB2).	Input	ASCK20/ $\overline{\text{SCK20}}$
P21				TxD20/SO20
P22				RxD20/SI20
P23				INTP0/CPT20/ $\overline{\text{SS20}}$
P24				INTP1/TO80/TO20
P25				INTP2/TI80
P50 to P53	Input/output	Port 5 4-bit N-ch open-drain I/O port Input/output can be specified in 1-bit units An on-chip pull-up resistor can be specified for mask ROM versions by the mask option.	Input	—
P60 to 63	Input	Port 6 4-bit input-only port	Input	ANI0 to ANI3

## 5.2 Port Configuration

A port consists of the following hardware.

**Table 5-2. Configuration of Port**

Parameter	Configuration
Control register	Port mode register (PMm: m = 0 to 2, 5) Pull-up resistor option register 0 (PU0) Pull-up option register B2 (PUB2)
Port	Total: 20 (input: 7, input/output: 16)
Pull-up resistor	<ul style="list-style-type: none"> <li>Mask ROM versions Total: 16 (software control: 12, mask option specification: 4)</li> <li>Flash memory versions Total: 12 (software control only)</li> </ul>

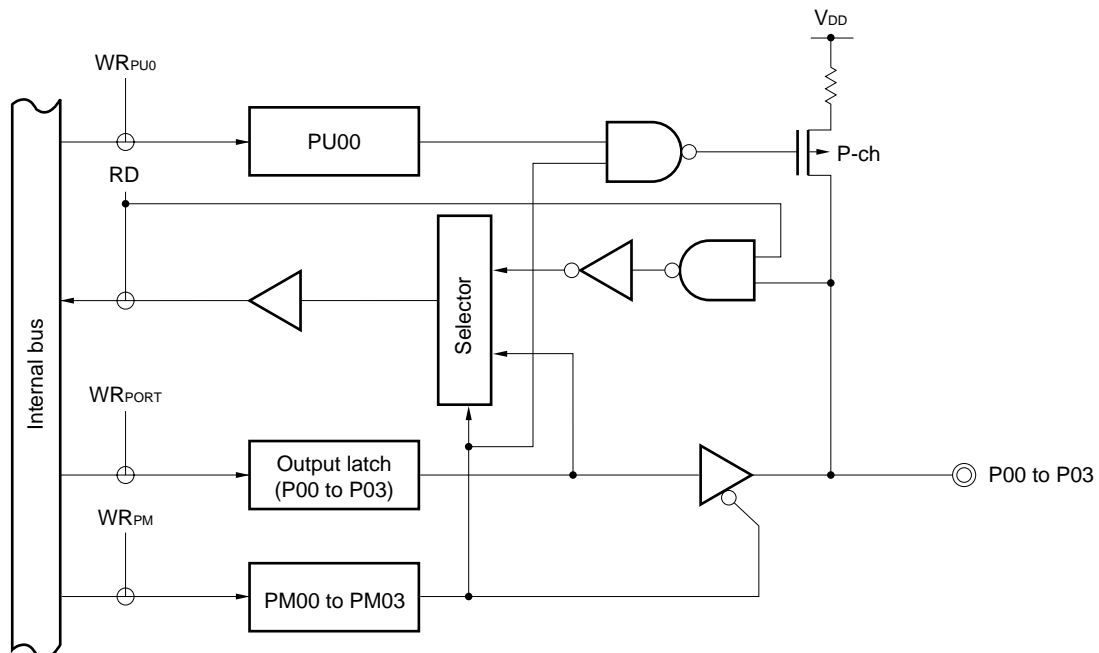
### 5.2.1 Port 0

This is a 4-bit I/O port with output latches. Port 0 can be specified as input or output mode in 1-bit units by using port mode register 0 (PM0). When pins P00 to P03 are used as input port pins, on-chip pull-up resistors can be connected in 4-bit units by using pull-up resistor option register 0 (PU0).

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

Figure 5-2 shows the block diagram of port 0.

**Figure 5-2. Block Diagram of P00 to P03**



PU0: Pull-up resistor option register 0

PM: Port mode register

RD: Port 0 read signal

WR: Port 0 write signal

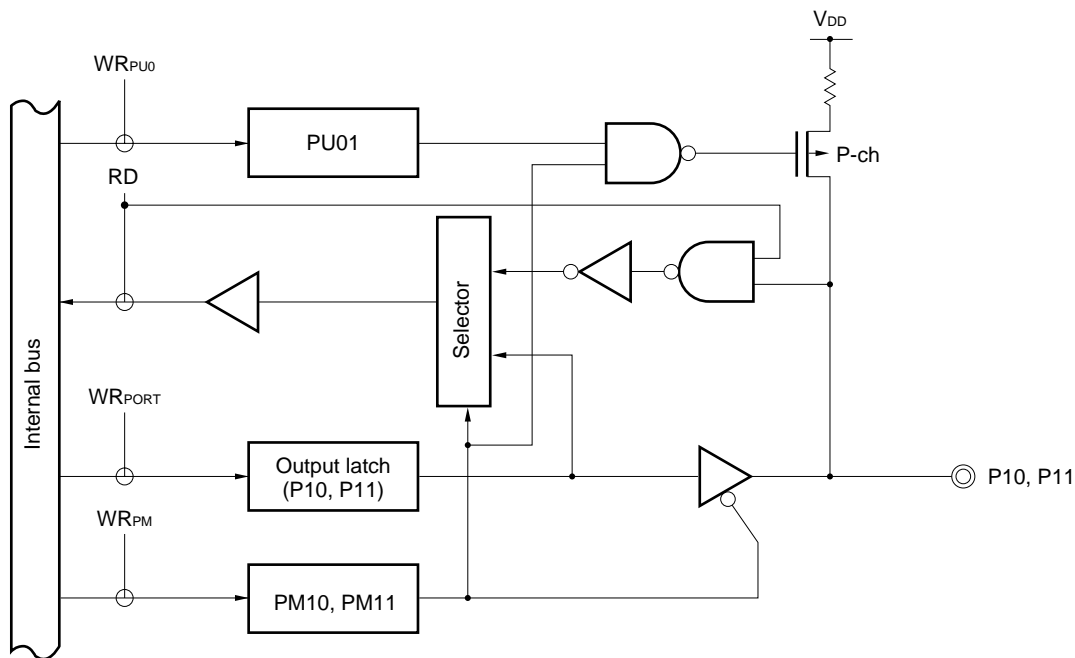
### 5.2.2 Port 1

This is a 2-bit I/O port with output latches. Port 1 can be specified as input or output mode in 1-bit units by using port mode register 1 (PM1). When pins P10 and P11 are used as input port pins, on-chip pull-up resistors can be connected in 2-bit units by using pull-up resistor option register 0 (PU0).

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

Figure 5-3 shows the block diagram of port 1.

**Figure 5-3. Block Diagram of P10 and P11**



PU0: Pull-up resistor option register 0

PM: Port mode register

RD: Port 1 read signal

WR: Port 1 write signal

### 5.2.3 Port 2

This is a 6-bit I/O port with output latches. Port 2 can be specified as input or output mode in 1-bit units by using port mode register 2 (PM2). Use of on-chip pull-up resistors can be specified for pins P20 to P25 in 1-bit units by using pull-up resistor option register B2 (PUB2).

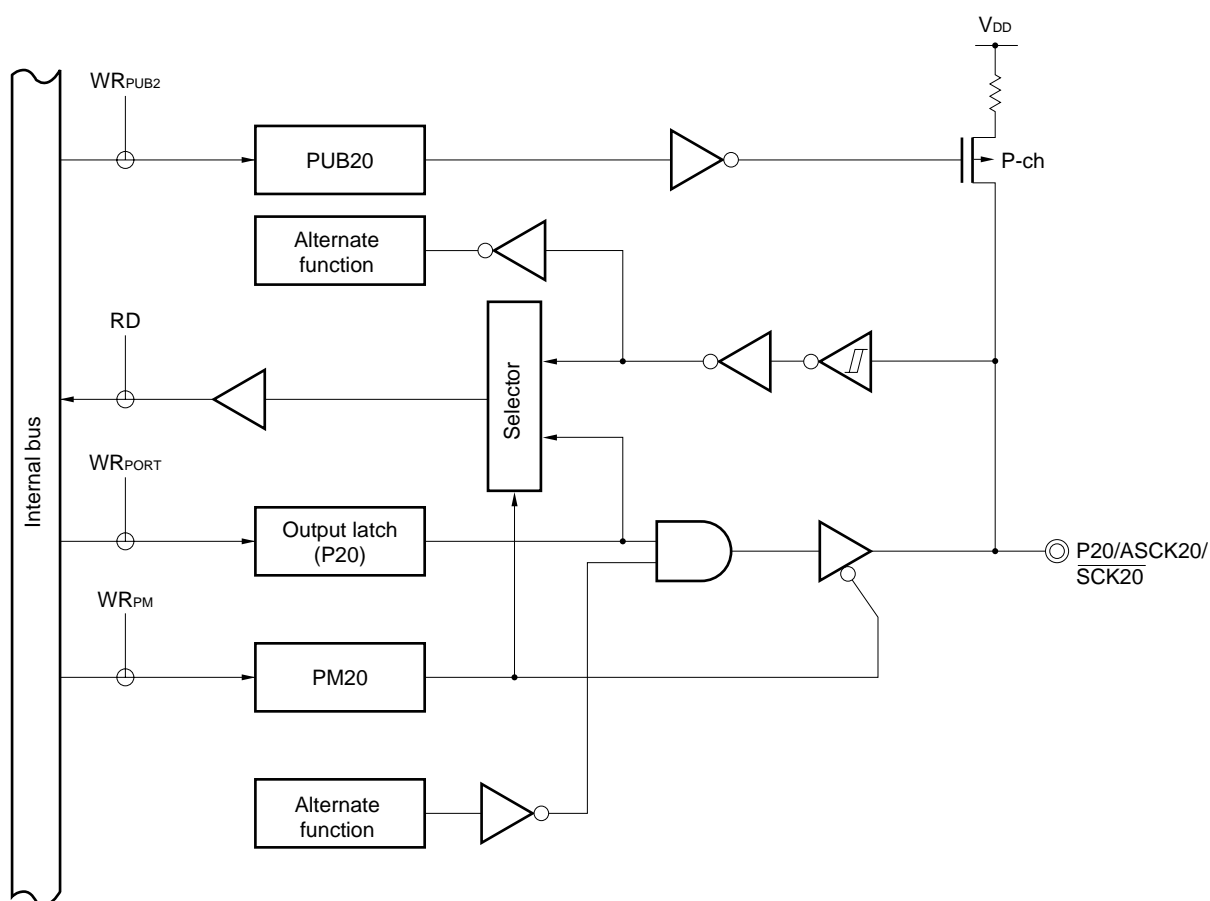
The port is also used as a serial interface I/O, clock I/O, timer I/O, and external interrupt input.

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

Figures 5-4 through 5-7 show block diagrams of port 2.

**Caution** When using the pins of port 2 as the serial interface, the I/O or output latch must be set according to the function to be used. For how to set the latches, see Table 13-2 Serial Interface 20 Operating Mode Settings.

Figure 5-4. Block Diagram of P20



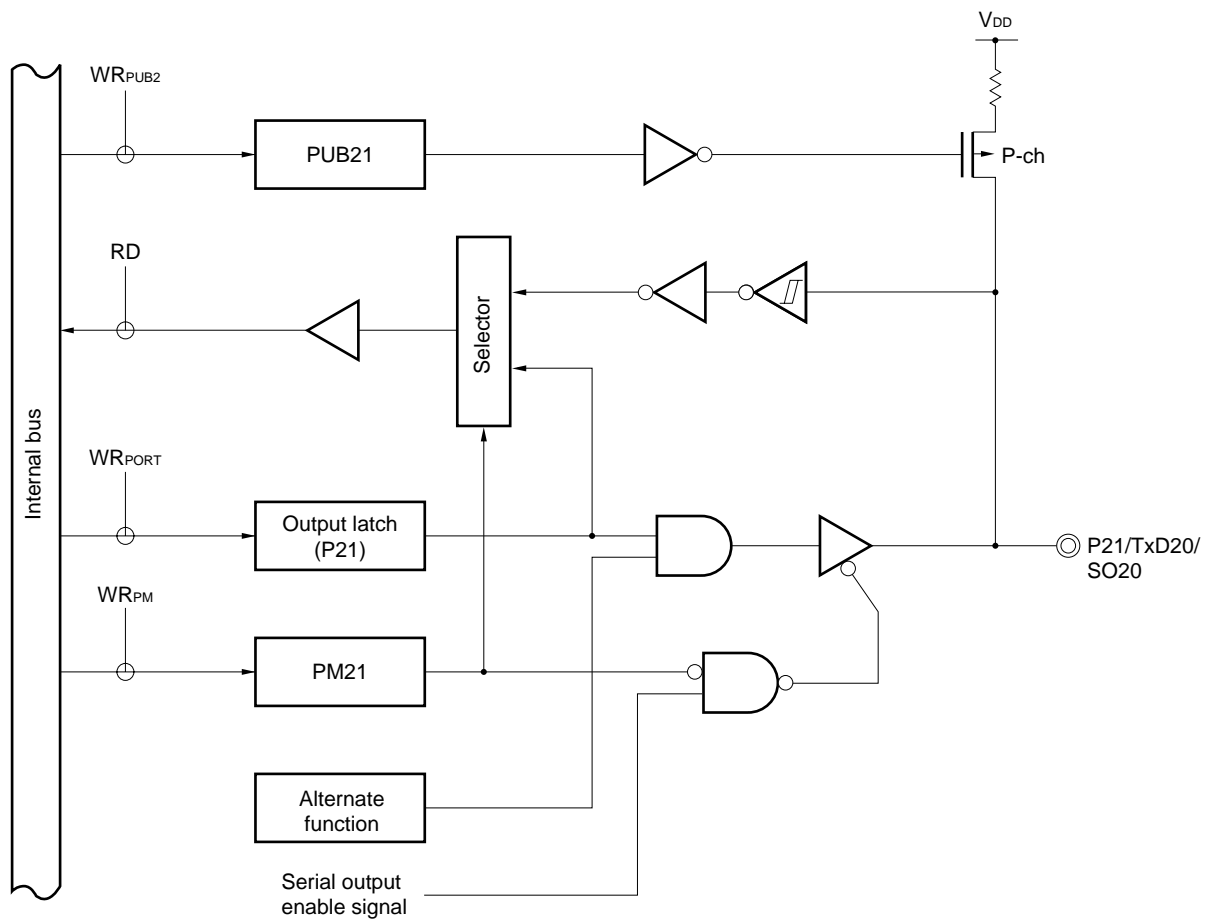
PUB2: Pull-up resistor option register B2

PM: Port mode register

RD: Port 2 read signal

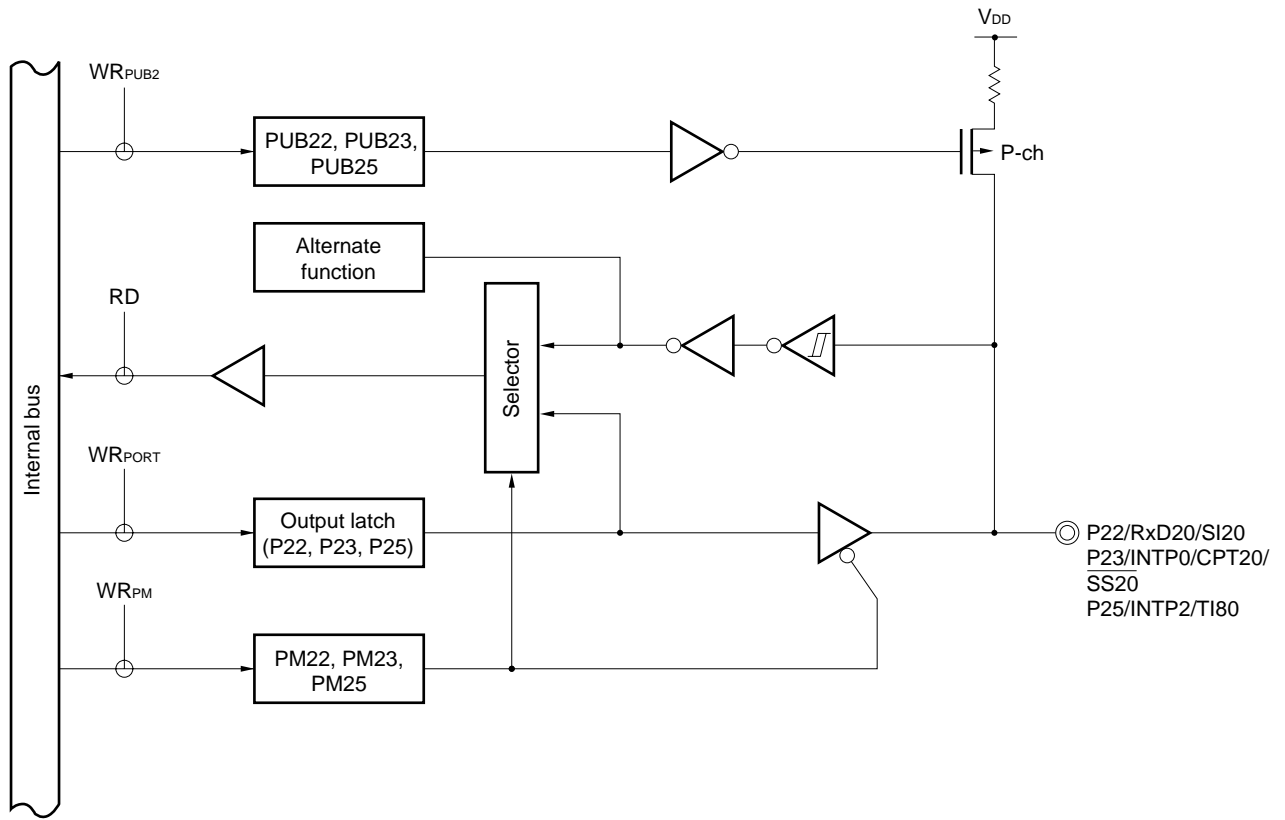
WR: Port 2 write signal

Figure 5-5. Block Diagram of P21



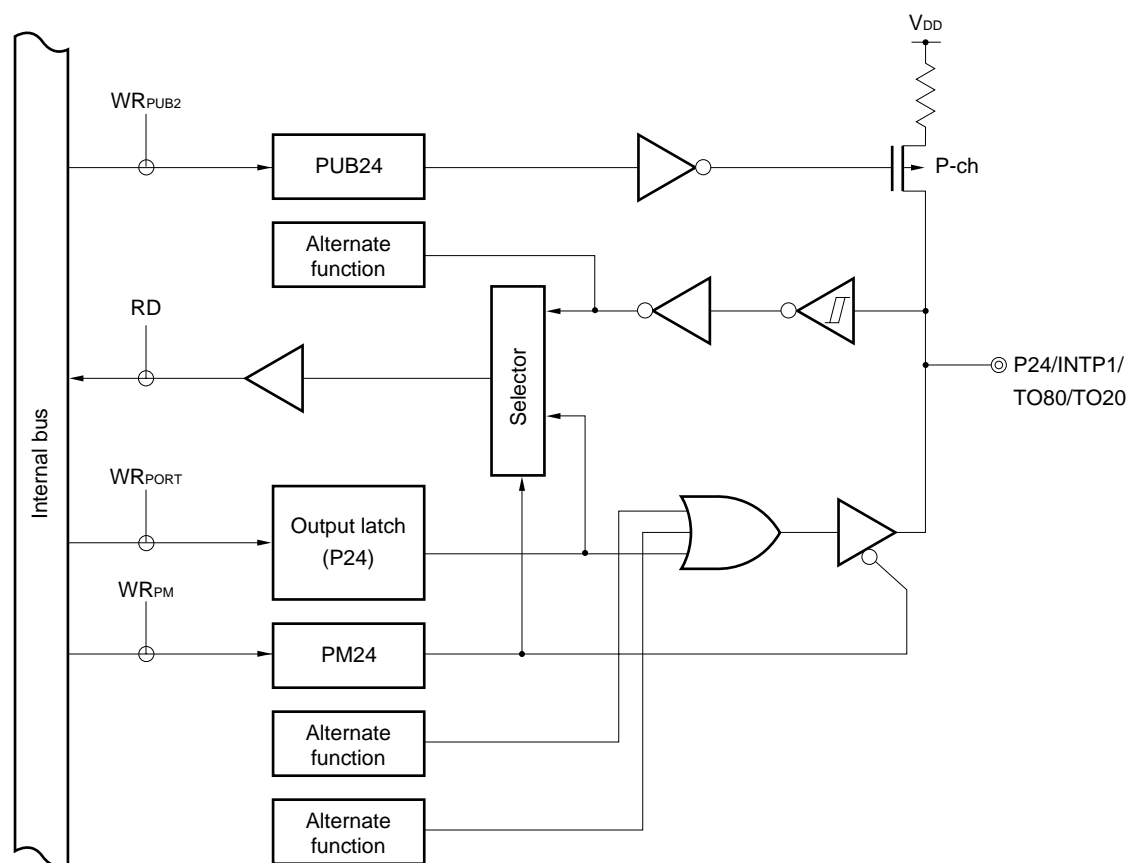
**PUB2:** Pull-up resistor option register B2  
**PM:** Port mode register  
**RD:** Port 2 read signal  
**WR:** Port 2 write signal

Figure 5-6. Block Diagram of P22, P23, and P25



PUB2: Pull-up resistor option register B2  
 PM: Port mode register  
 RD: Port 2 read signal  
 WR: Port 2 write signal

Figure 5-7. Block Diagram of P24



PUB2: Pull-up resistor option register B2

PM: Port mode register

RD: Port 2 read signal

WR: Port 2 write signal



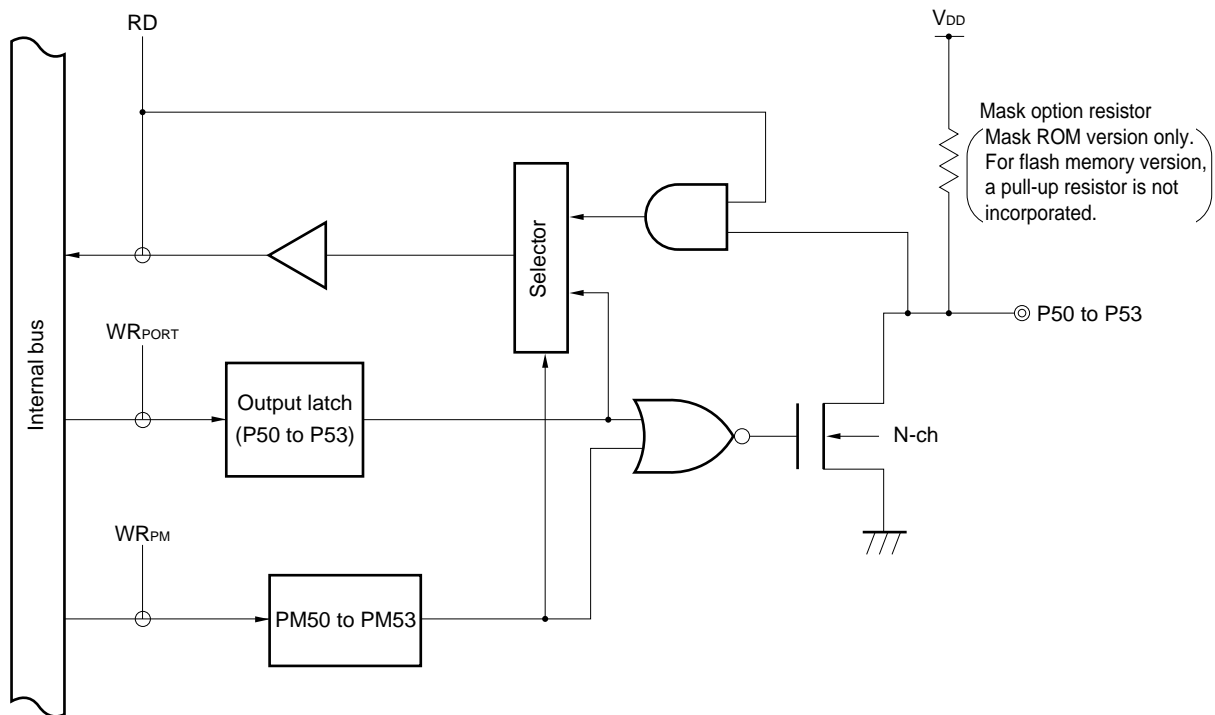
### 5.2.4 Port 5

This is a 4-bit N-ch open-drain I/O port with output latches. Port 5 can be specified as input or output mode in 1-bit units by using port mode register 5 (PM5). For a mask ROM version, whether a pull-up resistor is to be incorporated can be specified by the mask option.

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

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

Figure 5-8. Block Diagram of P50 to P53



PM: Port mode register  
RD: Port 5 read signal  
WR: Port 5 write signal

**Caution** When using port 5 of  $\mu\text{PD78F9116A}$  and  $78F9136A$  as an input port, be sure to observe the restrictions listed below.

<1> When  $V_{DD} = 1.8$  to  $5.5$  V

Use within the range of  $T_A = 25$  to  $85^\circ\text{C}$

<2> When  $T_A = -40$  to  $+85^\circ\text{C}$

Use within the range of  $V_{DD} = 2.7$  to  $5.5$  V

<3> When  $T_A = -40$  to  $+85^\circ\text{C}$  and  $V_{DD} = 1.8$  to  $5.5$  V

Issue three consecutive read instructions when reading port 5.

If the above restrictions are not observed, the input value may be read incorrectly.

**Note**, however, that these restrictions do not apply when port 5 pins are used as output pins, or when the product is other than  $\mu\text{PD78F9116A}$  or  $78F9136A$ .

### 5.2.5 Port 6

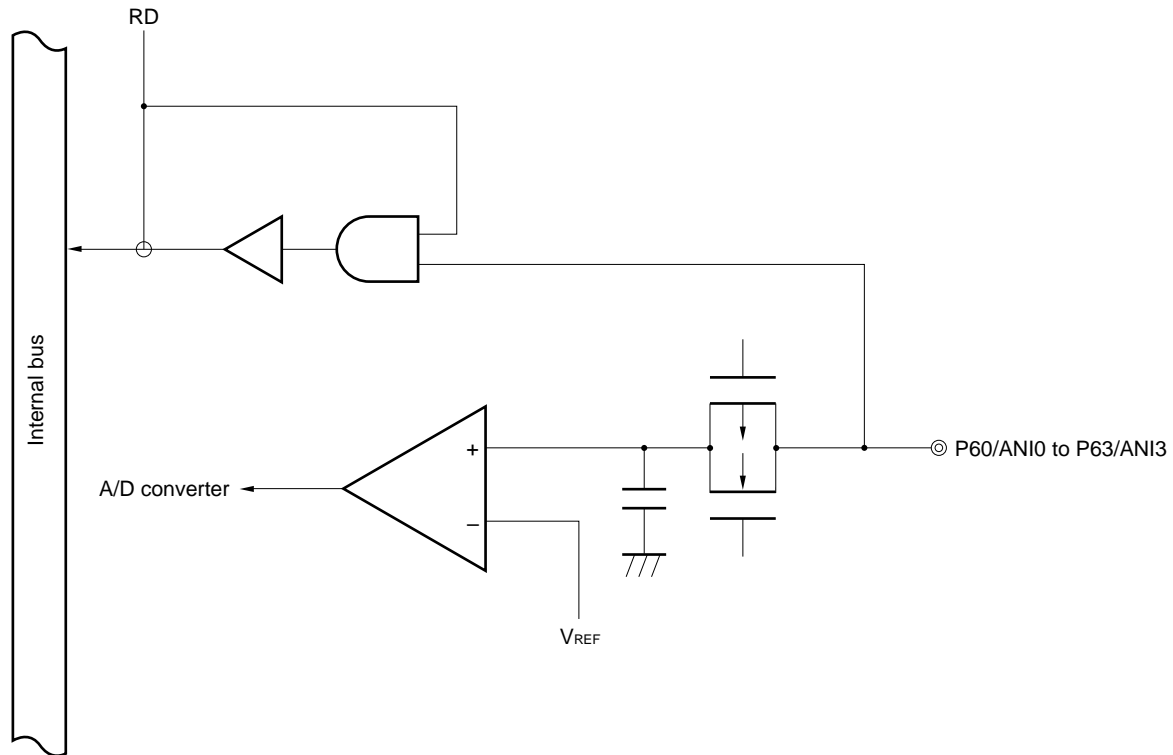
This is a 4-bit input port.

The port is also used as an analog input to the A/D converter.

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

Figure 5-9 shows a block diagram of port 6.

**Figure 5-9. Block Diagram of P60 to P63**



### 5.3 Port Function Control Registers

The following three types of registers control the ports.

- Port mode registers (PM0 to PM2, PM5)
- Pull-up resistor option register 0 (PU0)
- Pull-up resistor option register B2 (PUB2)

#### (1) Port mode registers (PM0 to PM2, 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 5-3.

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

**Table 5-3. Port Mode Register and Output Latch Settings When Using Alternate Functions**

Pin Name	Alternate Function		PM <sub>xx</sub>	P <sub>xx</sub>
	Name	Input/Output		
P23	INTP0	Input	1	×
	CPT20	Input	1	×
P24	INTP1	Input	1	×
	TO80	Output	0	0
	TO20	Output	0	0
P25	INTP2	Input	1	×
	TI80	Input	1	×

**Caution** When Port 2 is used for serial interface pin, the I/O latch or output latch must be set according to its function. For the setting method, refer to Table 13-2 Serial Interface 20 Operating Mode Settings.

**Remark** ×: don't care  
 PM<sub>xx</sub>: Port mode register  
 P<sub>xx</sub>: Port output latch

**Figure 5-10. Port Mode Register Format**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	1	1	1	1	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM1	1	1	1	1	1	1	PM11	PM10	FF21H	FFH	R/W
PM2	1	1	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM5	1	1	1	1	PM53	PM52	PM51	PM50	FF25H	FFH	R/W

PMmn	Pmn pin input/output mode selection (m = 0 to 2, 5, n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

**(2) Pull-up resistor option register 0 (PU0)**

The pull-up resistor option register (PU0) sets whether to use on-chip pull-up resistors at each port or not. At a port where use of on-chip pull-up resistors has been specified by PU0, the pull-up resistors can be internally used only for the bits set in input mode. No on-chip pull-up resistors can be used for the bits set in output mode, in spite of setting PU0. On-chip pull-up resistors can also not be used when the pins are used as the alternate-function output pins.

PU0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears PU0 to 00H.

**Figure 5-11. Pull-Up Resistor Option Register 0 Format**

Symbol	7	6	5	4	3	2	<1>	<0>	Address	After reset	R/W
PU0	0	0	0	0	0	0	PU01	PU00	FFF7H	00H	R/W

PU0m	Pm on-chip pull-up resistor selection (m = 0, 1)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

**(3) Pull-up resistor option register B2 (PUB2)**

This register specifies whether an on-chip pull-up resistor is connected to each pin of port 2. The pin so specified by PUB2 is connected to on-chip pull-up resistor regardless of the setting of the port mode register.

PUB2 is set with a 1-bit or 8-bit manipulation instruction.

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

**Figure 5-12. Pull-Up Resistor Option Register B2 Format**

Symbol	7	6	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
PUB2	0	0	PUB25	PUB24	PUB23	PUB22	PUB21	PUB20	FF32H	00H	R/W

PUB2n	P2n on-chip pull-up resistor selection (n = 0 to 5)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

## 5.4 Operation of Port Functions

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

### 5.4.1 Writing to I/O port

#### (1) In output mode

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

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

#### (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.

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

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

### 5.4.2 Reading from I/O port

#### (1) In output mode

The contents of the 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.

**Caution** When using port 5 of  $\mu$ PD78F9116A and 78F9136A as an input port, be sure to observe the restrictions listed below.

<1> When  $V_{DD} = 1.8$  to  $5.5$  V

Use within the range of  $T_A = 25$  to  $85^\circ\text{C}$

<2> When  $T_A = -40$  to  $+85^\circ\text{C}$

Use within the range of  $V_{DD} = 2.7$  to  $5.5$  V

<3> When  $T_A = -40$  to  $+85^\circ\text{C}$  and  $V_{DD} = 1.8$  to  $5.5$  V

Issue three consecutive read instructions when reading port 5.

If the above restrictions are not observed, the input value may be read incorrectly.

Note, however, that these restrictions do not apply when port 5 pins are used as output pins, or when the product is other than  $\mu$ PD78F9116A or 78F9136A.

### 5.4.3 Arithmetic operation of I/O port

#### (1) In output mode

An arithmetic operation can be performed with the contents of the 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.

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

#### (2) In input mode

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

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

## CHAPTER 6 CLOCK GENERATOR ( $\mu$ PD789104A, 789114A SUBSERIES)

### 6.1 Function of Clock Generator

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

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

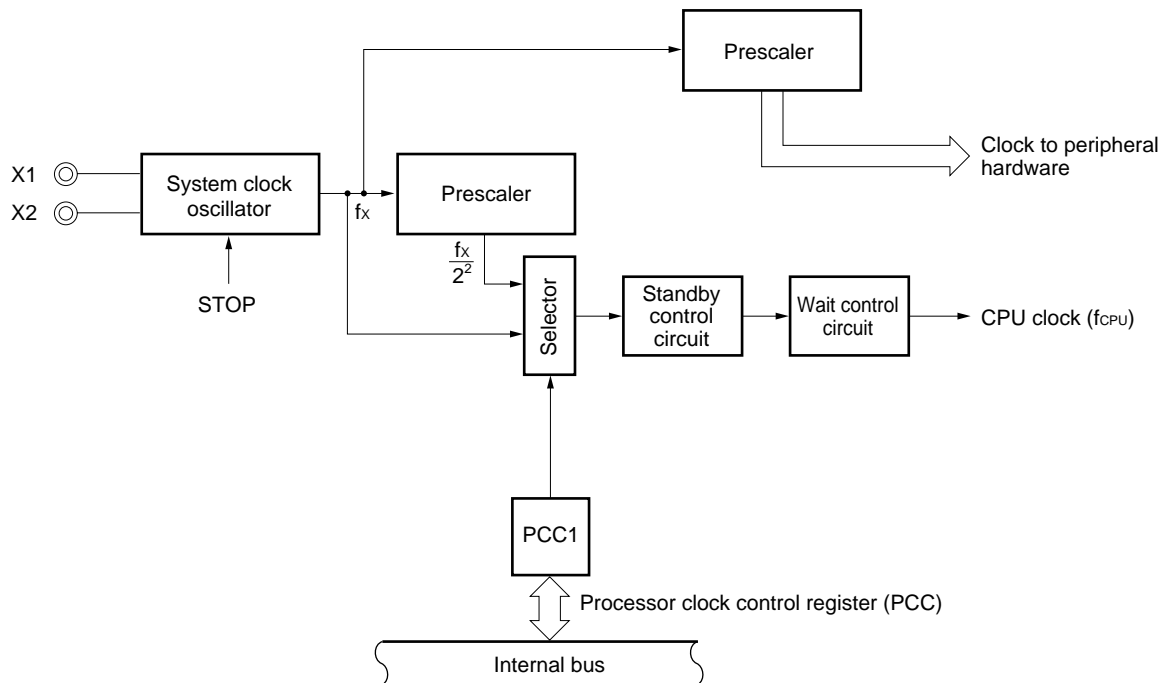
### 6.2 Configuration of Clock Generator

The clock generator consists of the following hardware.

**Table 6-1. Configuration of Clock Generator**

Item	Configuration
Control register	Processor clock control register (PCC)
Oscillator	Crystal/ceramic oscillator

**Figure 6-1. Block Diagram of Clock Generator**



6.3 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 the CPU clock selection and the ratio of division.  
PCC is set with a 1-bit or 8-bit memory manipulation instruction.  
RESET input sets PCC to 02H.

Figure 6-2. Processor Clock Control Register Format

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

PCC1	CPU clock (fCPU) selection
0	fx (0.2 μs)
1	fx/2 <sup>2</sup> (0.8 μs)

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

- Remarks
1. fx: System clock oscillation frequency
  2. Values in parentheses are when operating at fx = 5.0 MHz.
  3. Minimum instruction execution time: 2fCPU
    - When fCPU = 0.2 μs: 0.4 μs
    - When fCPU = 0.8 μs: 1.6 μs



## 6.4 System Clock Oscillator

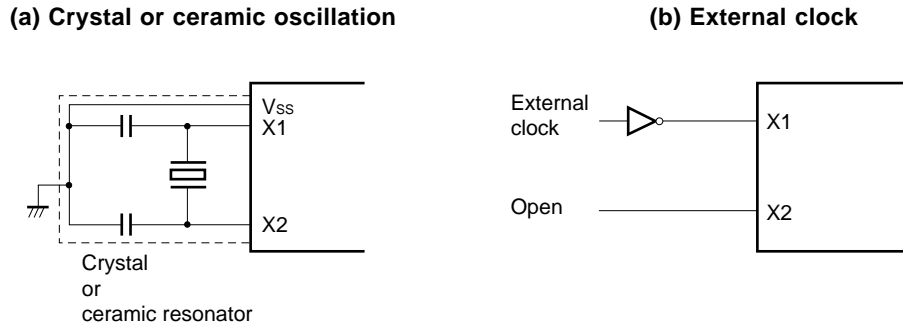
### 6.4.1 System clock oscillator

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 system clock oscillator. In this case, input the clock signal to the X1 pin, and leave the X2 pin open.

Figure 6-3 shows the external circuit of the system clock oscillator.

**Figure 6-3. 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 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.

Figure 6-4 shows incorrect examples of resonator connection.

Figure 6-4. Examples of Incorrect Resonator Connection (1/2)

(a) Too long wiring

(b) Crossed signal line

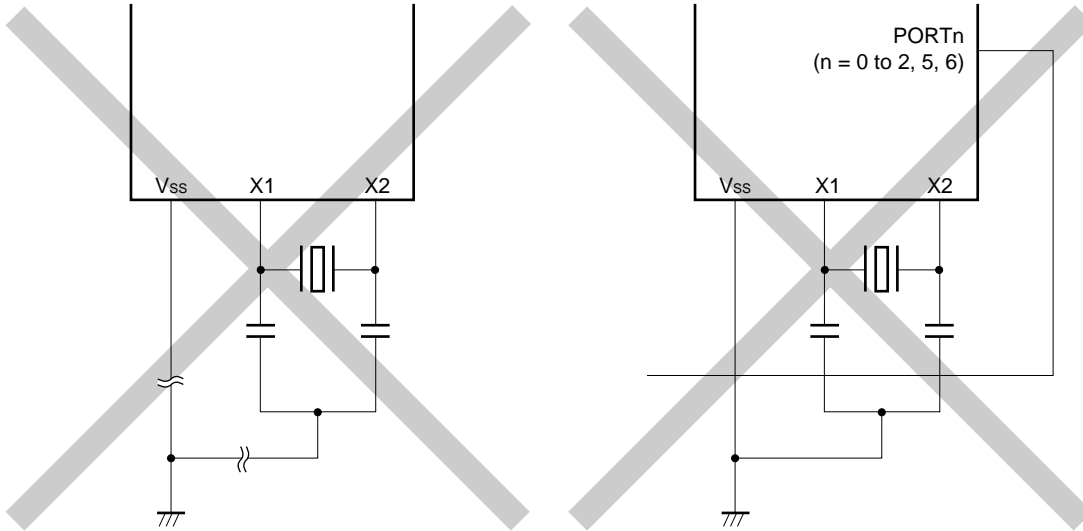
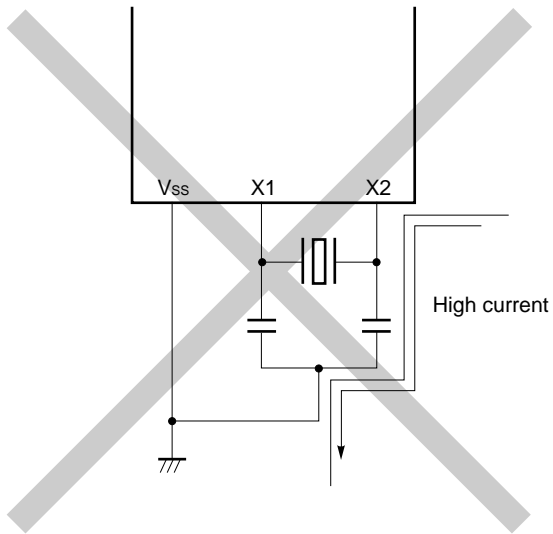
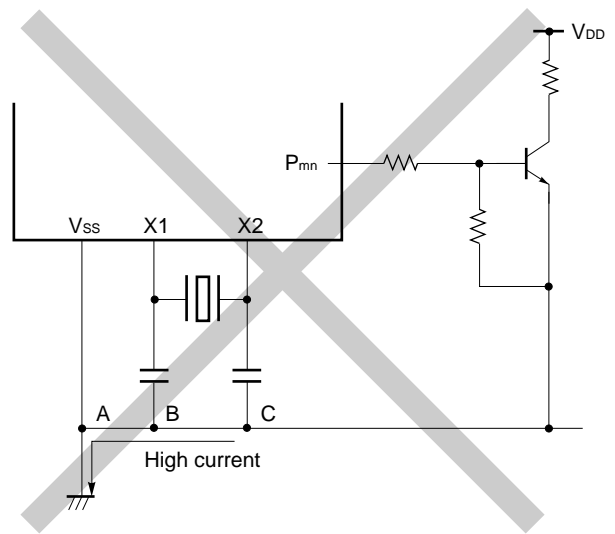


Figure 6-4. Examples of Incorrect Resonator Connection (2/2)

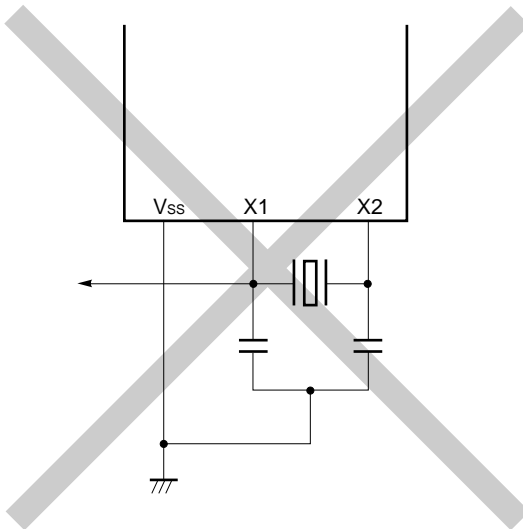
(c) Wiring near high fluctuating current



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



(e) Signal is fetched



### 6.4.2 Divider

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

## 6.5 Operation of Clock Generator

The clock generator generates the following clocks and controls the operating 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  $2f_{CPU}$  ( $1.6 \mu s$ : 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 input to the  $\overline{RESET}$  pin, oscillation of the system clock is stopped.
- (b) Two types of CPU clocks  $f_{CPU}$  ( $0.2 \mu s$  and  $0.8 \mu s$ : 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 the external clock input operation).

## 6.6 Changing Setting of CPU Clock

### 6.6.1 Time required for switching CPU clock

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

Actually, the specified clock is not selected immediately after the setting of PCC has been changed, and the old clock is used for the duration of several instructions after that (refer to **Table 6-2**).

**Table 6-2. Maximum Time Required for Switching CPU Clock**

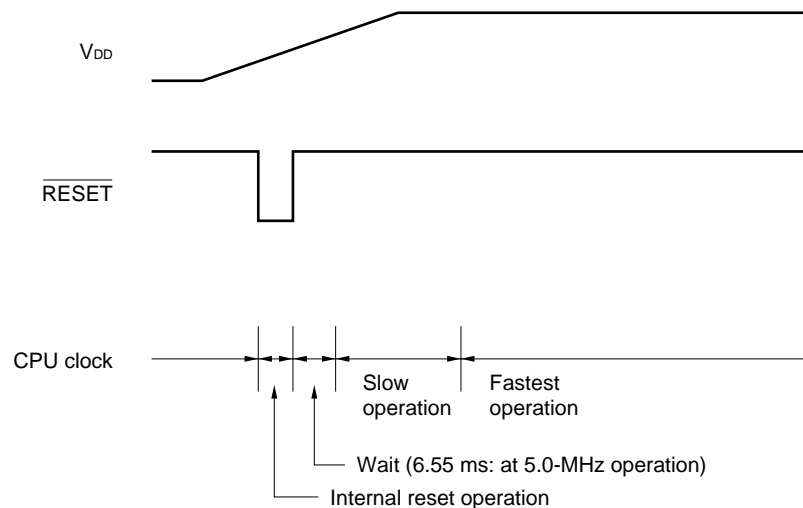
Set Value before Switching	Set Value after Switching	
PCC1	PCC1	PCC1
	0	1
0	2 clocks	4 clocks
1		

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

### 6.6.2 Switching CPU clock

The following figure illustrates how the CPU clock switches.

**Figure 6-5. Switching CPU Clock**



<1> The CPU is reset when the  $\overline{\text{RESET}}$  pin is made low on power application. The effect of resetting is released when the  $\overline{\text{RESET}}$  pin is later made high, and the system clock starts oscillating. At this time, the time during which oscillation stabilizes ( $2^{15}/f_x$ ) is automatically secured.

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

**[MEMO]**

## CHAPTER 7 CLOCK GENERATOR (μPD789124A, 789134A SUBSERIES)

### 7.1 Function of Clock Generator

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

- System clock (RC) oscillator

This circuit oscillates at frequencies of 2.0 to 4.0 MHz. Oscillation can be stopped by executing the STOP instruction.

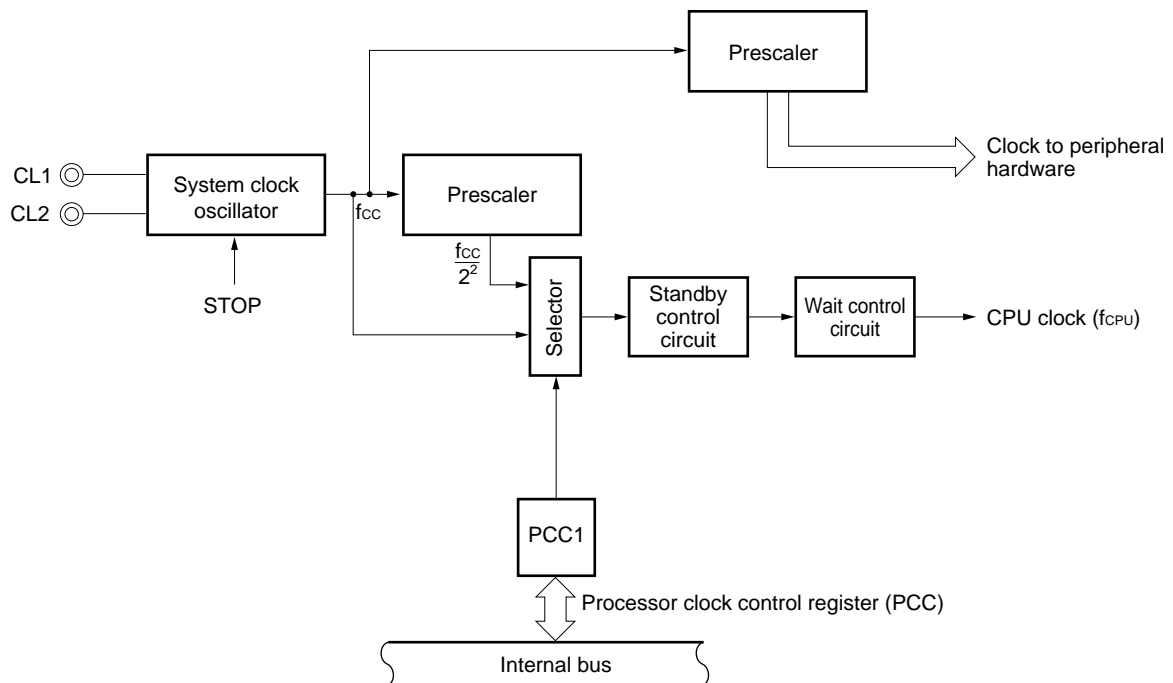
### 7.2 Configuration of Clock Generator

The clock generator consists of the following hardware.

**Table 7-1. Configuration of Clock Generator**

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

**Figure 7-1. Block Diagram of Clock Generator**



### 7.3 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 the CPU clock selection and the ratio of division.

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

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

**Figure 7-2. Processor Clock Control Register Format**

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

PCC1	CPU clock ( $f_{\text{CPU}}$ ) selection
0	$f_{\text{CC}}$ (0.25 $\mu\text{s}$ )
1	$f_{\text{CC}}/2^2$ (1.0 $\mu\text{s}$ )

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

- Remarks**
1.  $f_{\text{CC}}$ : System clock oscillation frequency
  2. Values in parentheses are when operating at  $f_x = 4.0$  MHz.
  3. Minimum instruction execution time:  $2f_{\text{CPU}}$ 
    - When  $f_{\text{CPU}} = 0.25 \mu\text{s}$ :  $0.5 \mu\text{s}$
    - When  $f_{\text{CPU}} = 1.0 \mu\text{s}$ :  $2.0 \mu\text{s}$



## 7.4 System Clock Oscillator

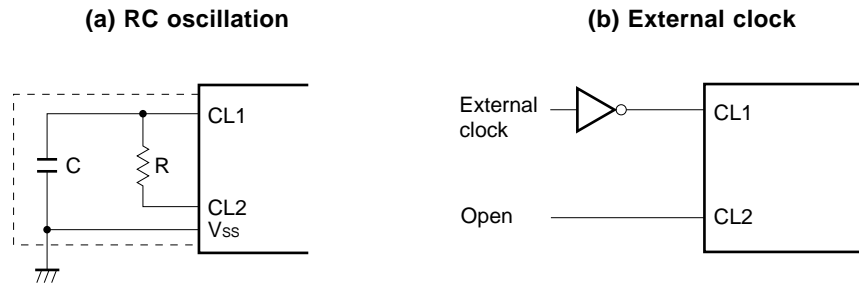
### 7.4.1 System clock oscillator

The system clock oscillator is oscillated by the resistor (R) and capacitor (C) (4.0 MHz TYP.) connected across the CL1 and CL2 pins.

An external clock can also be input to the system clock oscillator. In this case, input the clock signal to the CL1 pin, and leave the CL2 pin open.

Figure 7-3 shows the external circuit of the system clock oscillator.

**Figure 7-3. External Circuit of System Clock Oscillator**



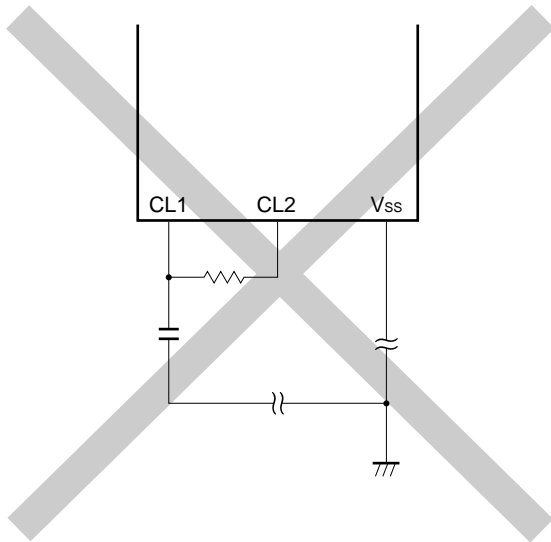
**Caution** When using the system clock oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

- 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 Vss. Do not ground the capacitor to a ground pattern through which a high current flows.
- Do not fetch signals from the oscillator.

Figure 7-4 shows incorrect examples of resonator connection.

Figure 7-4. Examples of Incorrect Resonator Connection (1/2)

(a) Too long wiring



(b) Crossed signal line

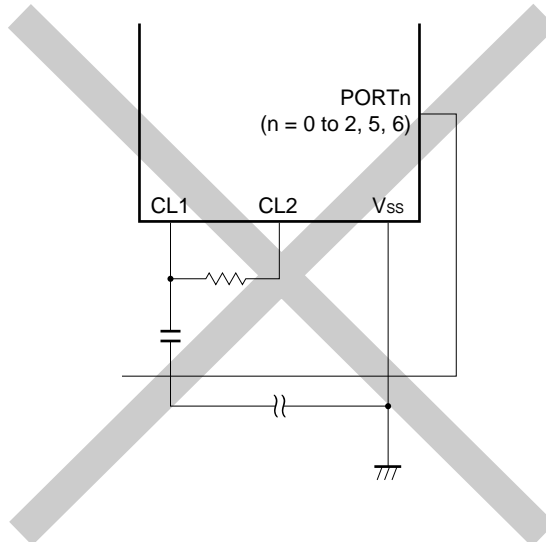
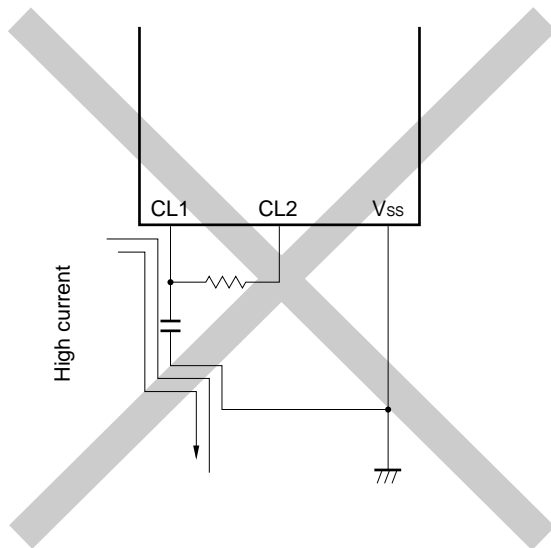
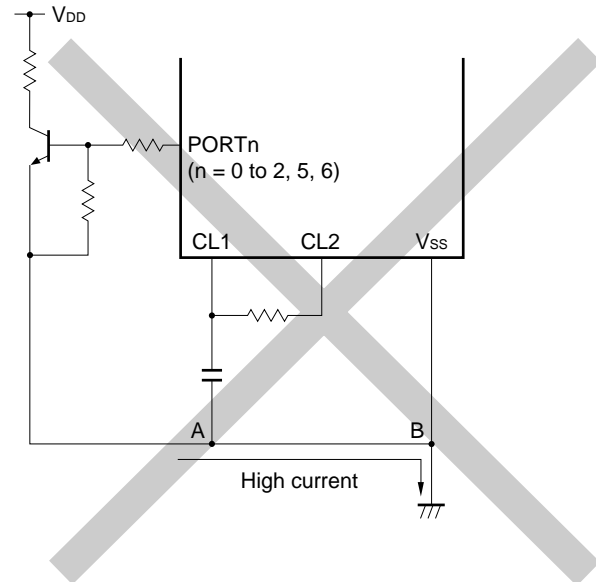


Figure 7-4. Examples of Incorrect Resonator Connection (2/2)

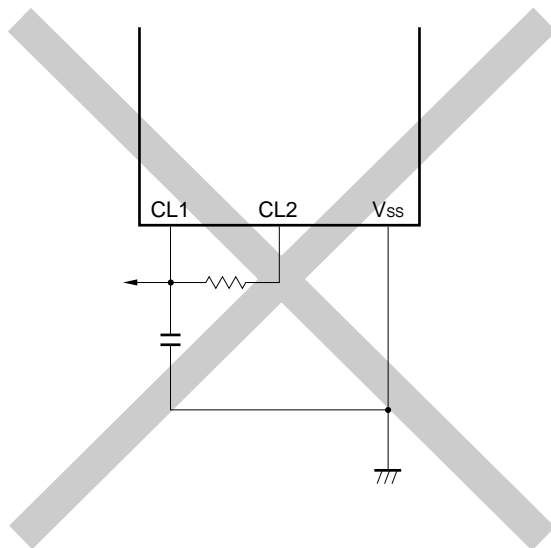
(c) Wiring near high fluctuating current



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



(e) Signal is fetched



### 7.4.2 Divider

The divider divides the output of the system clock oscillator ( $f_{CC}$ ) to generate various clocks.

## 7.5 Operation of Clock Generator

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

- System clock  $f_{CC}$
- 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  $2f_{CPU}$  ( $2.0 \mu s$ : at 4.0-MHz operation) of the system clock is selected when the  $\overline{RESET}$  signal is generated ( $PCC = 02H$ ). While a low level is input to the  $\overline{RESET}$  pin, oscillation of the system clock is stopped.
- (b) Two types of CPU clocks  $f_{CPU}$  ( $0.5 \mu s$  and  $1.0 \mu s$ : at 4.0-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 the external clock input operation).

## 7.6 Changing Setting of CPU Clock

### 7.6.1 Time required for switching CPU clock

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

Actually, the specified clock is not selected immediately after the setting of PCC has been changed, and the old clock is used for the duration of several instructions after that (refer to **Table 7-2**).

**Table 7-2. Maximum Time Required for Switching CPU Clock**

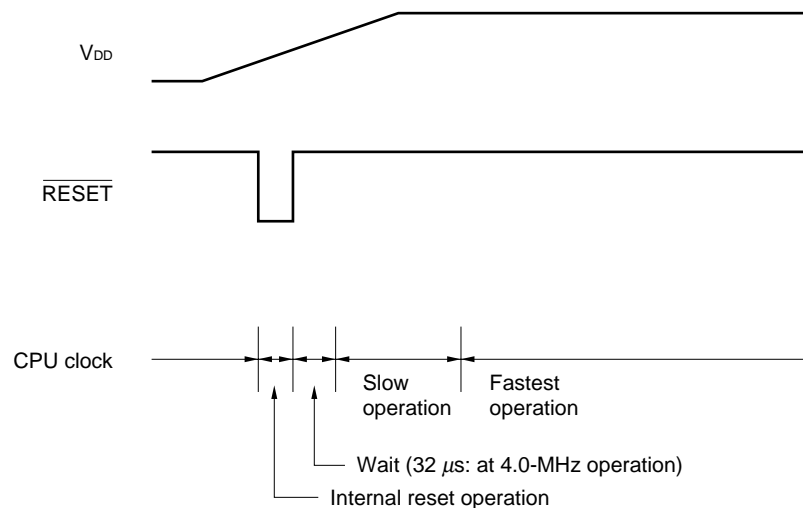
Set Value before Switching	Set Value after Switching	
PCC1	PCC1	PCC1
	0	1
0	2 clocks	4 clocks
1		

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

### 7.6.2 Switching CPU clock

The following figure illustrates how the CPU clock switches.

**Figure 7-5. Switching CPU Clock**



<1> The CPU is reset when the  $\overline{\text{RESET}}$  pin is made low on power application. The effect of resetting is released when the  $\overline{\text{RESET}}$  pin is later made high, and the system clock starts oscillating. At this time, the time during which oscillation stabilizes ( $2^7/f_{cc}$ ) is automatically secured.

After that, the CPU starts instruction execution at the low speed of the system clock (2.0  $\mu$ s: at 4.0-MHz operation).

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

**[MEMO]**

## CHAPTER 8 16-BIT TIMER 20

The 16-bit timer counter references the free running counter and provides the functions such as timer interrupt and timer output. In addition, the count value can be captured by a trigger pin.

### 8.1 16-Bit Timer 20 Functions

16-bit timer 20 has the following functions.

- Timer interrupt
- Timer output
- Count value capture

#### (1) Timer interrupt

An interrupt is generated when a count value and compare value matches.

#### (2) Timer output

Timer output control is possible when an count value and compare value matches.

#### (3) Count value capture

A TM20 count value is latched in synchronization with the capture trigger and retained.

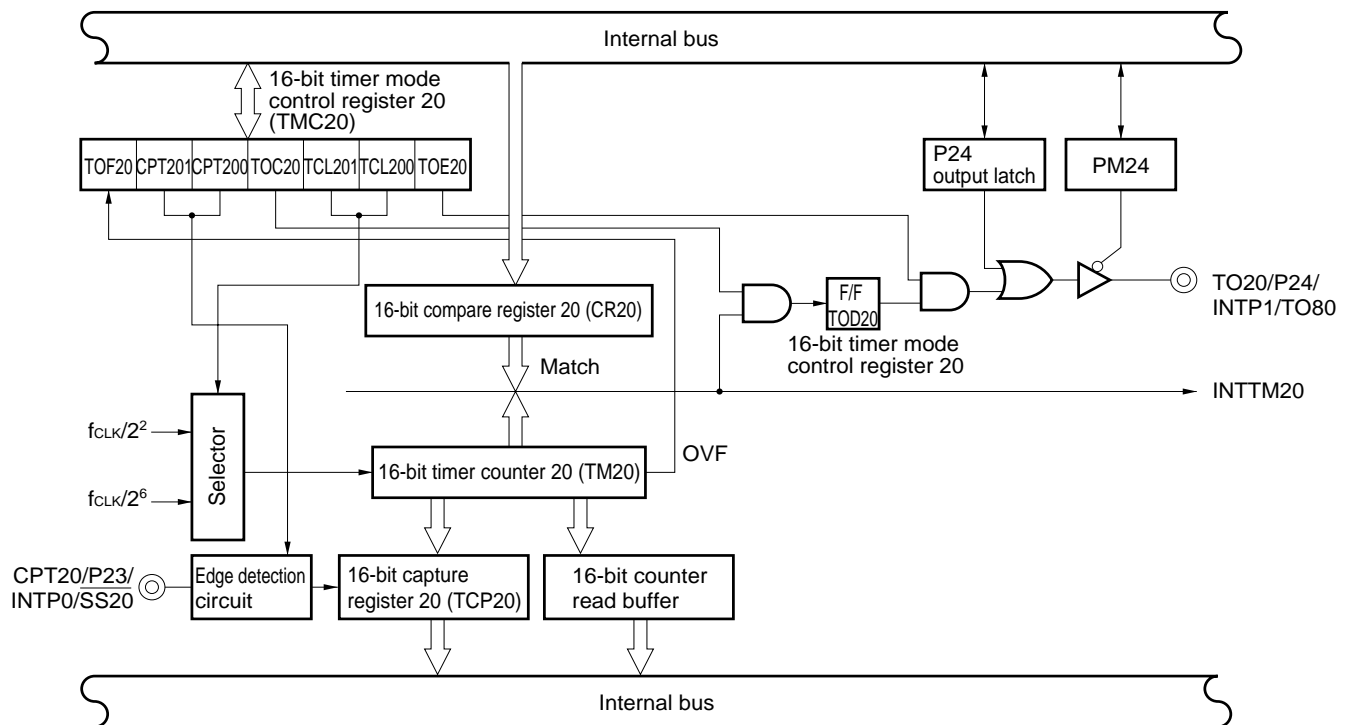
## 8.2 16-Bit Timer 20 Configuration

16-bit timer 20 consists of the following hardware.

**Table 8-1. Configuration of 16-Bit Timer 20**

Item	Configuration
Timer counter	16 bits × 1 (TM20)
Register	Compare register: 16 bits × 1 (CR20) Capture register: 16 bits × 1 (TCP20)
Timer output	1 (TO20)
Control register	16-bit timer mode control register 20 (TMC20) Port mode register 2 (PM2)

**Figure 8-1. Block Diagram of 16-Bit Timer 20**



**Remark**  $f_{CLK}$ :  $f_x$  or  $f_{cc}$



**(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 (INTTM20).

CR20 is set with a 16-bit memory manipulation instruction. The values 0000H to FFFFH can be set.  $\overline{\text{RESET}}$  input sets this register to FFFFH.

- Cautions**
1. Although this register is manipulated with a 16-bit memory manipulation instruction, 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.
  2. When rewriting CR20 during count operation, set CR20 to interrupt disable from interrupt mask flag register 0 (MK10) beforehand. Also, set the timer output data to inversion disable using 16-bit timer mode control register 20 (TMC20).  
When CR20 is rewritten in the interrupt-enabled state, an interrupt request may occur at the moment of rewrite.

**(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 free running during count clock input.

$\overline{\text{RESET}}$  input clears this register to 0000H and after which it resumes free running.

- Cautions**
1. The count value after releasing stop becomes undefined because the count operation is executed during the oscillation stabilization time.
  2. Although this register is manipulated with a 16-bit memory manipulation instruction, 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.
  3. When manipulated with an 8-bit memory manipulation instruction, readout should be performed in the 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 sets this register to undefined.

**Caution** Although this register is manipulated with a 16-bit memory manipulation instruction, 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.

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

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

### 8.3 Registers Controlling 16-Bit Timer 20

The following two types of registers control 16-bit timer 20.

- 16-bit timer mode control register 20 (TMC20)
- Port mode register 2 (PM2)

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

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

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

RESET input clears TMC20 to 00H.

Figure 8-2. 16-Bit Timer Mode Control Register 20 Format

Symbol	7	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC20	TOD20	TOF20	CPT201	CPT200	TOC20	TCL201	TCL200	TOE20	FF48H	00H	R/W <sup>Note</sup>

TOD20	Timer output data	
0	Timer output of 0	
1	Timer output of 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 CPT20
1	0	Falling edge of CPT20
1	1	Both edges of CPT20

TOC20	Timer output data inverse control	
0	Inverse disabled	
1	Inverse enabled	

TCL201	TCL200	16-bit timer counter 20 count clock selection	
		At $f_x = 5.0 \text{ MHz}$	At $f_{cc} = 4.0 \text{ MHz}$
0	0	$f_x/2^2$ (1.25 MHz)	$f_{cc}/2^2$ (1.0 MHz)
0	1	$f_x/2^6$ (78.1 kHz)	$f_{cc}/2^6$ (62.5 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.

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)  
 $f_{cc}$ : System clock oscillation frequency (RC oscillation)

**(2) Port mode register 2 (PM2)**

This register sets the input/output of port 2 in 1-bit units.

To use the P24/TO20/INTP1/TO80 pin for timer output, set the output latch of PM24 and P24 to 0.

PM2 is set with a 1-bit or 8-bit memory manipulation instruction.

RESET input sets PM2 to FFH.

**Figure 8-3. Port Mode Register 2 Format**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM2	1	1	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W

PM24	P24 pin input/output mode selection
0	Output mode (output buffer on)
1	Input mode (output buffer off)

## 8.4 16-Bit Timer 20 Operation

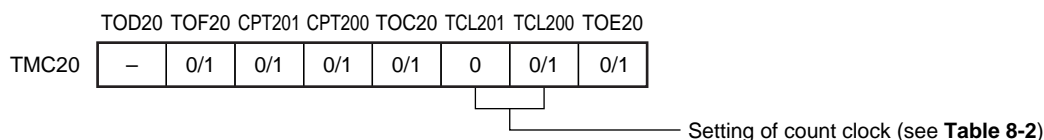
### 8.4.1 Operation as timer interrupt

In the timer interrupt function, interrupts are repeatedly generated at the count value set to 16-bit compare register 20 (CR20) in advance based on the intervals of the value set in TCL201 and TCL200.

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

- Set count values to CR20
- Set 16-bit timer mode control counter 20 (TMC20) as shown in Figure 8-4.

**Figure 8-4. Settings of 16-Bit Timer Mode Control Register 20 at Timer Interrupt Operation**



**Caution** If both the CPT201 and CPT200 flags are set to 0, the capture edge becomes setting prohibited.

When the count value of 16-bit timer counter 20 (TM20) coincides with the value set to CR20, counting of TM20 continues and an interrupt request signal (INTTM20) is generated.

Table 8-3 shows the interval time, and Figure 8-5 shows the timing of the timer interrupt operation.

**Caution** When rewriting CR20 during count operation, be sure to follow the procedure below.

- <1> Set CR20 to interrupt disable (by setting bit 7 of interrupt mask flag register 0 (MK0) to 1).
- <2> Set inversion control of timer output data to disable (TOC20 = 0)

When CR20 is rewritten in the interrupt-enabled state, an interrupt request may occur at the moment of rewrite.

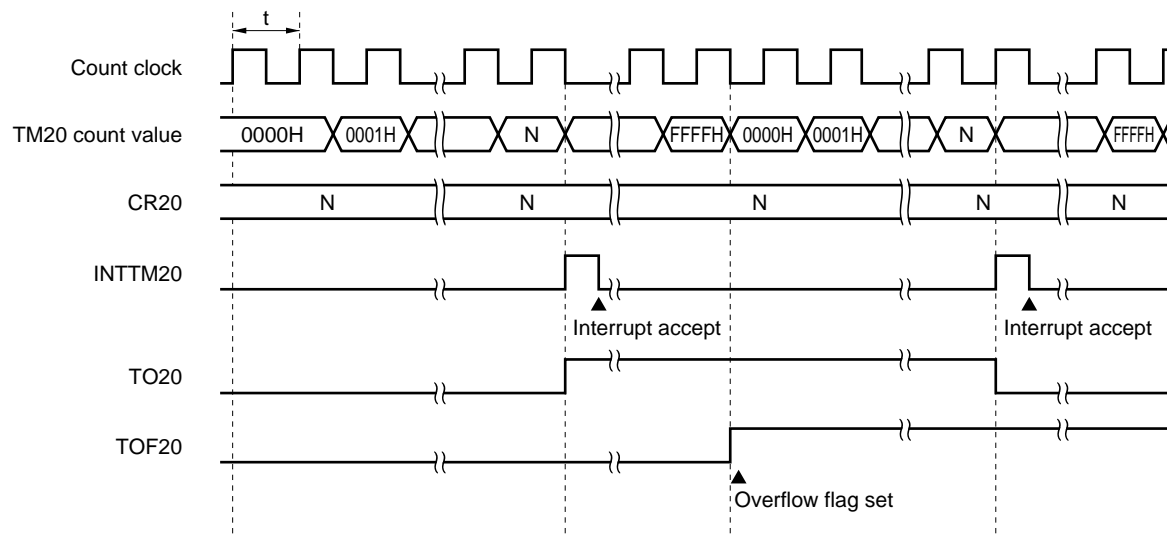
**Table 8-2. Interval Time of 16-Bit Timer 20**

TCL201	TCL200	Count Clock		Interval Time	
		At $f_x = 5.0 \text{ MHz}$	At $f_{cc} = 4.0 \text{ MHz}$	At $f_x = 5.0 \text{ MHz}$	At $f_{cc} = 4.0 \text{ MHz}$
0	0	$2^2/f_x$ (0.8 $\mu\text{s}$ )	$2^2/f_{cc}$ (1.0 $\mu\text{s}$ )	$2^{18}/f_x$ (52.4 ms)	$2^{18}/f_{cc}$ (65.5 ms)
0	1	$2^6/f_x$ (12.8 $\mu\text{s}$ )	$2^6/f_{cc}$ (16 $\mu\text{s}$ )	$2^{22}/f_x$ (838.9 ms)	$2^{22}/f_{cc}$ (1,048 ms)
Other than above		Setting prohibited			

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)

Figure 8-5. Timing of Timer Interrupt Operation



**Remark** N = 0000H to FFFFH

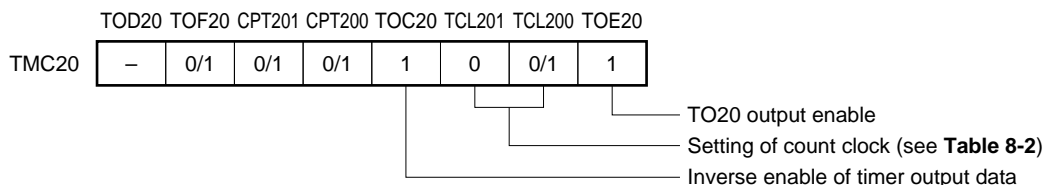
### 8.4.2 Operation as timer output

Timer outputs are repeatedly generated at the count value set to 16-bit compare register 20 (CR20) in advance based on the intervals of the value set in TCL201 and TCL200.

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

- Set P24 to output mode (PM24 = 0)
- Set P24 output latch to 0
- Set the count value to CR20
- Set 16-bit timer mode control register 20 (TMC20) as shown in Figure 8-6

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

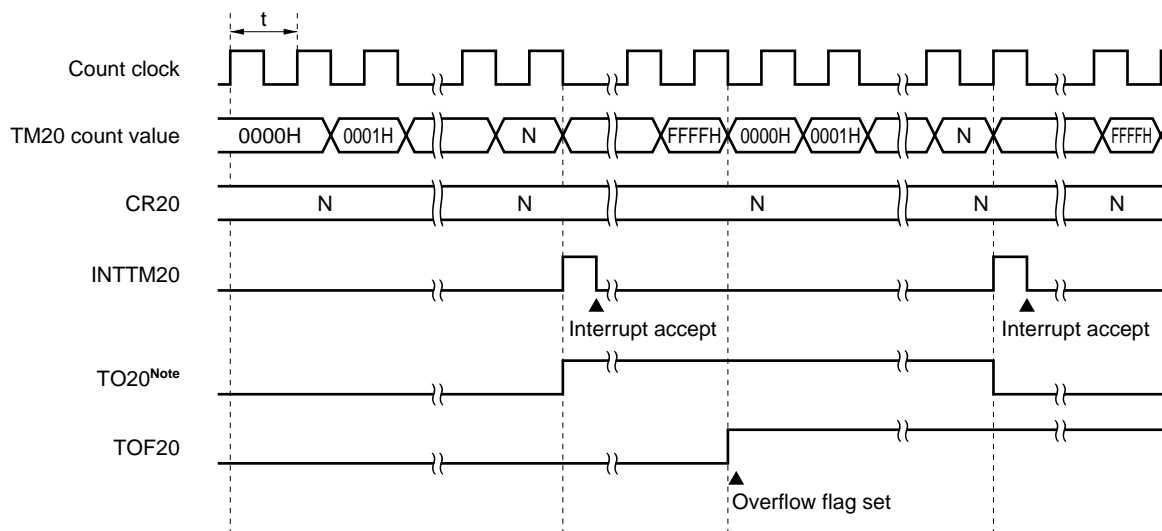


**Caution** If both CPT201 flag and CPT200 flag are set to 0, the capture edge becomes operation prohibited.

When the count value of the 16-bit timer counter 20 (TM20) matches the value set in CR20, the output status of the TO20/P24/INTP1/TO80 pin is inverted. This enables timer output. At that time, TM20 count is continued and an interrupt request signal (INTTM20) is generated.

Figure 8-7 shows the timing of timer output (see **Table 8-2** for the interval time of the 16-bit timer 20).

**Figure 8-7. Timer Output Timing**



**Note** The TO20 initial value becomes low level during output enable (TOE20 = 1).

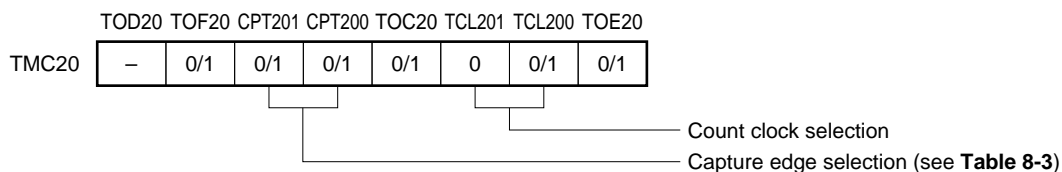
**Remark** N = 0000H to FFFFH

### 8.4.3 Capture operation

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

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

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



16-bit capture register 20 (TCP20) starts the capture operation after the CPT20 capture trigger edge has been 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.

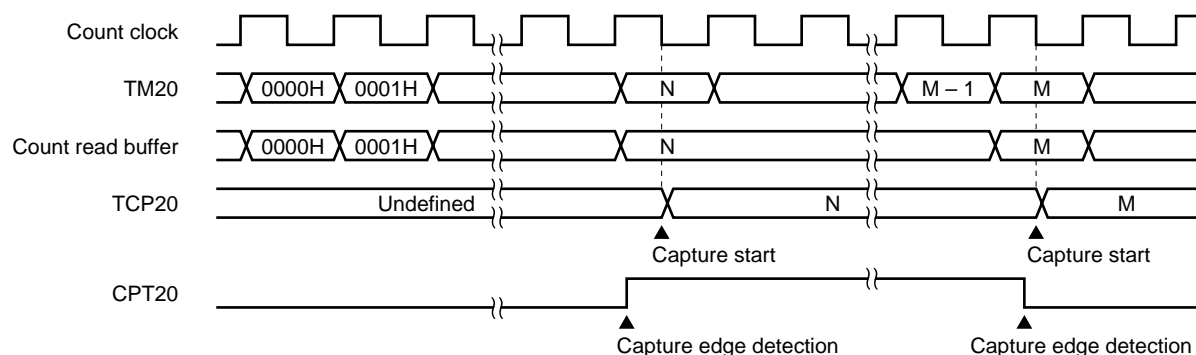
Table 8-3 and Figure 8-9 show the setting contents of the capture edge and capture operation timing, respectively.

**Table 8-3. Settings of Capture Edge**

CPT201	CPT200	Capture Edge Selection
0	0	Capture operation prohibited
0	1	CPT20 pin rising edge
1	0	CPT20 pin falling edge
1	1	CPT20 pin both edges

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

**Figure 8-9. Capture Operation Timing (Both Edges of CPT20 Pin Are Specified)**





#### 8.4.4 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 through a counter read buffer. The counter read buffer latches the TM20 count value. 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 at the retention state can be read out as the count value.

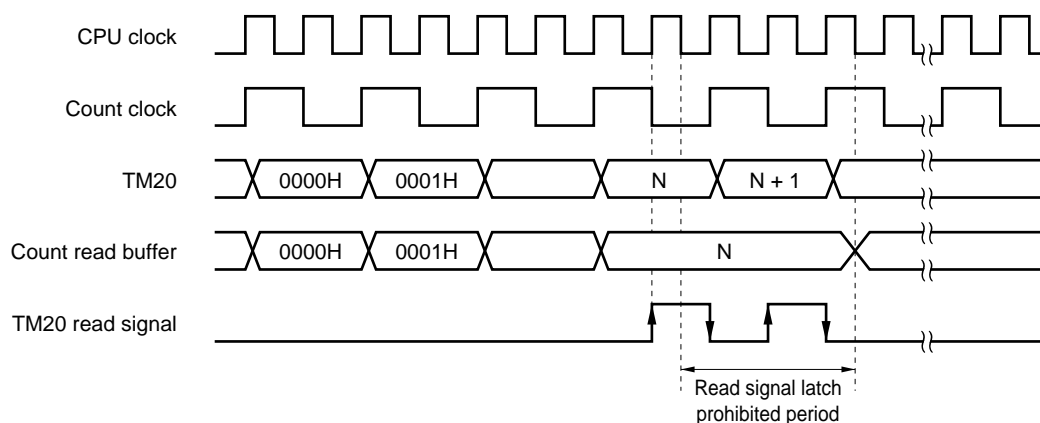
Cancellation of the pending state is performed at the CPU clock falling edge after the read signal of the TM20 higher byte falls.

$\overline{\text{RESET}}$  input clears TM20 to 0000H and restarts free running.

Figure 8-10 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 oscillation stabilization time.
  2. Although TM20 is a dedicated 16-bit transfer instruction register, an 8-bit transfer instruction can be used.  
Execute an 8-bit transfer instruction by direct addressing.
  3. When using an 8-bit transfer instruction, execute in the 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 8-10. 16-Bit Timer Counter 20 Readout Timing



**[MEMO]**

## CHAPTER 9 8-BIT TIMER/EVENT COUNTER 80

The 8-bit timer/event counter can be used as an interval timer, external event counter, and for square wave output and PWM output of arbitrary frequency.

### 9.1 Functions of 8-Bit Timer/Event Counter 80

8-bit timer/event counter 80 has the following functions:

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

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

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

**Table 9-1. Interval Time of 8-Bit Timer/Event Counter 80**

	Minimum Interval Time	Maximum Interval Time	Resolution
At $f_x = 5.0$ MHz	$1/f_x$ (200 ns)	$2^8/f_x$ (51.2 $\mu$ s)	$1/f_x$ (200 ns)
	$2^3/f_x$ (1.6 $\mu$ s)	$2^{11}/f_x$ (409.6 $\mu$ s)	$2^3/f_x$ (1.6 $\mu$ s)
At $f_{cc} = 4.0$ MHz	$1/f_{cc}$ (250 ns)	$2^8/f_{cc}$ (64 $\mu$ s)	$1/f_{cc}$ (250 ns)
	$2^3/f_{cc}$ (2.0 $\mu$ s)	$2^{11}/f_{cc}$ (512 $\mu$ s)	$2^3/f_{cc}$ (2.0 $\mu$ s)

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)

#### (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 9-2. Square Wave Output Range of 8-Bit Timer/Event Counter 80**

	Minimum Pulse Width	Maximum Pulse Width	Resolution
At $f_x = 5.0$ MHz	$1/f_x$ (200 ns)	$2^8/f_x$ (51.2 $\mu$ s)	$1/f_x$ (200 ns)
	$2^3/f_x$ (1.6 $\mu$ s)	$2^{11}/f_x$ (409.6 $\mu$ s)	$2^3/f_x$ (1.6 $\mu$ s)
At $f_{cc} = 4.0$ MHz	$1/f_{cc}$ (250 ns)	$2^8/f_{cc}$ (64 $\mu$ s)	$1/f_{cc}$ (250 ns)
	$2^3/f_{cc}$ (2.0 $\mu$ s)	$2^{11}/f_{cc}$ (512 $\mu$ s)	$2^3/f_{cc}$ (2.0 $\mu$ s)

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)

**(4) PWM output**

8-bit resolution PWM output can be produced.

**9.2 8-Bit Timer/Event Counter 80 Configuration**

8-bit timer/event counter 80 consists of the following hardware.

**Table 9-3. 8-Bit Timer/Event Counter 80 Configuration**

Item	Configuration
Timer counter	8 bits × 1 (TM80)
Register	Compare register: 8 bits × 1 (CR80)
Timer output	1 (TO80)
Control register	8-bit timer mode control register 80 (TMC80) Port mode register 2 (PM2)

This is an 8-bit register that compares the value set to CR80 with the 8-bit timer counter 80 (TM80) count value,

CR80 is set with an 8-bit memory manipulation instruction. The values 00H to FFH can be set.

RESET input makes CR80 undefined.

2. Do not set CR80 to 00H in the PWM output mode (when PWME80 = 1); otherwise, PWM

This is an 8-bit register to count count pulses.

TM80 is read with an 8-bit memory manipulation instruction.

RESET input clears TM80 to 00H.

### 9.3 8-Bit Timer/Event Counter 80 Control Registers

The following two types of registers are used to control the 8-bit timer/event counter 80.

- 8-bit timer mode control register 80 (TMC80)
- Port mode register 2 (PM2)

#### (1) 8-bit timer mode control register 80 (TMC80)

This register enables/stops operation of 8-bit timer counter 80 (TM80), sets the counter clock of TM80, and controls the operation of the output control circuit of 8-bit timer/event counter 80.

TMC80 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears TMC80 to 00H.

**Figure 9-2. 8-Bit Timer Mode Control Register 80 Format**

Symbol	<7>	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC80	TCE80	PWME80	0	0	0	TCL801	TCL800	TOE80	FF53H	00H	R/W

TCE80	8-bit timer counter 80 operation control
0	Operation stop (TM80 cleared to 0)
1	Operation enable

PWME80	Operation mode selection
0	Timer counter operating mode
1	PWM output operating mode

TCL801	TCL800	8-bit timer counter 80 count clock selection	
		At $f_x = 5.0\text{-MHz}$ operation	At $f_{cc} = 4.0\text{-MHz}$ operation
0	0	$f_x$ (5.0 MHz)	$f_{cc}$ (4.0 MHz)
0	1	$f_x/2^3$ (625 kHz)	$f_{cc}/2^3$ (500 kHz)
1	0	Rising edge of TI80	
1	1	Falling edge of TI80	

TOE80	8-bit timer/event counter 80 output control
0	Output disable (port mode)
1	Output enable

**Caution** Be sure to set TMC80 after stopping timer operation.

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)  
 $f_{cc}$ : System clock oscillation frequency (RC oscillation)

**(2) Port mode register 2 (PM2)**

This register sets port 2 to input/output in 1-bit units.

When using the P24/TO80/INTP1/TO20 pin for timer output, set the output latch of PM24 and P24 to 0.

PM2 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input sets PM2 to FFH.

**Figure 9-3. Port Mode Register 2 Format**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM2	1	1	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W

PM2n	P2n pin input/output mode selection (n = 0 to 5)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

## 9.4 Operation of 8-Bit Timer/Event Counter 80

### 9.4.1 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 80 (CR80) in advance.

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

- <1> Set 8-bit timer counter 80 (TM80) to operation disable (by setting TCE80 (bit 7 of 8-bit timer mode control register 80 (TMC80)) to 0).
- <2> Set the count clock of the 8-bit timer/event counter 80 (see **Tables 9-4** and **9-5**)
- <3> Set the count value to CR80
- <4> Set TM80 to operation enable (TCE80 = 1)

When the count value of 8-bit timer counter 80 (TM80) matches the value set to CR80, the value of TM80 is cleared to 0 and TM80 continue counting. At the same time, an interrupt request signal (INTTM80) is generated.

Tables 9-4 and 9-5 show the interval time, and Figure 9-4 shows the timing of interval timer operation.

- Cautions**
1. Before rewriting CR80, stop the timer operation once. If CR80 is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.
  2. If the count clock setting and TM80 operation-enabled are set in TMC80 simultaneously using an 8-bit memory manipulation instruction, an error of more than one clock in one cycle may occur after the timer starts.
- Therefore, always follow the above procedure when operating the 8-bit timer/event counter as an interval timer.

**Table 9-4. Interval Time of 8-Bit Timer/Event Counter 80 (At  $f_x = 5.0\text{-MHz}$  Operation)**

TCL801	TCL800	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	$1/f_x$ (200 ns)	$2^8/f_x$ (51.2 $\mu\text{s}$ )	$1/f_x$ (200 ns)
0	1	$2^3/f_x$ (1.6 $\mu\text{s}$ )	$2^{11}/f_x$ (409.6 $\mu\text{s}$ )	$2^3/f_x$ (1.6 $\mu\text{s}$ )
1	0	TI80 input cycle	$2^8 \times \text{TI80 input cycle}$	TI80 input edge cycle
1	1	TI80 input cycle	$2^8 \times \text{TI80 input cycle}$	TI80 input edge cycle

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

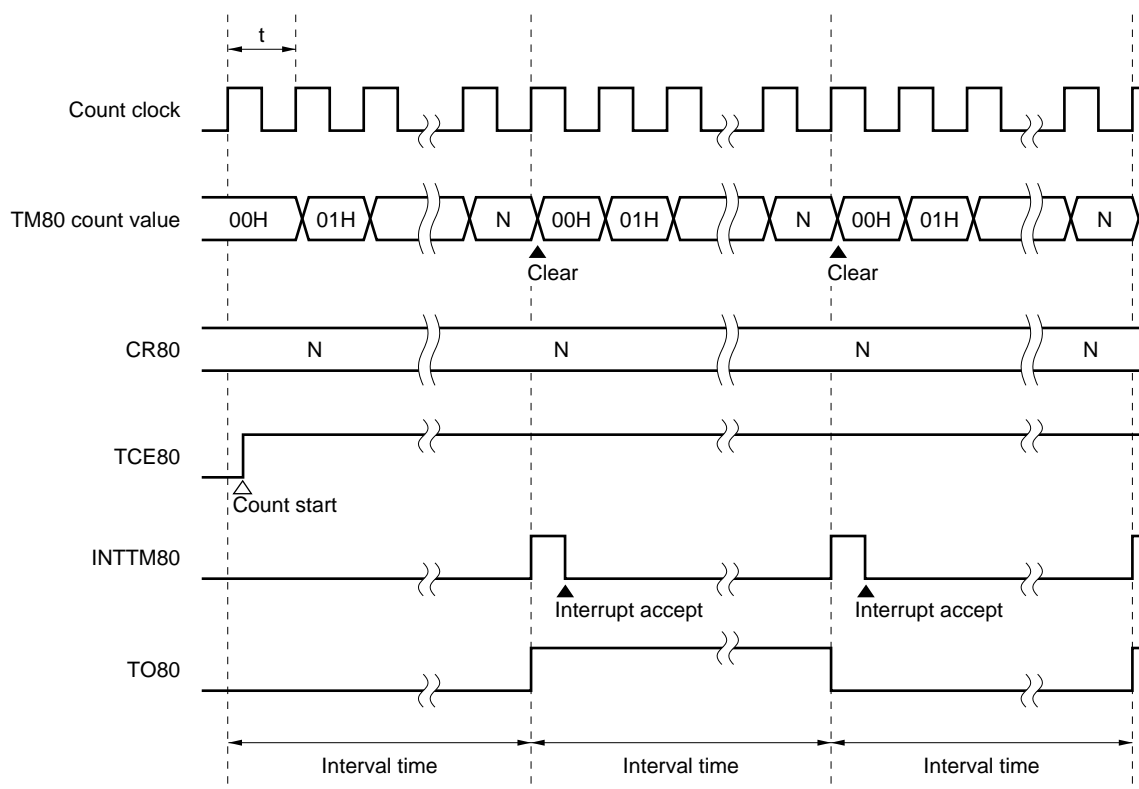
**Table 9-5. Interval Time of 8-Bit Timer/Event Counter 80 (At  $f_{cc} = 4.0\text{-MHz}$  Operation)**

TCL801	TCL800	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	$1/f_{cc}$ (250 ns)	$2^8/f_{cc}$ (64 $\mu\text{s}$ )	$1/f_{cc}$ (250 ns)
0	1	$2^3/f_{cc}$ (2.0 $\mu\text{s}$ )	$2^{11}/f_{cc}$ (512 $\mu\text{s}$ )	$2^3/f_{cc}$ (2.0 $\mu\text{s}$ )
1	0	TI80 input cycle	$2^8 \times \text{TI80 input cycle}$	TI80 input edge cycle
1	1	TI80 input cycle	$2^8 \times \text{TI80 input cycle}$	TI80 input edge cycle

**Remark**  $f_{cc}$ : System clock oscillation frequency (RC oscillation)



Figure 9-4. Interval Timer Operation Timing



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

### 9.4.2 Operation as external event counter

The external event counter counts the number of external clock pulses input to the TI80/P25/INTP2 pin by using 8-bit timer counter 80 (TM80).

To operate the 8-bit timer/event counter 80 as an external event counter, the following settings are required.

- <1> Set P25 to input mode (PM25 = 1)
- <2> Set 8-bit timer counter 80 (TM80) to operation disable (by setting TCE80 (bit 7 of 8-bit timer mode control register 80 (TMC80)) to 0).
- <3> Specify the rising/falling edges of TI80 (see **Tables 9-4** and **9-5**), and set TO80 to output disable (i.e. set TOE80 (bit 0 of TMC80) to 0) and PWM output to disable (i.e. set PWME80 (bit 6 of TMC80) to 0).
- <4> Set the count value to CR80.
- <5> Set TM80 to operation enable (TCE80 = 1)

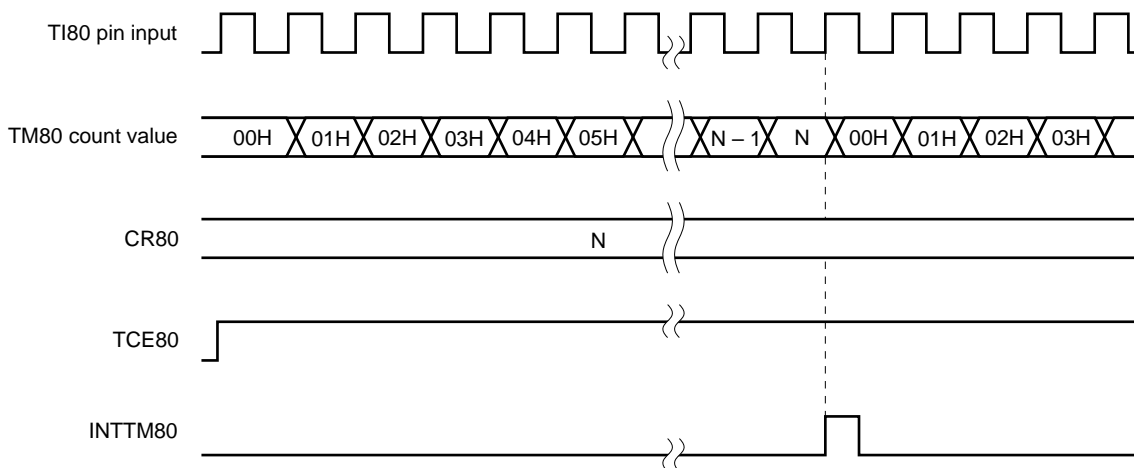
Each time the valid edge specified by bit 1 (TCL800) of TMC80 is input, the value of 8-bit timer counter 80 (TM80) is incremented.

When the count value of TM80 matches the value set to CR80, the value of TM80 is cleared to 0 and TM80 continues counting. At the same time, an interrupt request signal (INTTM80) is generated.

Figure 9-5 shows the timing of the external event counter operation (with rising edge specified).

- Cautions**
1. Before rewriting CR80, stop the timer operation once. If CR80 is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.
  2. If the count clock setting and TM80 operation-enabled are set in TMC80 simultaneously using an 8-bit memory manipulation instruction, an error of more than one clock in one cycle may occur after the timer starts.
- Therefore, always follow the above procedure when operating the 8-bit timer/event counter as an interval timer.

**Figure 9-5. External Event Counter Operation Timing (with Rising Edge Specified)**



**Remark** N = 00H to FFH

### 9.4.3 Operation as square wave output

The 8-bit timer/event counter can generate output square waves of a given frequency at intervals specified by the count value set to the 8-bit compare register 80 (CR80) in advance.

To operate the 8-bit timer/event counter 80 for square wave output, the following settings are required.

- <1> Set P24 to output mode (PM24 = 0) and the P24 output latch to 0.
- <2> Set 8-bit timer counter 80 (TM80) to operation disable (TCE80 = 0).
- <3> Set the count clock of the 8-bit timer/event counter 80 (see **Tables 9-4** and **9-5**), TO80 to output enable (TOE80 = 1), and PWM output to disable (PWME80 = 0).
- <4> Set the count value to CR80.
- <5> Set TM80 to operation enable (TCE80 = 1).

When the count value of 8-bit timer counter 80 (TM80) matches the value set in CR80, the TO80/P24/INTP1/TO20 pin output will be inverted. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, the TM80 value is cleared to 0 and TM80 continues counting. At the same time, an interrupt request signal (INTTM80) is generated.

Square wave output is cleared (0) when bit 7 (TCE80) in TMC80 is set to 0.

Table 9-6 shows square wave output range, and Figure 9-6 shows timing of square wave output.

- Cautions**
1. Before rewriting CR80, stop the timer operation once. If CR80 is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.
  2. If the count clock setting and TM80 operation-enabled are set in TMC80 simultaneously using an 8-bit memory manipulation instruction, an error of more than one clock in one cycle may occur after the timer starts.
- Therefore, always follow the above procedure when operating the 8-bit timer/event counter as an interval timer.

**Table 9-6. Square Wave Output Range of 8-Bit Timer/Event Counter 80 (At  $f_x = 5.0$ -MHz Operation)**

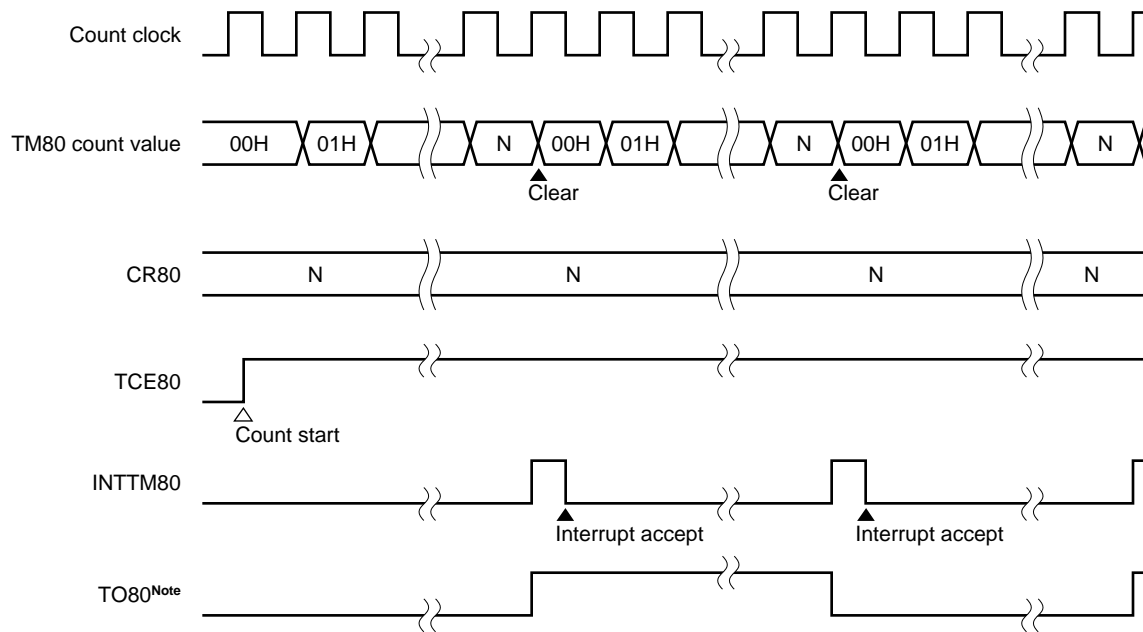
TCL801	TCL800	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	$1/f_x$ (200 ns)	$2^8/f_x$ (51.2 $\mu$ s)	$1/f_x$ (200 ns)
0	1	$2^3/f_x$ (1.6 $\mu$ s)	$2^{11}/f_x$ (409.6 $\mu$ s)	$2^3/f_x$ (1.6 $\mu$ s)

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

**Table 9-7. Square Wave Output Range of 8-Bit Timer/Event Counter 80 (At  $f_{cc} = 4.0$ -MHz Operation)**

TCL801	TCL800	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	$1/f_{cc}$ (250 ns)	$2^8/f_{cc}$ (64 $\mu$ s)	$1/f_{cc}$ (250 ns)
0	1	$2^3/f_{cc}$ (2.0 $\mu$ s)	$2^{11}/f_{cc}$ (512 $\mu$ s)	$2^3/f_{cc}$ (2.0 $\mu$ s)

**Remark**  $f_{cc}$ : System clock oscillation frequency (RC oscillation)

**Figure 9-6. Square Wave Output Timing**

**Note** The initial value of TO80 during output enable (TOE80 = 1) becomes low level.

#### 9.4.4 Operation as PWM output

PWM output enables interrupt generation repeatedly at intervals specified by the count value set to 8-bit compare register 80 (CR80) in advance.

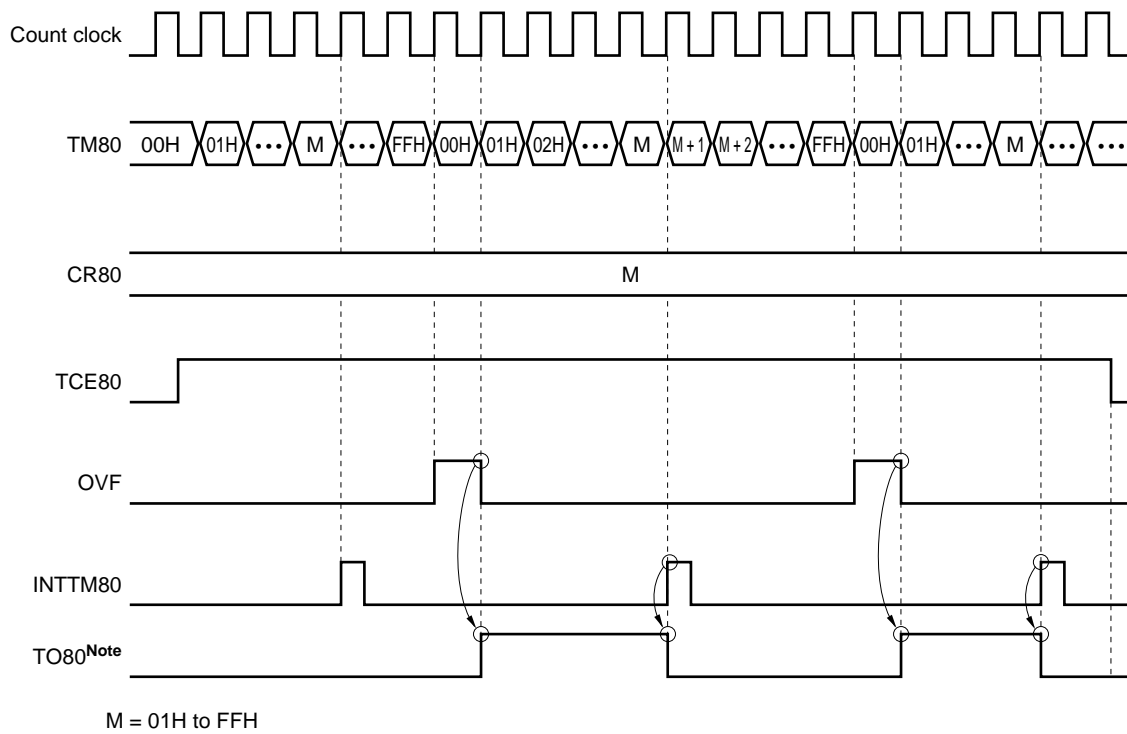
To use 8-bit timer/counter 80 for PWM output, the following settings are required.

- <1> Set P24 to output mode (PM24 = 0) and the P24 output latch to 0.
- <2> Set 8-bit timer counter 80 (TM80) to operation disable (TCE80 = 0).
- <3> Set the count clock of the 8-bit timer/event counter 80 (see **Tables 9-4** and **9-5**), TO80 to output enable (TOE80 = 1), and PWM output to enable (PWME80 = 1).
- <4> Set the count value to CR80
- <5> Set TM80 to operation enable (TCE80 = 1)

When the count value of 8-bit timer counter 80 (TM80) matches the value set to CR80, TM80 continues counting, and an interrupt request signal (INTTM80) is generated.

- Cautions**
1. When CR80 is rewritten during timer operation, a high level may be output for the next one cycle (refer to 9.5 (2) Setting of 8-bit compare register 80).
  2. If the count clock setting and TM80 operation-enabled are set in TMC80 simultaneously using an 8-bit memory manipulation instruction, an error of more than one clock in one cycle may occur after the timer starts. Therefore, always follow the above procedure when operating 8-bit compare register 80 as a PWM output.

Figure 9-7. PWM Output Timing



**Note** The initial value of TO80 upon output enable (TOE80 = 1) is low level.

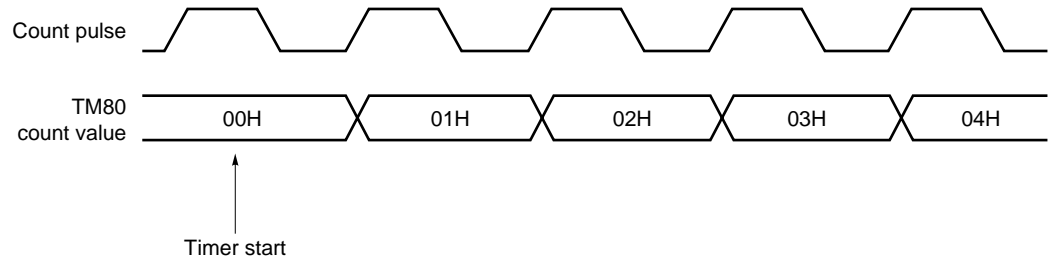
**Caution** Do not set CR80 to 00H in the PWM output mode; otherwise PWM may not be output normally.

## 9.5 Notes on Using 8-Bit Timer/Event Counter 80

### (1) Error on starting timer

An error of up to 1 clock occurs after the timer has been started until a coincidence signal is generated. This is because 8-bit timer counter 80 (TM80) is started in asynchronization with the count pulse.

**Figure 9-8. Start Timing of 8-Bit Timer Counter**

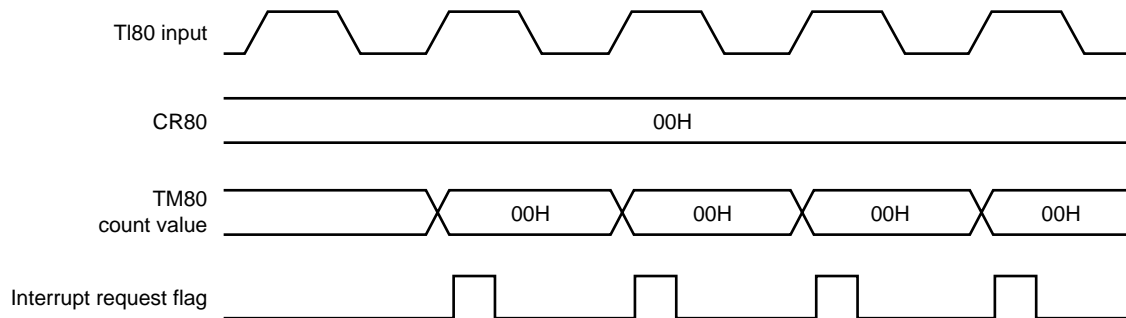


### (2) Setting of 8-bit compare register 80

8-bit compare register 80 (CR80) can be set to 00H.

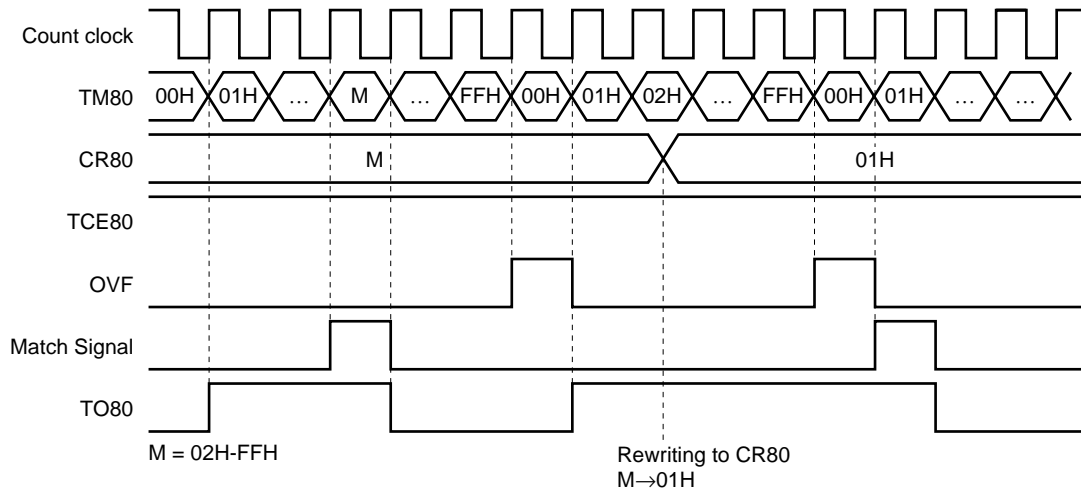
Therefore, one pulse can be counted when the 8-bit timer/event counter operates as an event counter.

**Figure 9-9. External Event Counter Operation Timing**



**Cautions** 1. When rewriting CR80 in timer counter operation mode (i.e. PWME80 (bit 6 of 8-bit timer mode control register 80 (TMC80) is set to 0), be sure to stop the timer operation before hand. If CR80 is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.

2. When rewriting CR80 in PWM output operation mode ( $PWME80 = 1$ ), a high level may be output for the next one cycle (count pulse x 256). This phenomenon occurs if a value smaller than the value of TM80 is written to CR80.



3. Do not set CR80 to 00H in the PWM output mode (when  $PWME80 = 1$ ); otherwise, PWM may not be output normally.



## CHAPTER 10 WATCHDOG TIMER

The watchdog timer can generate non-maskable interrupts, maskable interrupts and  $\overline{\text{RESET}}$  with arbitrary preset intervals.

### 10.1 Functions of Watchdog Timer

The watchdog timer has the following functions:

- Watchdog timer
- Interval timer

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

#### (1) Watchdog timer

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

**Table 10-1. Runaway Detection Time of Watchdog Timer**

Runaway Detection Time	At $f_x = 5.0\text{-MHz}$ Operation	At $f_{cc} = 4.0\text{-MHz}$ Operation
$2^{11} \times 1/f_w$	410 $\mu\text{s}$	512 $\mu\text{s}$
$2^{13} \times 1/f_w$	1.64 ms	2.05 ms
$2^{15} \times 1/f_w$	6.55 ms	8.19 ms
$2^{17} \times 1/f_w$	26.2 ms	32.8 ms

$f_w$ :  $f_x$  or  $f_{cc}$

$f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)

#### (2) Interval timer

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

**Table 10-2. Interval Time**

Interval Time	At $f_x = 5.0\text{-MHz}$ Operation	At $f_{cc} = 4.0\text{-MHz}$ Operation
$2^{11} \times 1/f_w$	410 $\mu\text{s}$	512 $\mu\text{s}$
$2^{13} \times 1/f_w$	1.64 ms	2.05 ms
$2^{15} \times 1/f_w$	6.55 ms	8.19 ms
$2^{17} \times 1/f_w$	26.2 ms	32.8 ms

$f_w$ :  $f_x$  or  $f_{cc}$

$f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)



### 10.3 Watchdog Timer Control Register

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

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

#### (1) Timer clock select register 2 (TCL2)

This register sets the watchdog timer count clock.

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

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

**Figure 10-2. Timer Clock Select Register 2 Format**

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

TCL22	TCL21	TCL20	Watchdog timer count clock selection		Interval time	
			At $f_x = 5.0\text{-MHz}$ operation	At $f_{cc} = 4.0\text{-MHz}$ operation	At $f_x = 5.0\text{-MHz}$ operation	At $f_{cc} = 4.0\text{-MHz}$ operation
0	0	0	$f_x/2^4$ (312.5 kHz)	$f_{cc}/2^4$ (250 KHZ)	$2^{11}/f_x$ (410 $\mu\text{s}$ )	$2^{11}/f_{cc}$ (512 $\mu\text{s}$ )
0	1	0	$f_x/2^6$ (78.1 kHz)	$f_{cc}/2^6$ (62.5 KHZ)	$2^{13}/f_x$ (1.64 ms)	$2^{13}/f_{cc}$ (2.05 ms)
1	0	0	$f_x/2^8$ (19.5 kHz)	$f_{cc}/2^8$ (15.6 KHZ)	$2^{15}/f_x$ (6.55 ms)	$2^{15}/f_{cc}$ (8.19 ms)
1	1	0	$f_x/2^{10}$ (4.88 kHz)	$f_{cc}/2^{10}$ (3.91 KHZ)	$2^{17}/f_x$ (26.2 ms)	$2^{17}/f_{cc}$ (32.8 ms)
Other than above			Setting prohibited			

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)  
 $f_{cc}$ : System clock oscillation frequency (RC oscillation)

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

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

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

RESET input clears WDTM to 00H.

**Figure 10-3. 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	Selects operation of watchdog timer <sup>Note 1</sup>
0	Stops counting
1	Clears counter and starts counting

WDTM4	WDTM3	Selects operation mode of watchdog timer <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 (1), it cannot be cleared (0) by software. Therefore, when counting is started, it cannot be stopped by any means other than RESET input.
  2. Once WDTM3 and WDTM4 have been set (1), they cannot be cleared (0) by software.
  3. The watchdog timer starts operations as an interval timer when RUN is set to 1.

- Cautions**
1. When the watchdog timer is cleared by setting 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 TMIF4 (bit 0 of interrupt request flag 0) has been 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.

## 10.4 Operation of Watchdog Timer

### 10.4.1 Operation as watchdog timer

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

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

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

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

**Table 10-4. Runaway Detection Time of Watchdog Timer**

TCL22	TCL21	TCL20	Runaway Detection Time	At $f_x = 5.0\text{-MHz}$ Operation	At $f_{cc} = 4.0\text{-MHz}$ Operation
0	0	0	$2^{11} \times 1/f_w$	410 $\mu\text{s}$	512 $\mu\text{s}$
0	1	0	$2^{13} \times 1/f_w$	1.64 ms	2.05 ms
1	0	0	$2^{15} \times 1/f_w$	6.55 ms	8.19 ms
1	1	0	$2^{17} \times 1/f_w$	26.2 ms	32.8 ms

$f_w$ :  $f_x$  or  $f_{cc}$

$f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)

### 10.4.2 Operation as interval timer

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

Select a count clock (or interval time) by setting bits 0 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 10-5. Interval Time of Interval Timer**

TCL22	TCL21	TCL20	Interval Time	At $f_x = 5.0\text{-MHz}$ Operation	At $f_{cc} = 4.0\text{-MHz}$ Operation
0	0	0	$2^{11} \times 1/f_w$	410 $\mu\text{s}$	512 $\mu\text{s}$
0	1	0	$2^{13} \times 1/f_w$	1.64 ms	2.05 ms
1	0	0	$2^{15} \times 1/f_w$	6.55 ms	8.19 ms
1	1	0	$2^{17} \times 1/f_w$	26.2 ms	32.8 ms

$f_w$ :  $f_x$  or  $f_{cc}$

$f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)

## CHAPTER 11 8-BIT A/D CONVERTER ( $\mu$ PD789104A, 789124A SUBSERIES)

### 11.1 8-Bit A/D Converter Functions

The 8-bit A/D converter is an 8-bit resolution converter to convert analog input to digital signals. This converter can control up to four channels of analog inputs (ANI0 to ANI3).

A/D conversion can only be started by software.

One of analog inputs ANI0 to ANI3 is selected for A/D conversion. A/D conversion is performed repeatedly, with an interrupt request (INTAD0) being issued each time an A/D session is completed.

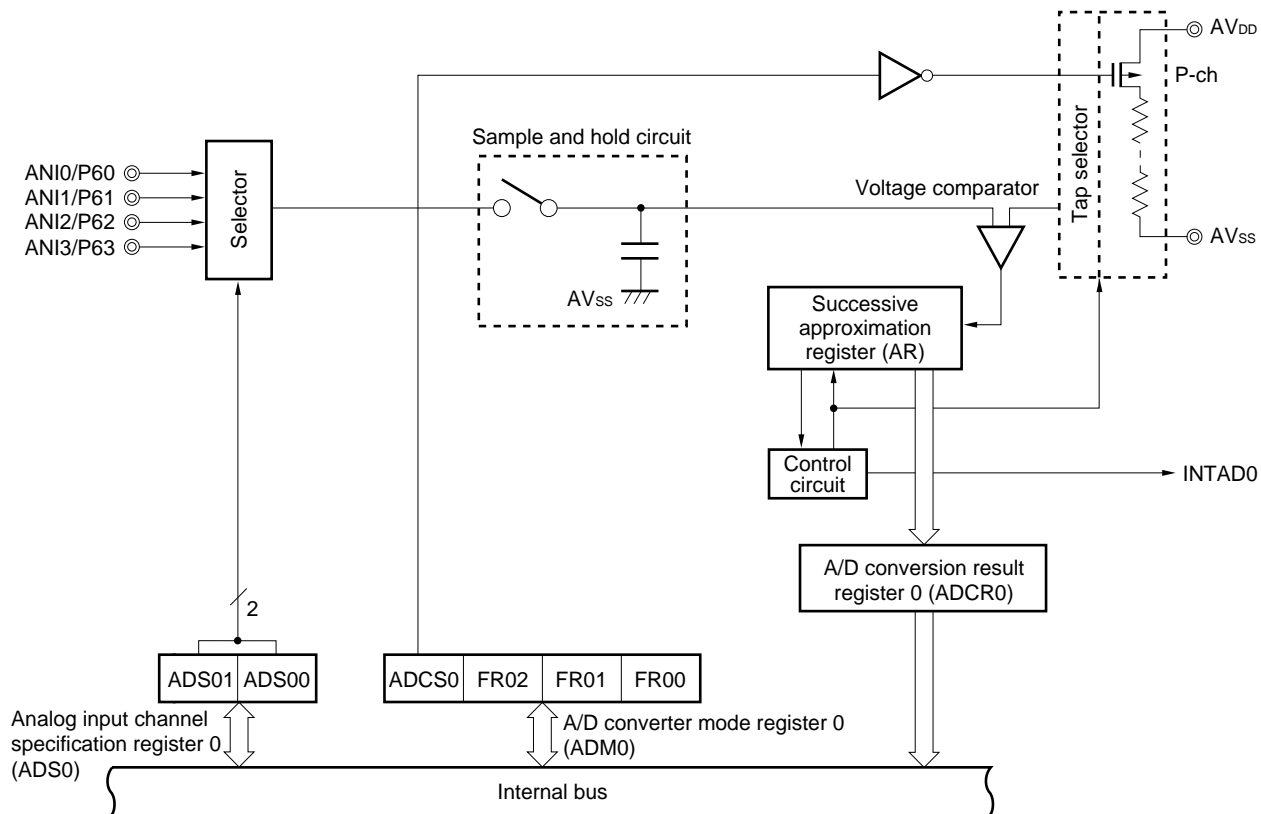
### 11.2 8-Bit A/D Converter Configuration

The 8-bit A/D converter consists of the following hardware.

**Table 11-1. Configuration of 8-Bit A/D Converter**

Item	Configuration
Analog input	4 channels (ANI0 to ANI3)
Register	Successive approximation register (SAR) A/D conversion result register 0 (ADCR0)
Control register	A/D converter mode register 0 (ADM0) Analog input channel specification register 0 (ADS0)

Figure 11-1. Block Diagram of 8-Bit A/D Converter

**(1) Successive approximation register (SAR)**

The SAR receives the result of comparing an analog input voltage and a voltage at a voltage tap (comparison voltage), received from the series resistor string, starting from the most significant bit (MSB).

Upon receiving all the bits, down to the least significant bit (LSB), that is, upon the completion of A/D conversion, the SAR sends its contents to A/D conversion result register 0 (ADCR0).

**(2) A/D conversion result register 0 (ADCR0)**

ADCR0 holds the result of A/D conversion. Each time A/D conversion ends, the conversion result received from the successive approximation register is loaded into ADCR0, which is an 8-bit register that holds the result of A/D conversion.

ADCR0 can be read with an 8-bit memory manipulation instruction.

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

**(3) Sample and hold circuit**

The sample and hold circuit samples consecutive analog inputs from the input circuit, one by one, and sends them to the voltage comparator. The sampled analog input voltage is held during A/D conversion.

**(4) Voltage comparator**

The voltage comparator compares an analog input with the voltage output by the series resistor string.



**(5) Series resistor string**

The series resistor string is configured between  $AV_{DD}$  and  $AV_{SS}$ . It generates the reference voltages against which analog inputs are compared.

**(6) ANI0 to ANI3 pins**

Pins ANI0 to ANI3 are the 4-channel analog input pins for the A/D converter. They are used to receive the analog signals for A/D conversion.

**Caution** Do not supply pins ANI0 to ANI3 with voltages that fall outside the rated range. If a voltage greater than  $AV_{DD}$  or less than  $AV_{SS}$  (even if within the absolute maximum rating) is supplied to any of these pins, the conversion value for the corresponding channel will be undefined. Furthermore, the conversion values for the other channels may also be affected.

**(7)  $AV_{SS}$  pin**

The  $AV_{SS}$  pin is a ground potential pin for the A/D converter. This pin must be held at the same potential as the  $V_{SS}$  pin, even while the A/D converter is not being used.

**(8)  $AV_{DD}$  pin**

The  $AV_{DD}$  pin is an analog power supply pin for the A/D converter. This pin must be held at the same potential as the  $V_{DD}$  pin, even while the A/D converter is not being used.

### 11.3 Registers Controlling 8-Bit A/D Converter

The following two registers are used to control the 8-bit A/D converter.

- A/D converter mode register 0 (ADM0)
- Analog input channel specification register 0 (ADS0)

#### (1) A/D converter mode register 0 (ADM0)

ADM0 specifies the conversion time for analog inputs. It also specifies whether to enable conversion.

ADM0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears ADM0 to 00H.

**Figure 11-2. Format of A/D Converter Mode Register 0**

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
ADM0	ADCS0	0	FR02	FR01	FR00	0	0	0	FF80H	00H	R/W

ADCS0	A/D conversion control
0	Conversion disabled
1	Conversion enabled

FR02	FR01	FR00	A/D conversion time selection <sup>Note 1</sup>	
			At $f_x = 5.0\text{-MHz}$ operation	At $f_{cc} = 4.0\text{-MHz}$ operation
0	0	0	$144/f_x$ (28.8 $\mu\text{s}$ )	$144/f_{cc}$ (36 $\mu\text{s}$ )
0	0	1	$120/f_x$ (24 $\mu\text{s}$ )	$120/f_{cc}$ (30 $\mu\text{s}$ )
0	1	0	$96/f_x$ (19.2 $\mu\text{s}$ )	$96/f_{cc}$ (24 $\mu\text{s}$ )
1	0	0	$72/f_x$ (14.4 $\mu\text{s}$ )	$72/f_{cc}$ (18 $\mu\text{s}$ )
1	0	1	$60/f_x$ (Setting prohibited <sup>Note 2</sup> )	$60/f_{cc}$ (15 $\mu\text{s}$ )
1	1	0	$48/f_x$ (Setting prohibited <sup>Note 2</sup> )	$48/f_{cc}$ (Setting prohibited <sup>Note 2</sup> )
Other than above			Setting prohibited	

- Notes**
1. The specifications of FR02, FR01, and FR00 must be such that the A/D conversion time is at least 14  $\mu\text{s}$ .
  2. These bit combinations must not be used, as the A/D conversion time will fall below 14  $\mu\text{s}$ .

- Cautions**
1. The result of conversion performed immediately after bit 7 (ADCS0) is set is undefined.
  2. The result of conversion after ADCS0 is cleared may be undefined (for details, refer to 11.5 (5) Timing when A/D conversion result become undefined).

**Remark**

$f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)

**(2) Analog input channel specification register 0 (ADS0)**

The ADS0 register specifies the port used to input the analog voltages to be converted to a digital signal.

ADS0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears ADS0 to 00H.

**Figure 11-3. Format of Analog Input Channel Specification Register 0**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADS0	0	0	0	0	0	0	ADS01	ADS00	FF84H	00H	R/W

ADS01	ADS00	Analog input channel specification
0	0	ANI0
0	1	ANI1
1	0	ANI2
1	1	ANI3

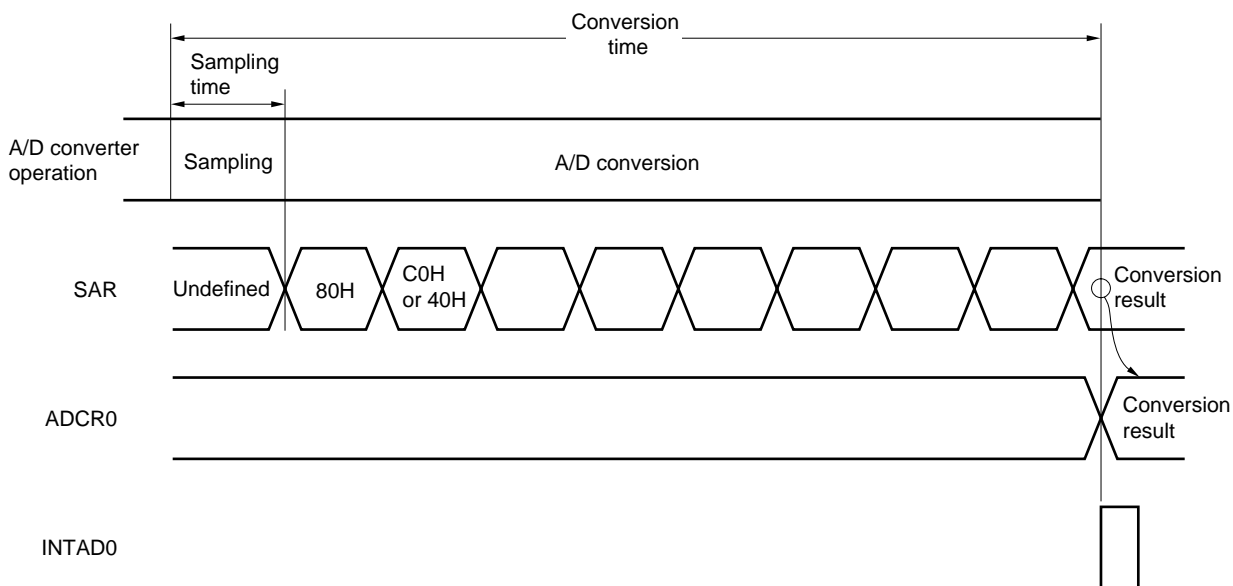
## 11.4 8-Bit A/D Converter Operation

### 11.4.1 Basic operation of 8-bit A/D converter

- <1> Select a channel for A/D conversion, using **Analog input channel specification register 0 (ADS0)**.
- <2> The voltage supplied to the selected analog input channel is sampled using the sample and hold circuit.
- <3> After sampling continues for a certain period of time, the sample and hold circuit is put on hold to keep the input analog voltage until A/D conversion is completed.
- <4> Bit 7 of the successive approximation A/D conversion register (SAR) is set. The series resistor string voltage tap at the tap selector is set to half of  $AV_{DD}$ .
- <5> The series resistor string tap voltage is compared with the analog input voltage using the voltage comparator. If the analog input voltage is higher than half of  $AV_{DD}$ , the MSB of the SAR is left set. If it is lower than half of  $AV_{DD}$ , the MSB is reset.
- <6> Bit 6 of the SAR is set automatically, and comparison shifts to the next stage. The next voltage tap of the series resistor string is selected according to bit 7, which reflects the previous comparison result, as follows:
  - Bit 7 = 1: Three quarters of  $AV_{DD}$
  - Bit 7 = 0: One quarter of  $AV_{DD}$
 The tap voltage is compared with the analog input voltage. Bit 6 is set or reset according to the result of comparison.
  - Analog input voltage  $\geq$  tap voltage: Bit 6 = 1
  - Analog input voltage < tap voltage: Bit 6 = 0
- <7> Comparison is repeated until bit 0 of the SAR is reached.
- <8> When comparison is completed for all of the 8 bits, a significant digital result is left in the SAR. This value is sent to and latched in A/D conversion result register 0 (ADCR0). At the same time, it is possible to generate an A/D conversion end interrupt request (INTAD0).

- Cautions**
1. The first A/D conversion value immediately after starting the A/D conversion operation may be undefined.
  2. When in standby mode, the A/D converter stops operation.

Figure 11-4. Basic Operation of 8-Bit A/D Converter



A/D conversion continues until bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) is reset (0) by software.

If an attempt is made to write to ADM0 or Analog input channel specification register 0 (ADS0) during A/D conversion, the ongoing A/D conversion is canceled. In this case, if ADCS0 is set (1), A/D conversion is restarted from the beginning.

$\overline{\text{RESET}}$  input makes the A/D conversion result register 0 (ADCR0) undefined.

#### 11.4.2 Input voltage and conversion result

The relationships between the analog input voltage at the analog input pins (ANI0 to ANI3) and the A/D conversion result (A/D conversion result register 0 (ADCR0)) are represented by:

$$\text{ADCR0} = \text{INT} \left( \frac{V_{\text{IN}}}{AV_{\text{DD}}} \times 256 + 0.5 \right)$$

or

$$(\text{ADCR0} - 0.5) \times \frac{AV_{\text{DD}}}{256} \leq V_{\text{IN}} < (\text{ADCR0} + 0.5) \times \frac{AV_{\text{DD}}}{256}$$

INT( ): Function that returns the integer part of a parenthesized value

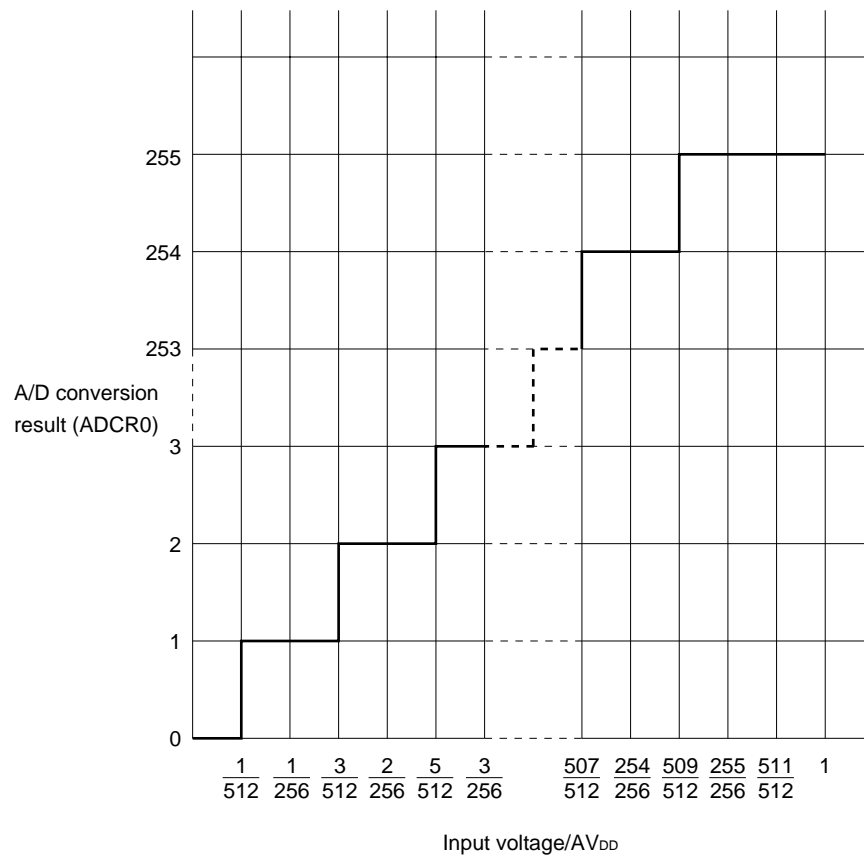
$V_{\text{IN}}$ : Analog input voltage

$AV_{\text{DD}}$ : A/D converter supply voltage

ADCR0: Value in the A/D conversion result register 0 (ADCR0)

Figure 11-5 shows the relationships between the analog input voltage and the A/D conversion result.

Figure 11-5. Relationships between Analog Input Voltage and A/D Conversion Result



### 11.4.3 Operation mode of 8-bit A/D converter

The 8-bit A/D converter is initially in the select mode. In this mode, Analog input channel specification register 0 (ADS0) is used to select an analog input channel from ANI0 to ANI3 for A/D conversion.

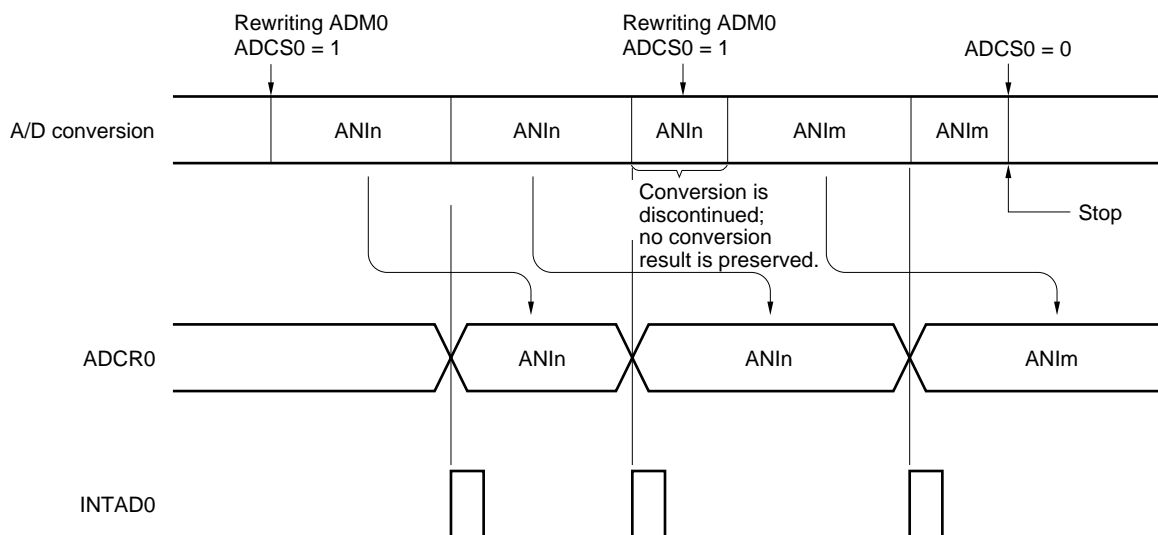
A/D conversion can only be started by software, that is, by setting A/D converter mode register 0 (ADM0).

The A/D conversion result is saved to A/D conversion result register 0 (ADCR0). At the same time, an interrupt request signal (INTAD0) is generated.

#### • Software-started A/D conversion

Setting bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) triggers A/D conversion for the voltage applied to the analog input pin specified in Analog input channel specification register 0 (ADS0). Upon completion of A/D conversion, the conversion result is saved to A/D conversion result register 0 (ADCR0). At the same time, an interrupt request signal (INTAD0) is generated. Once A/D conversion is activated, and completed, another session of A/D conversion is started. A/D conversion is repeated until new data is written to ADM0. If data where ADCS0 is 1 is written to ADM0 again during A/D conversion, the ongoing session of A/D conversion is discontinued, and a new session of A/D conversion begins for the new data. If data where ADCS0 is 0 is written to the ADM0 again during A/D conversion, A/D conversion is completely stopped.

**Figure 11-6. Software-Started A/D Conversion**



- Remarks**
1.  $n = 0, 1, 2, 3$
  2.  $m = 0, 1, 2, 3$

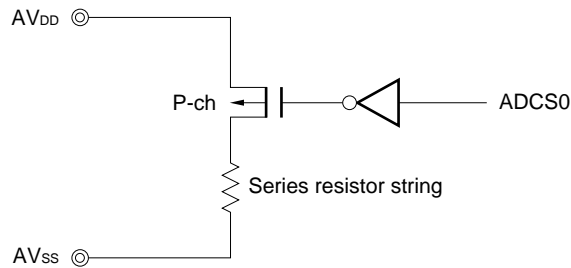
## 11.5 Cautions Related to 8-Bit A/D Converter

### (1) Current consumption in the standby mode

When the A/D converter enters the standby mode, it stops its operation. Stopping conversion (bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) = 0) can reduce the current consumption.

Figure 11-7 shows how to reduce the current consumption in the standby mode.

**Figure 11-7. How to Reduce Current Consumption in Standby Mode**



### (2) Input range for the ANI0 to ANI3 pins

Be sure to keep the input voltage at ANI0 to ANI3 within its rating. If a voltage not lower than AVDD or not higher than AVSS (even within the absolute maximum rating) is input to a conversion channel, the conversion output of the channel becomes undefined. It may affect the conversion output of the other channels.

### (3) Conflict

**<1> Conflict between writing to A/D conversion result register 0 (ADCR0) at the end of conversion and reading from ADCR0**

Reading from ADCR0 takes precedence. After reading, the new conversion result is written to the ADCR0.

**<2> Conflict between writing to ADCR0 at the end of conversion and writing to A/D converter mode register 0 (ADM0) or Analog input channel specification register 0 (ADS0)**

Writing to ADM0 or ADS0 takes precedence. A request to write to ADCR0 is ignored. No conversion end interrupt request signal (INTAD0) is generated.

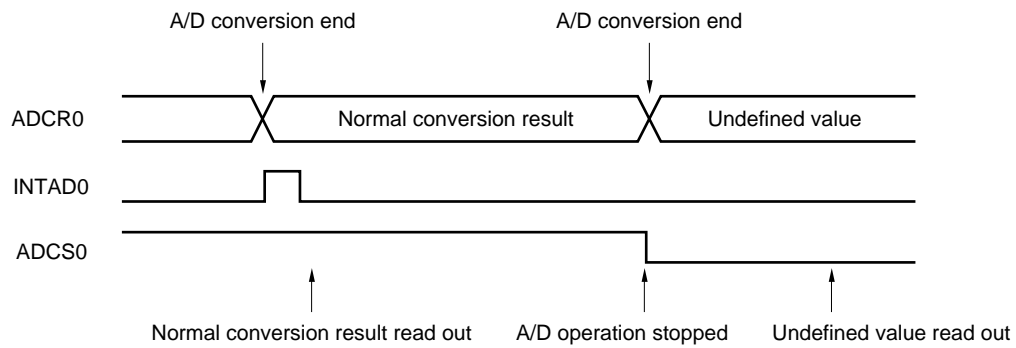
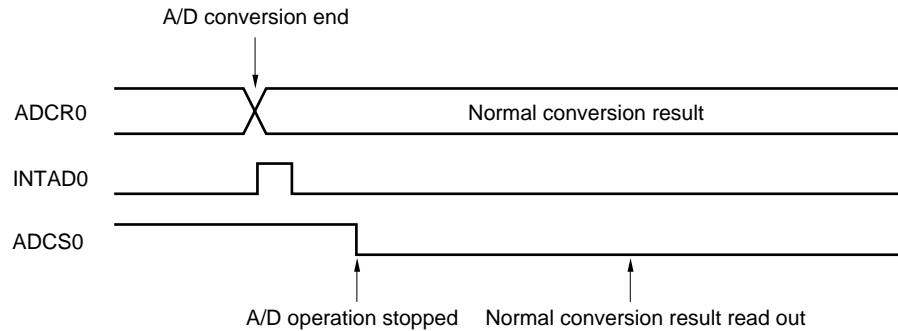


**(4) Conversion results immediately following start of A/D conversion**

The first A/D conversion value immediately following the start of A/D converter operation may be undefined. Be sure to poll the A/D conversion end interrupt request (INTAD0) and perform processing such as annulling the first conversion result.

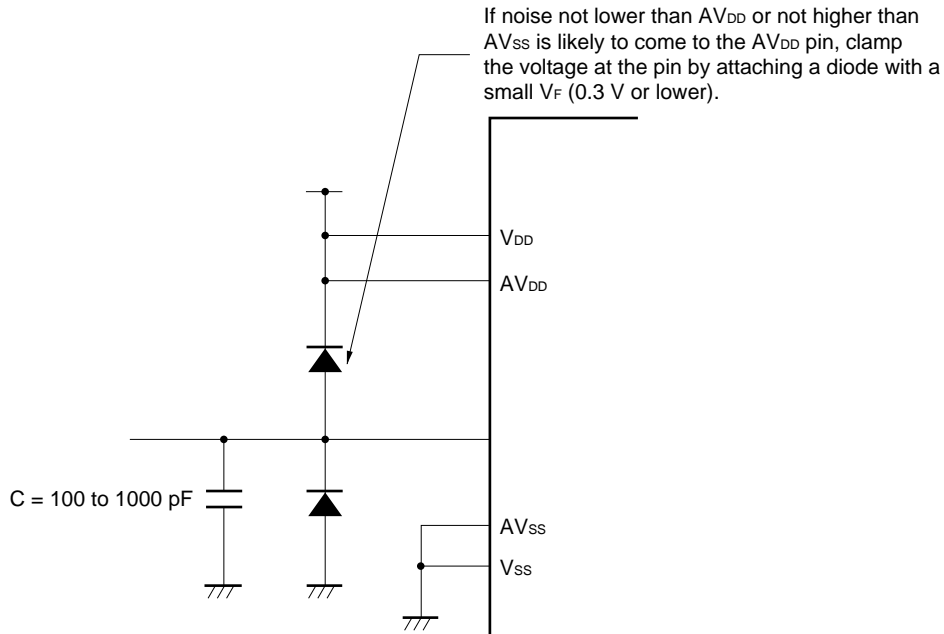
**(5) Timing that makes the A/D conversion result undefined**

If the timing of the end of A/D conversion and the timing of the stop of operation of the A/C converter conflict, the A/D conversion value may be undefined. Because of this, be sure to read out the A/D conversion result while the A/D converter is in operation. Furthermore, when reading out an A/D conversion result after A/D converter operation has stopped, be sure to have done so by the time the next conversion result is complete. The conversion result readout timing is shown in Figures 11-8 and 11-9.

**Figure 11-8. Conversion Result Readout Timing (When Conversion Result Is Undefined Value)****Figure 11-9. Conversion Result Readout Timing (When Conversion Result Is Normal Value)**

**(6) Noise prevention**

To maintain a resolution of 8 bits, watch for noise to the  $AV_{DD}$  and  $ANI0$  to  $ANI3$  pins. The higher the output impedance of the analog input source is, the larger the effect by noise. To reduce noise, attach an external capacitor to the relevant pins as shown in Figure 11-8.

**Figure 11-10. Analog Input Pin Treatment****(7)  $ANI0$  to  $ANI3$** 

The analog input pins ( $ANI0$  to  $ANI3$ ) are alternate-function pins. They are used also as port pins ( $P60$  to  $P63$ ).

If any of  $ANI0$  to  $ANI3$  has been selected for A/D conversion, do not execute input instructions for the ports; otherwise the conversion resolution may become lower.

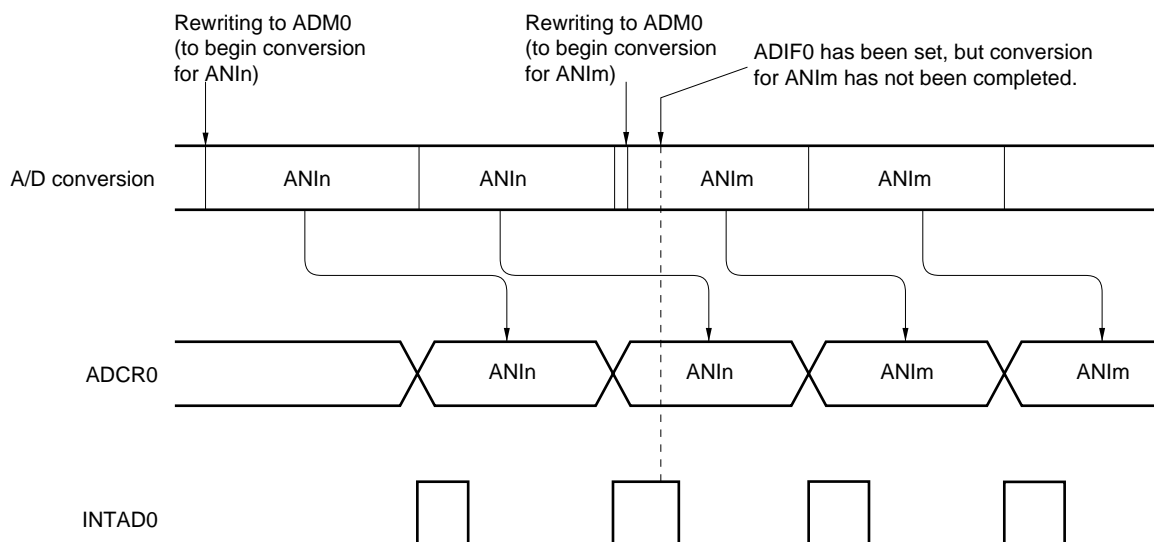
If a digital pulse is applied to a pin adjacent to the analog input pins during A/D conversion, coupling noise may occur which prevents an A/D conversion result from being attained as expected. Avoid applying a digital pulse to pins adjacent to the analog input pins during A/D conversion.

**(8) Interrupt request flag (ADIF0)**

Changing the content of A/D converter mode register 0 (ADM0) does not clear the interrupt request flag (ADIF0).

If the analog input pins are changed during A/D conversion, therefore, the conversion result and the conversion end interrupt request flag may reflect the previous analog input immediately before writing to ADM0 occurs. In this case, ADIF0 may appear to be set if it is read-accessed immediately after ADM0 is write-accessed, even when A/D conversion has not been completed for the new analog input.

In addition, when A/D conversion is restarted, ADIF0 must be cleared beforehand.

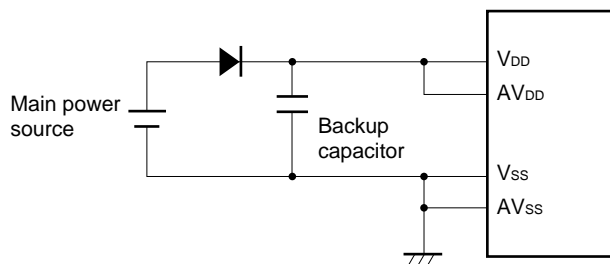
**Figure 11-11. A/D Conversion End Interrupt Request Generation Timing**

- Remarks**
1.  $n = 0, 1, 2, 3$
  2.  $m = 0, 1, 2, 3$

**(9) AV<sub>DD</sub> pin**

The AV<sub>DD</sub> pin is used to supply power to the analog circuit. It is also used to supply power to the ANI0 to ANI3 input circuit.

Therefore, if the application is designed to be switched to backup power, the AV<sub>DD</sub> pin must be supplied with the same voltage level as for the V<sub>DD</sub> pin, as shown in Figure 11-12.

**Figure 11-12. AV<sub>DD</sub> Pin Treatment****(10) Input impedance of the AV<sub>DD</sub> pin**

A series resistor string of several 10 k $\Omega$  is connected across the AV<sub>DD</sub> and AV<sub>SS</sub> pins.

Therefore, if the output impedance of the reference voltage source is high, this high impedance is eventually connected in parallel with the series resistor string across the AV<sub>DD</sub> and AV<sub>SS</sub> pins, leading to a higher reference voltage error.

**[MEMO]**

## CHAPTER 12 10-BIT A/D CONVERTER ( $\mu$ PD789114A, 789134A SUBSERIES)

### 12.1 10-Bit A/D Converter Functions

The 10-bit A/D converter is a 10-bit resolution converter to convert an analog input to digital signals. This converter can control up to four channels of analog inputs (ANI0 to ANI3).

A/D conversion can only be started by software.

One of analog inputs ANI0 to ANI3 is selected for A/D conversion. A/D conversion is performed repeatedly, with an interrupt request (INTAD0) being issued each time an A/D session is completed.

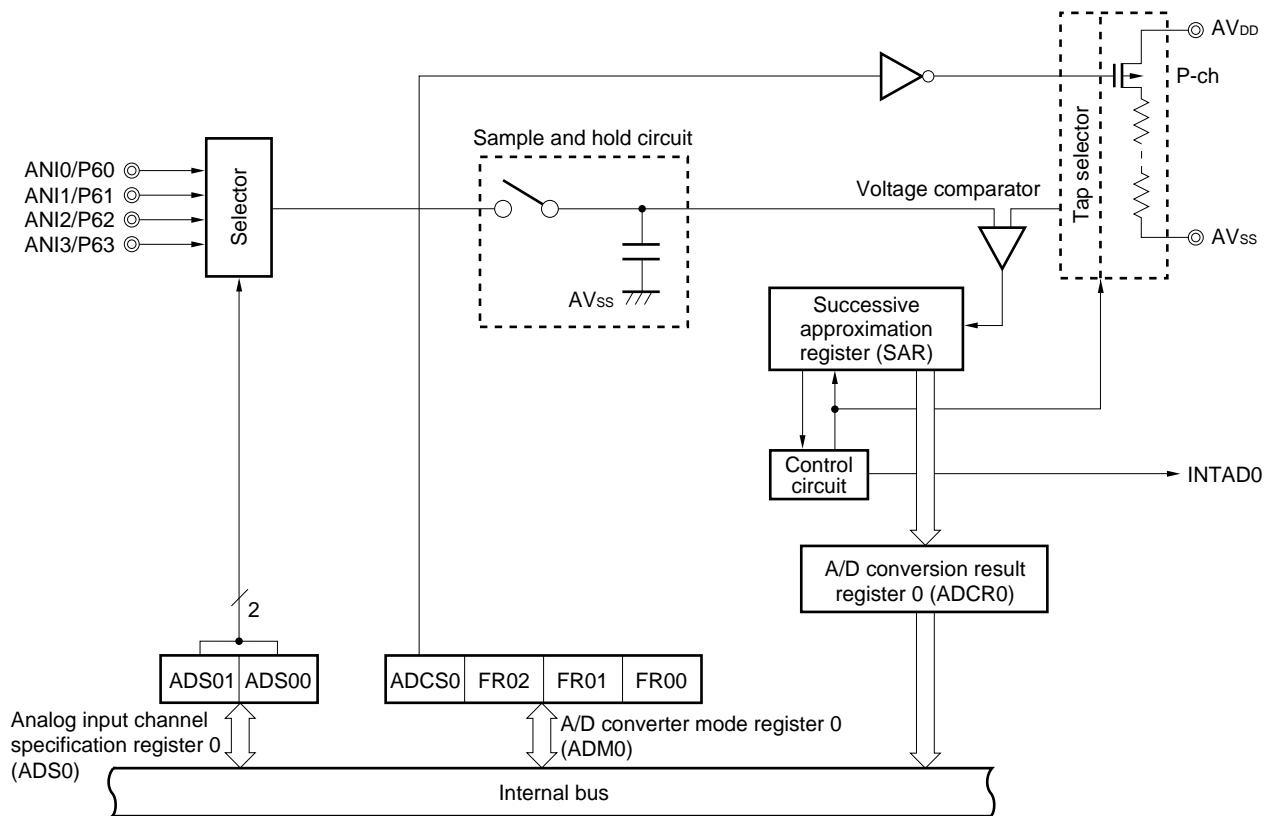
### 12.2 10-Bit A/D Converter Configuration

The A/D converter consists of the following hardware.

**Table 12-1. Configuration of 10-Bit A/D Converter**

Item	Configuration
Analog input	4 channels (ANI0 to ANI3)
Register	Successive approximation register (SAR) A/D conversion result register 0 (ADCR0)
Control register	A/D converter mode register 0 (ADM0) Analog input channel specification register 0 (ADS0)

Figure 12-1. Block Diagram of 10-Bit A/D Converter

**(1) Successive approximation register (SAR)**

The SAR receives the result of comparing an analog input voltage and a voltage at a voltage tap (comparison voltage), received from the series resistor string, starting from the most significant bit (MSB).

Upon receiving all the bits, down to the least significant bit (LSB), that is, upon the completion of A/D conversion, the SAR sends its contents to A/D conversion result register 0 (ADCR0).

**(2) A/D conversion result register 0 (ADCR0)**

ADCR0 holds the result of A/D conversion. Each time A/D conversion ends, the conversion result received from the successive approximation register is loaded into ADCR0, which is a 10-bit register that holds the result of A/D conversion.

ADCR0 can be read with an 8-bit memory manipulation instruction.

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

**Caution** When using the  $\mu$ PD78F9116A as a flash memory version of the  $\mu$ PD789101A, 789102A, or 789104A, or the  $\mu$ PD78F9136A as a flash memory version of the  $\mu$ PD789121A, 789122A, or 789124A, an 8-bit access can be made by ADCR0. However, it is performed only with the object file assembled by the  $\mu$ PD789101A, 789102A, or 789104A, or by the  $\mu$ PD789121A, 789122A, or 789124A, respectively.

**(3) Sample and hold circuit**

The sample-and-hold circuit samples consecutive analog inputs from the input circuit, one by one, and sends them to the voltage comparator. The sampled analog input voltage is held during A/D conversion.

**(4) Voltage comparator**

The voltage comparator compares an analog input with the voltage output by the series resistor string.

**(5) Series resistor string**

The series resistor string is configured between  $AV_{DD}$  and  $AV_{SS}$ . It generates the reference voltages against which analog inputs are compared.

**(6) ANI0 to ANI3 pins**

Pins ANI0 to ANI3 are the 4-channel analog input pins for the A/D converter. They are used to receive the analog signals for A/D conversion.

**Caution** Do not supply pins ANI0 to ANI3 with voltages that fall outside the rated range. If a voltage greater than  $AV_{DD}$  or less than  $AV_{SS}$  (even if within the absolute maximum rating) is supplied to any of these pins, the conversion value for the corresponding channel will be undefined. Furthermore, the conversion values for the other channels may also be affected.

**(7)  $AV_{SS}$  pin**

The  $AV_{SS}$  pin is a ground potential pin for the A/D converter. This pin must be held at the same potential as the  $V_{SS}$  pin, even while the A/D converter is not being used.

**(8)  $AV_{DD}$  pin**

The  $AV_{DD}$  pin is an analog power supply pin for the A/D converter. This pin must be held at the same potential as the  $V_{DD}$  pin, even while the A/D converter is not being used.

### 12.3 Registers Controlling 10-Bit A/D Converter

The following two registers are used to control the 10-bit A/D converter.

- A/D converter mode register 0 (ADM0)
- Analog input channel specification register 0 (ADS0)

#### (1) A/D converter mode register 0 (ADM0)

ADM0 specifies the conversion time for analog inputs. It also specifies whether to enable conversion.

ADM0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears the ADM0 to 00H.

**Figure 12-2. Format of A/D Converter Mode Register 0**

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
ADM0	ADCS0	0	FR02	FR01	FR00	0	0	0	FF80H	00H	R/W

ADCS0	A/D conversion control
0	Conversion disabled
1	Conversion enabled

FR02	FR01	FR00	A/D conversion time selection <sup>Note 1</sup>	
			At $f_x = 5.0\text{-MHz}$ operation	At $f_{cc} = 4.0\text{-MHz}$ operation
0	0	0	$144/f_x$ (28.8 $\mu\text{s}$ )	$144/f_{cc}$ (36 $\mu\text{s}$ )
0	0	1	$120/f_x$ (24 $\mu\text{s}$ )	$120/f_{cc}$ (30 $\mu\text{s}$ )
0	1	0	$96/f_x$ (19.2 $\mu\text{s}$ )	$96/f_{cc}$ (24 $\mu\text{s}$ )
1	0	0	$72/f_x$ (14.4 $\mu\text{s}$ )	$72/f_{cc}$ (18 $\mu\text{s}$ )
1	0	1	$60/f_x$ (Setting prohibited <sup>Note 2</sup> )	$60/f_{cc}$ (15 $\mu\text{s}$ )
1	1	0	$48/f_x$ (Setting prohibited <sup>Note 2</sup> )	$48/f_{cc}$ (Setting prohibited <sup>Note 2</sup> )
Other than above			Setting prohibited	

**Notes** 1. The specifications of FR02, FR01, and FR00 must be such that the A/D conversion time is at least 14  $\mu\text{s}$ .

2. These bit combinations must not be used, as the A/D conversion time will fall below 14  $\mu\text{s}$ .

**Cautions** 1. The result of conversion performed immediately after bit 7 (ADCS0) is set is undefined.

2. The result of conversion after ADCS0 is cleared may be undefined (for details, refer to 12.5 (5) Timing when A/D conversion result becomes undefined).

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$f_{cc}$ : System clock oscillation frequency (RC oscillation)



**(2) Analog input channel specification register 0 (ADS0)**

The ADS0 register specifies the port used to input the analog voltages to be converted to a digital signal. ADS0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears ADS0 to 00H.

**Figure 12-3. Format of Analog Input Channel Specification Register 0**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADS0	0	0	0	0	0	0	ADS01	ADS00	FF84H	00H	R/W

ADS01	ADS00	Analog input channel specification
0	0	ANI0
0	1	ANI1
1	0	ANI2
1	1	ANI3

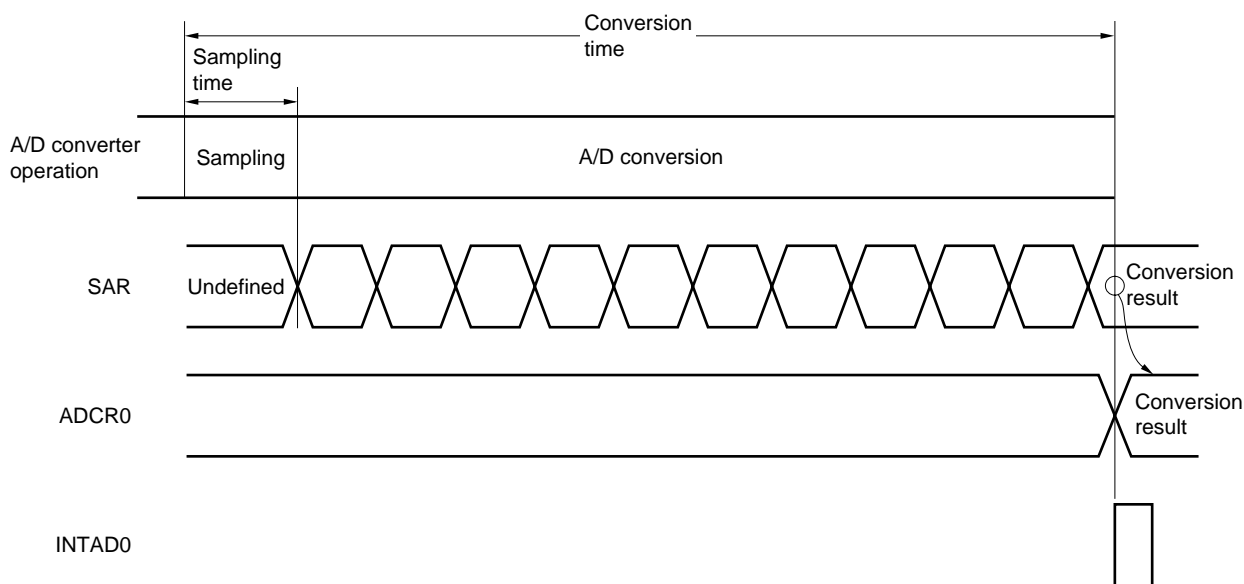
## 12.4 10-Bit A/D Converter Operation

### 12.4.1 Basic operation of 10-bit A/D converter

- <1> Select a channel for A/D conversion, using Analog input channel specification register 0 (ADS0).
- <2> The voltage supplied to the selected analog input channel is sampled using the sample and hold circuit.
- <3> After sampling continues for a certain period of time, the sample and hold circuit is put on hold to keep the input analog voltage until A/D conversion is completed.
- <4> Bit 9 of the successive approximation A/D conversion register (SAR) is set. The series resistor string voltage tap at the tap selector is set to half of  $AV_{DD}$ .
- <5> The series resistor string tap voltage is compared with the analog input voltage using the voltage comparator. If the analog input voltage is higher than half of  $AV_{DD}$ , the MSB of the SAR is left set. If it is lower than half of  $AV_{DD}$ , the MSB is reset.
- <6> Bit 8 of the SAR is set automatically, and comparison shifts to the next stage. The next voltage tap of the series resistor string is selected according to bit 9, which reflects the previous comparison result, as follows:
  - Bit 9 = 1: Three quarters of  $AV_{DD}$
  - Bit 9 = 0: One quarter of  $AV_{DD}$
 The tap voltage is compared with the analog input voltage. Bit 8 is set or reset according to the result of comparison.
  - Analog input voltage  $\geq$  tap voltage: Bit 8 = 1
  - Analog input voltage < tap voltage: Bit 8 = 0
- <7> Comparison is repeated until bit 0 of the SAR is reached.
- <8> When comparison is completed for all of the 10 bits, a significant digital result is left in the SAR. This value is sent to and latched in A/D conversion result register 0 (ADCR0). At the same time, it is possible to generate an A/D conversion end interrupt request (INTAD0).

- Cautions**
1. The A/D conversion value immediately after starting the A/D conversion operation may be undefined.
  2. When in standby mode, the A/D converter stops operation.

Figure 12-4. Basic Operation of 10-Bit A/D Converter



A/D conversion continues until bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) is reset (0) by software.

If an attempt is made to write to ADM0 or Analog input channel specification register 0 (ADS0) during A/D conversion, the ongoing A/D conversion is canceled. In this case, A/D conversion is restarted from the beginning, if ADCS0 is set (1).

RESET input makes A/D conversion result register 0 (ADCR0) undefined.

#### 12.4.2 Input voltage and conversion result

The relationships between the analog input voltage at the analog input pins (ANI0 to ANI3) and the A/D conversion result (A/D conversion result register 0 (ADCR0)) are represented by:

$$\text{ADCR0} = \text{INT} \left( \frac{V_{\text{IN}}}{A V_{\text{DD}}} \times 1,024 + 0.5 \right)$$

or

$$(\text{ADCR0} - 0.5) \times \frac{A V_{\text{DD}}}{1,024} \leq V_{\text{IN}} < (\text{ADCR0} + 0.5) \times \frac{A V_{\text{DD}}}{1,024}$$

INT( ): Function that returns the integer part of a parenthesized value

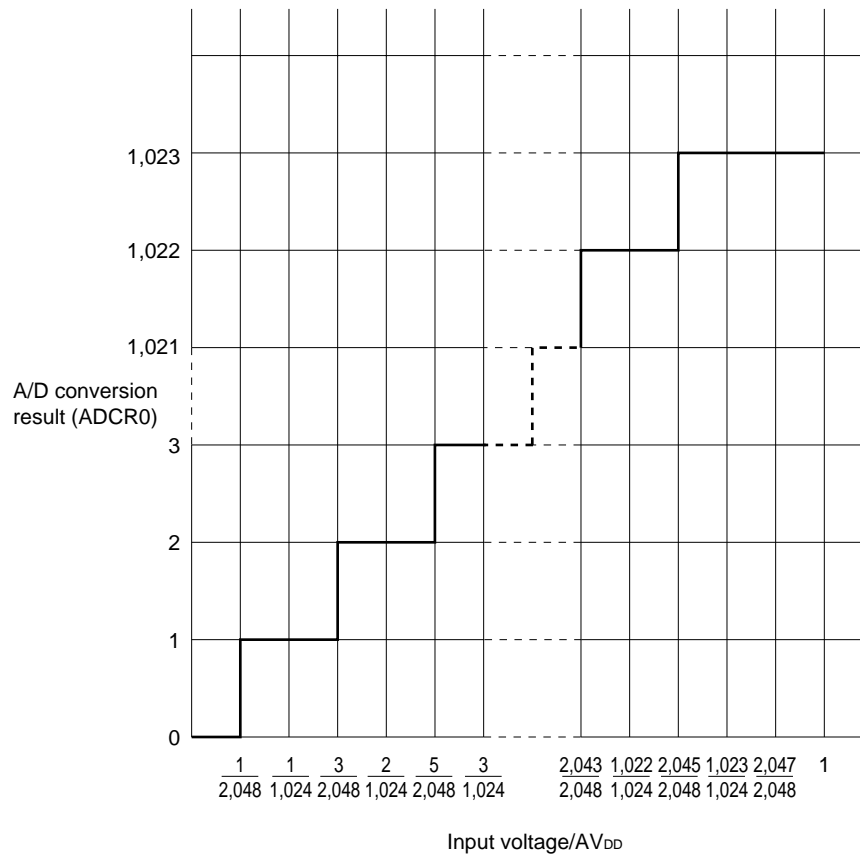
$V_{\text{IN}}$ : Analog input voltage

$A V_{\text{DD}}$ : A/D converter supply voltage

ADCR0: Value in A/D conversion result register 0 (ADCR0)

Figure 12-5 shows the relationships between the analog input voltage and the A/D conversion result.

Figure 12-5. Relationships between Analog Input Voltage and A/D Conversion Result



### 12.4.3 Operation mode of 10-bit A/D converter

The 10-bit A/D converter is initially in the select mode. In this mode, Analog input channel specification register 0 (ADS0) is used to select an analog input channel from ANI0 to ANI3 for A/D conversion.

A/D conversion can be started only by software, that is, by setting A/D converter mode register 0 (ADM0).

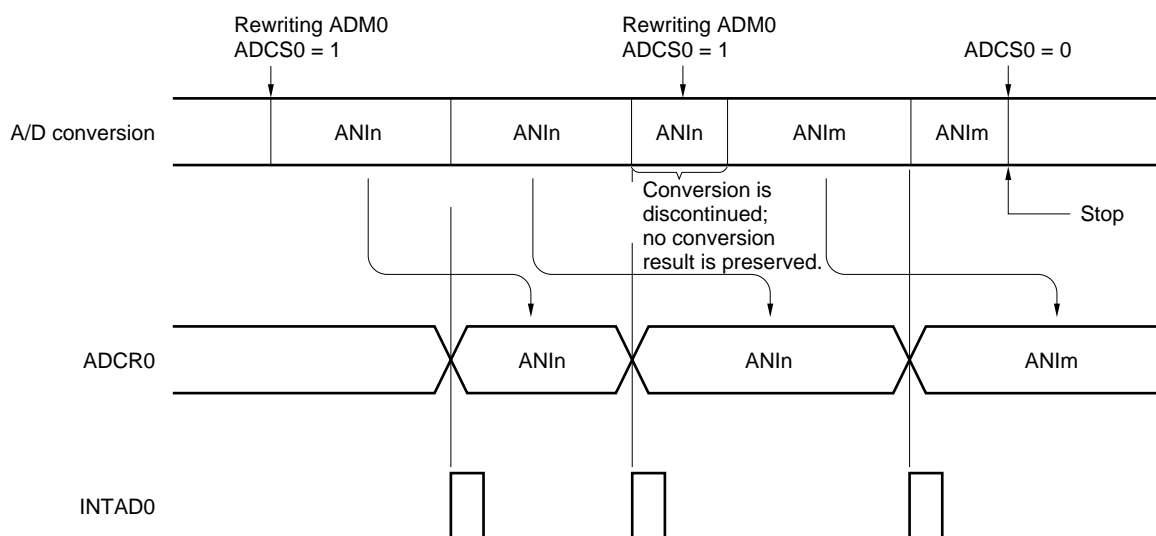
The A/D conversion result is saved to A/D conversion result register 0 (ADCR0). At the same time, an interrupt request signal (INTAD0) is generated.

- **Software-started A/D conversion**

Setting bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) triggers A/D conversion for a voltage applied to the analog input pin specified in Analog input channel specification register 0 (ADS0).

Upon completion of A/D conversion, the conversion result is saved to A/D conversion result register 0 (ADCR0). At the same, an interrupt request signal (INTAD0) is generated. Once A/D conversion is activated, and completed, another session of A/D conversion is started. A/D conversion is repeated until new data is written to the ADM0. If data where ADCS0 is 1 is written to ADM0 again during A/D conversion, the ongoing session of A/D conversion is discontinued, and a new session of A/D conversion begins for the new data. If data where ADCS0 is 0 is written to ADM0 again during A/D conversion, A/D conversion is completely stopped.

**Figure 12-6. Software-Started A/D Conversion**



- Remarks**
1.  $n = 0, 1, 2, 3$
  2.  $m = 0, 1, 2, 3$

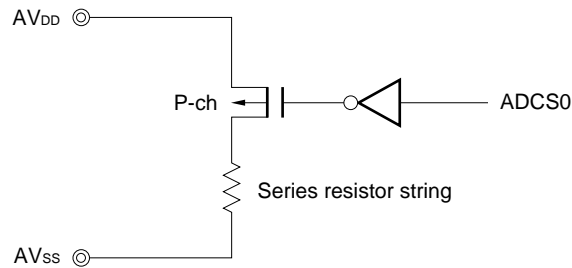
## 12.5 Cautions Related to 10-Bit A/D Converter

### (1) Current consumption in the standby mode

When the A/D converter enters the standby mode, it stops its operation. Stopping conversion (bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) = 0) can reduce the current consumption.

Figure 12-7 shows how to reduce the current consumption in the standby mode.

**Figure 12-7. How to Reduce Current Consumption in Standby Mode**



### (2) Input range for the ANI0 to ANI3 pins

Be sure to keep the input voltage at ANI0 to ANI3 within its rating. If a voltage not lower than AVDD or not higher than AVSS (even within the absolute maximum rating) is input a conversion channel, the conversion output of the channel becomes undefined. It may affect the conversion output of the other channels.

### (3) Conflict

**<1> Conflict between writing to A/D conversion result register 0 (ADCR0) at the end of conversion and reading from ADCR0**

Reading from ADCR0 takes precedence. After reading, the new conversion result is written to ADCR0.

**<2> Conflict between writing to ADCR0 at the end of conversion and writing to A/D converter mode register 0 (ADM0) or Analog input channel specification register 0 (ADS0)**

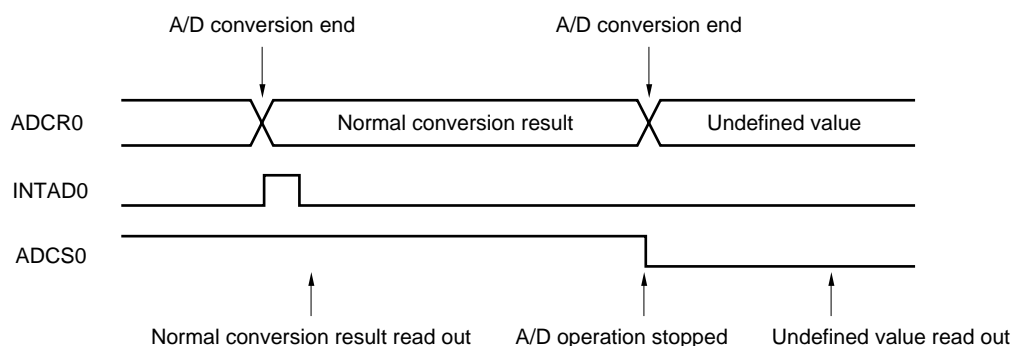
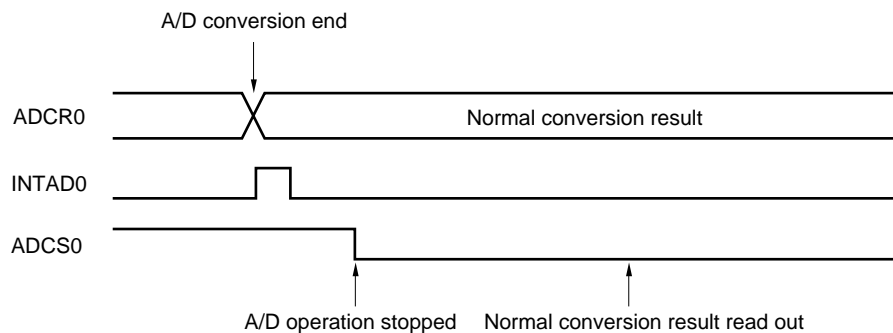
Writing to ADM0 or ADS0 takes precedence. A request to write to ADCR0 is ignored. No conversion end interrupt request signal (INTAD0) is generated.

**(4) Conversion results immediately following start of A/D conversion**

The first A/D conversion value immediately following the start of A/D converter operation may be undefined. Be sure to poll the A/D conversion end interrupt request (INTAD0) and perform processing such as annulling the first conversion result.

**(5) Timing that makes the A/D conversion result undefined**

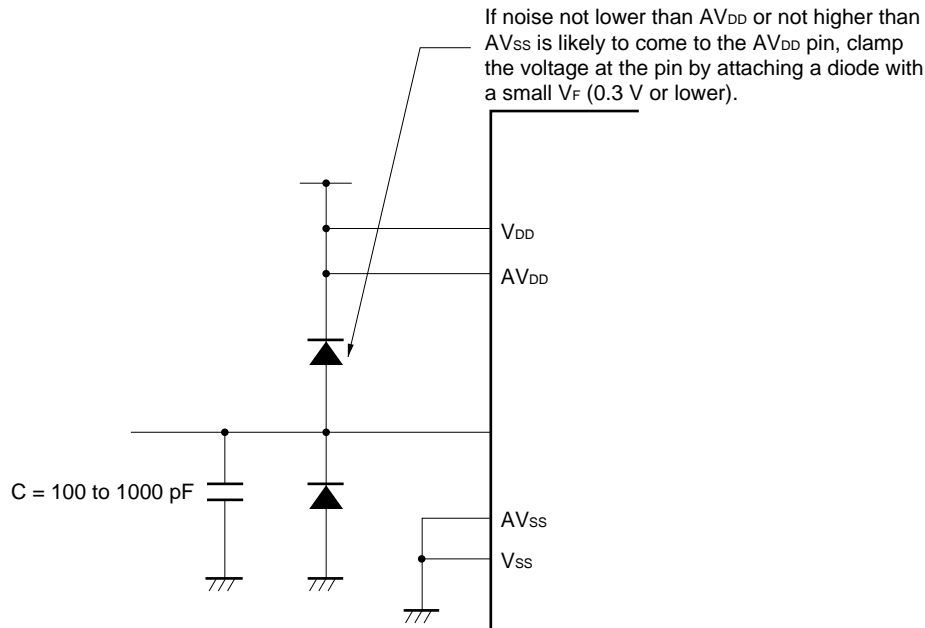
If the timing of the end of A/D conversion and the timing of the stop of operation of the A/C converter conflict, the A/D conversion value may be undefined. Because of this, be sure to read out the A/D conversion result while the A/D converter is in operation. Furthermore, when reading out an A/D conversion result after A/D converter operation has stopped, be sure to have done so by the time the next conversion result is complete. The conversion result readout timing is shown in Figures 12-8 and 12-9.

**Figure 12-8. Conversion Result Readout Timing (When Conversion Result Is Undefined Value)****Figure 12-9. Conversion Result Readout Timing (When Conversion Result Is Normal Value)**

**(6) Noise prevention**

To maintain a resolution of 10 bits, watch for noise to the  $AV_{DD}$  and ANI0 to ANI3 pins. The higher the output impedance of the analog input source is, the larger the effect by noise is. To reduce noise, attach an external capacitor to the relevant pins as shown in Figure 12-10.

**Figure 12-10. Analog Input Pin Treatment**

**(7) ANI0 to ANI3**

The analog input pins (ANI0 to ANI3) are alternate-function pins. They are used also as port pins (P60 to P63).

If any of ANI0 to ANI3 has been selected for A/D conversion, do not execute input instructions for the ports; otherwise, the conversion resolution may become lower.

If a digital pulse is applied to a pin adjacent to the analog input pins during A/D conversion, coupling noise may occur which prevents an A/D conversion result from being attained as expected. Avoid applying a digital pulse to pins adjacent to the analog input pins during A/D conversion.

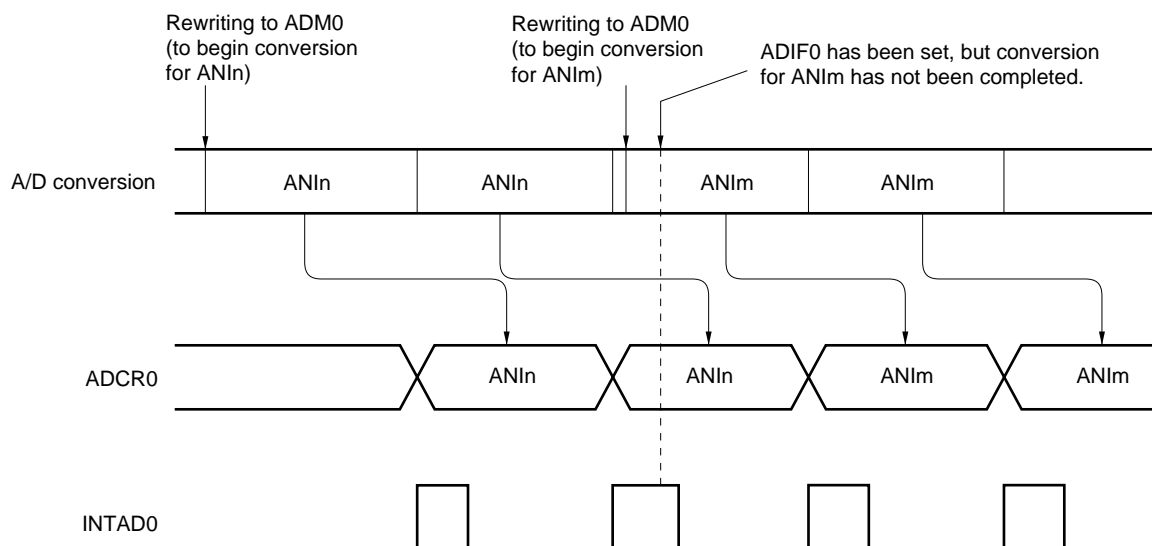
**(8) Interrupt request flag (ADIF0)**

Changing content of the A/D converter mode register 0 (ADM0) does not clear the interrupt request flag (ADIF0).

If the analog input pins are changed during A/D conversion, therefore, the conversion result and the conversion end interrupt request flag may reflect the previous analog input immediately before writing to ADM0 occurs. In this case, ADIF0 may appear to be set if it is read-accessed immediately after ADM0 is write-accessed, even when A/D conversion has not been completed for the new analog input.

In addition, when A/D conversion is restarted, ADIF0 must be cleared beforehand.



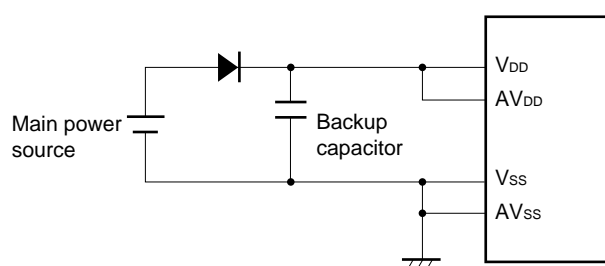
**Figure 12-11. A/D Conversion End Interrupt Request Generation Timing**

- Remarks**
1.  $n = 0, 1, 2, 3$
  2.  $m = 0, 1, 2, 3$

**(9) AV<sub>DD</sub> pin**

The AV<sub>DD</sub> pin is used to supply power to the analog circuit. It is also used to supply power to the ANI0 to ANI3 input circuit.

Therefore, if the application is designed to be changed to backup power, the AV<sub>DD</sub> pin must be supplied with the same voltage level as for the V<sub>DD</sub> pin, as shown in Figure 12-10.

**Figure 12-12. AV<sub>DD</sub> Pin Treatment****(10) Input impedance of the AV<sub>DD</sub> pin**

A series resistor string of several 10 k $\Omega$  is connected across the AV<sub>DD</sub> and AV<sub>SS</sub> pins.

Therefore, if the output impedance of the reference voltage source is high, this high impedance is eventually connected in parallel with the series resistor string across the AV<sub>DD</sub> and AV<sub>SS</sub> pins, leading to a higher reference voltage error.

**[MEMO]**

## CHAPTER 13 SERIAL INTERFACE 20

### 13.1 Functions of Serial Interface 20

Serial interface 20 has 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 performed. Power consumption is minimized in this mode.

#### (2) Asynchronous serial interface (UART) mode

This mode is used to send and receive the one byte of data that follows a start bit. It supports full-duplex communication.

Serial interface channel 0 contains a dedicated UART baud rate generator, enabling communication over a wide range of baud rates. It is also possible to define baud rates by dividing the frequency of the input clock pulse at the ASCK20 pin.

It is recommended that ceramic/crystal oscillation be used for the system clock in the UART mode. Because the frequency deviation is large in RC oscillation, if an internal clock is selected as the source clock for the baud rate generator, there may be problems in send/receive operations.

#### (3) 3-wire serial I/O mode (switchable between MSB-first and LSB-first transmission)

This mode is used to transmit 8-bit data, using three lines: a serial clock ( $\overline{\text{SCK20}}$ ) line and two serial data lines (SI20 and SO20).

As it supports simultaneous transmission and reception, 3-wire serial I/O mode requires less processing time for data transmission than asynchronous serial interface mode.

Because, in 3-wire serial I/O mode, it is possible to select whether 8-bit data transmission begins with the MSB or LSB, channel 0 can be connected to any device regardless of whether that device is designed for MSB-first or LSB-first transmission.

3-wire serial I/O mode is useful for connecting peripheral I/O circuits and display controllers having conventional clock synchronous serial interfaces, such as those of the 75XL, 78K, and 17K Series devices.

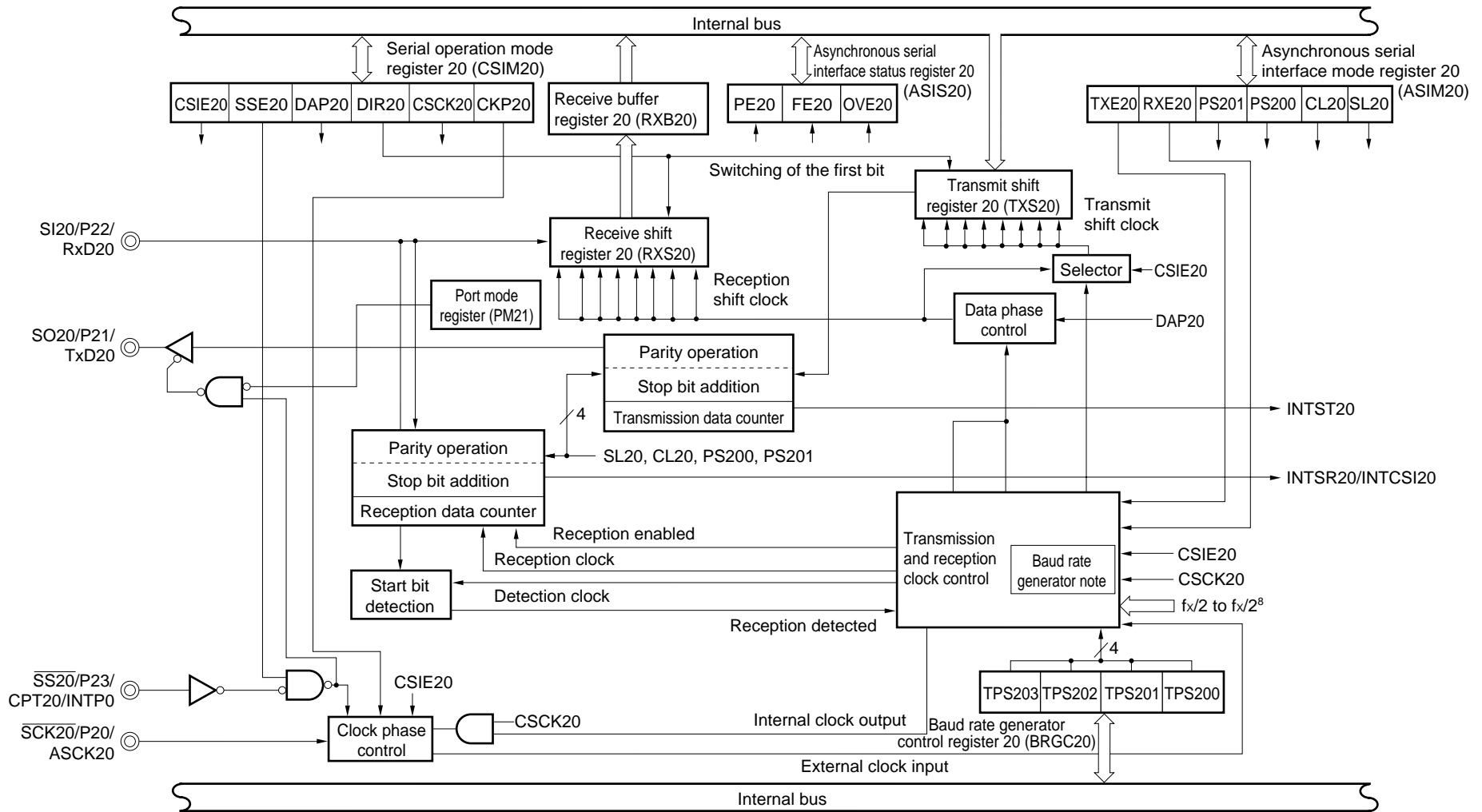
### 13.2 Serial Interface 20 Configuration

Serial interface 20 consists of the following hardware.

**Table 13-1. Serial Interface 20 Configuration**

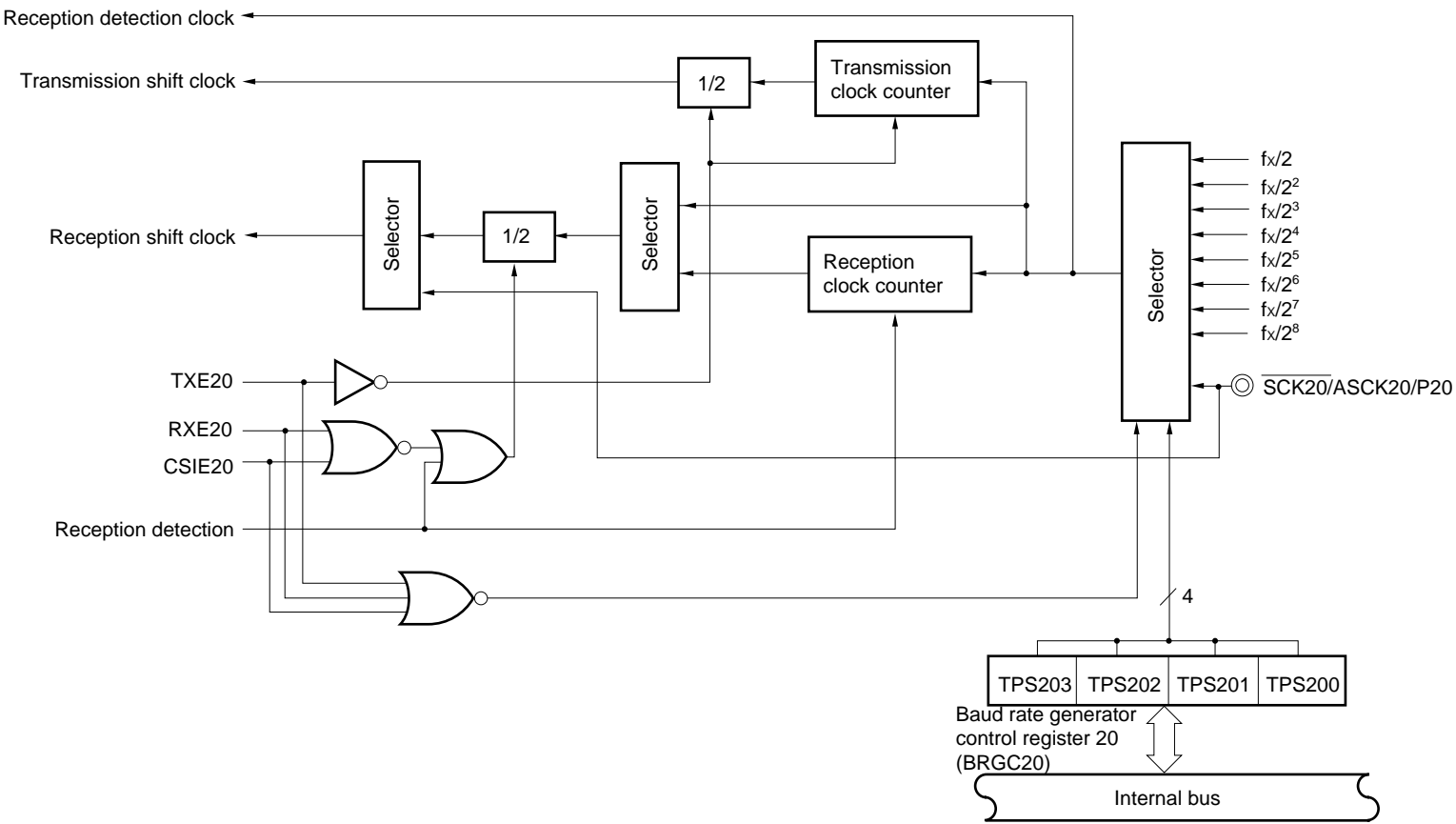
Item	Configuration
Register	Transmit shift register 20 (TXS20) Receive shift register 20 (RXS20) Receive buffer register 20 (RXB20)
Control register	Serial operating mode register 20 (CSIM20) Asynchronous serial interface mode register 20 (ASIM20) Asynchronous serial interface status register 20 (ASIS20) Baud rate generator control register 20 (BRGC20)

Figure 13-1. Serial Interface 20 Block Diagram



**Note** See Figure 13-2 for the configuration of the baud rate generator.

Figure 13-2. Baud Rate Generator Block Diagram



**(1) Transmit shift register 20 (TXS20)**

TXS20 is a register in which transmission data is prepared. The transmission data is output from TXS20 bit-serially.

When the data length is seven bits, bits 0 to 6 of the data in TXS20 will be transmission data. Writing data to TXS20 triggers transmission.

TXS20 can be written with an 8-bit memory manipulation instruction, but cannot be read.

$\overline{\text{RESET}}$  input sets TXS20 to FFH.

**Caution** Do not write to TXS20 during transmission.

**TXS20 and receive buffer register 20 (RXB20) are mapped at the same address, so that any attempt to read from TXS20 results in a value being read from RXB20.**

**(2) Receive shift register 20 (RXS20)**

RXS20 is a register in which serial data, received at the RxD20 pin, is converted to parallel data. Once one entire byte has been received, RXS20 transfers the reception data to receive buffer register 20 (RXB20). RXS20 cannot be manipulated directly by a program.

**(3) Receive buffer register 20 (RXB20)**

RXB20 holds receive data. New receive data is transferred from receive shift register 0 (RXS20) per 1 byte of data received.

When the data length is specified as seven bits, the receive data is sent to bits 0 to 6 of RXB20, in which the MSB is always fixed to 0.

RXB20 can be read with an 8-bit memory manipulation instruction, but cannot be written to.

$\overline{\text{RESET}}$  input makes RXB20 undefined.

**Caution** RXB20 and transmit shift register 20 (TXS20) are mapped at the same address, so that any attempt to write to RXB20 results in a value being written to TXS20.

**(4) Transmission control circuit**

The transmission control circuit controls transmission. For example, it adds start, parity, and stop bits to the data in transmit shift register 20 (TXS20), according to the setting of asynchronous serial interface mode register 20 (ASIM20).

**(5) Reception control circuit**

The reception control circuit controls reception according to the setting of asynchronous serial interface mode register 20 (ASIM20). It also checks for errors, such as parity errors, during reception. If an error is detected, asynchronous serial interface status register 20 (ASIS20) is set according to the status of the error.

### 13.3 Serial Interface 20 Control Registers

Serial interface 20 is controlled by the following four registers.

- Serial operating mode register 20 (CSIM20)
- Asynchronous serial interface mode register 20 (ASIM20)
- Asynchronous serial interface status register 20 (ASIS20)
- Baud rate generator control register 20 (BRGC20)

#### (1) Serial operating mode register 20 (CSIM20)

CSIM20 is used to make the settings related to 3-wire serial I/O mode.

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears CSIM20 to 00H.

**Figure 13-3. Serial Operating Mode Register 20 Format**

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	SSE20	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control										
0	Operation disabled										
1	Operation enabled										

SSE20	$\overline{\text{SS20}}$ -pin selection				Function of the $\overline{\text{SS20}}$ /P23 pin				Communication status		
0	Not used				Port function				Communication enabled		
1	Used				0				Communication enabled		
					1				Communication disabled		

DAP20	3-wire serial I/O mode data phase selection										
0	Outputs at the falling edge of $\overline{\text{SCK20}}$ .										
1	Outputs at the rising edge of $\overline{\text{SCK20}}$ .										

DIR20	First-bit specification										
0	MSB										
1	LSB										

CSCK20	3-wire serial I/O mode clock selection										
0	External clock pulse input to the $\overline{\text{SCK20}}$ pin.										
1	Output of the dedicated baud rate generator										

CKP20	3-wire serial I/O mode clock phase selection										
0	Clock is low active, and $\overline{\text{SCK20}}$ is at high level in the idle state										
1	Clock is high active, and $\overline{\text{SCK20}}$ is at low level in the idle state										

- Cautions**
1. Bits 4 and 5 must be fixed to 0.
  2. CSIM20 must be cleared to 00H, if UART mode is selected.

**(2) Asynchronous serial interface mode register 20 (ASIM20)**

ASIM20 is used to make the settings related to asynchronous serial interface mode.

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears ASIM20 to 00H.

**Figure 13-4. Asynchronous Serial Interface Mode Register 20 Format**

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Send operation stop
1	Send operation enable

RXE20	Receive operation control
0	Receive operation stop
1	Receive operation enable

PS201	PS200	Parity bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception (No parity error is generated).
1	0	Odd parity
1	1	Even parity

CL20	Transmit data character length specification
0	7 bits
1	8 bits

SL20	Transmit data stop bit length
0	1 bit
1	2 bits

- Cautions**
1. Bits 0 and 1 must be fixed to 0.
  2. If 3-wire serial I/O mode is selected, ASIM20 must be cleared to 00H.
  3. Switch operating modes after halting a serial transmit/receive operation.



Table 13-2. Serial Interface 20 Operating Mode Settings

## (1) Operation stopped mode

ASIM20		CSIM20			PM22	P21	PM21	P21	PM20	P20	First Bit	Shift Clock	P22/SI20/RxD20 Pin Function	P21/SO20/TxD20 Pin Function	P20/ $\overline{\text{SCK20}}$ / ASCK20 Pin Function
TXE20	RXE20	CSIE20	DIR20	CSCK20											
0	0	0	×	×	×	×	×	×	×	×	—	—	P22	P21	P20
Other than above											Setting prohibited				

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

ASIM20		CSIM20			PM22	P22	PM21	P21	PM20	P20	First Bit	Shift Clock	P22/SI20/RxD20 Pin Function	P21/SO20/TxD20 Pin Function	P20/ $\overline{\text{SCK20}}$ / ASCK20 Pin Function	
TXE20	RXE20	CSIE20	DIR20	CSCK20												
0	0	1	0	0	$\times$ <sup>Note 1</sup>	$\times$ <sup>Note 2</sup>	0	1	1	$\times$	MSB	External clock	SI20 <sup>Note 2</sup>	SCK20 (CMOS output)	$\overline{\text{SCK20}}$ input	
				1					0	1		Internal clock				$\overline{\text{SCK20}}$ output
		1	1	0					1	$\times$	LSB	External clock				$\overline{\text{SCK20}}$ input
									1	0		1				Internal clock
Other than above											Setting prohibited					

## (3) Asynchronous serial interface mode

ASIM20		CSIM20			PM22	P22	PM21	P21	PM20	P20	First Bit	Shift Clock	P22/SI20/RxD20 Pin Function	P21/SO20/TxD20 Pin Function	P20/SCK20/ ASCK20 Pin Function
TXE20	RXE20	CSIE20	DIR20	CSCCK20											
1	0	0	0	0	×	×	0	1	1	×	LSB	External clock	P22	TxD20 (CMOS output)	ASCK20 input
								×	×	Internal clock		P20			
0	1	0	0	0	1	×	×	×	1	×		External clock	RD20	P21	ASCK20 input
								×	×	Internal clock		P20			
1	1	0	0	0	1	×	0	1	1	×		External clock		TxD20 (CMOS output)	ASCK20 input
								×	×	Internal clock		P20			
Other than above											Setting prohibited				

**Notes** 1. These pins can be used for port functions.

2. When only transmission is used, these pins can be used as P22 (CMOS input/output).

**Remark** ×: don't care.

**(3) Asynchronous serial interface status register 20 (ASIS20)**

ASIS20 is used to display the type of a reception error, if it occurs while asynchronous serial interface mode is set.

ASIS20 is read with a 1-bit or 8-bit memory manipulation instruction.

The contents of ASIS20 are undefined in 3-wire serial I/O mode.

$\overline{\text{RESET}}$  input clears ASIS20 to 00H.

**Figure 13-5. Asynchronous Serial Interface Status Register 20 Format**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ASIS20	0	0	0	0	0	PE20	FE20	OVE20	FF71H	00H	R

PE20	Parity error flag
0	No parity error has occurred.
1	A parity error has occurred (parity mismatch in transmission data).

FE20	Flaming error flag
0	No framing error has occurred.
1	A framing error has occurred (no stop bit detected). <sup>Note 1</sup>

OVE20	Overflow error flag
0	No overflow error has occurred.
1	An overflow error has occurred. <sup>Note 2</sup> (Before data was read from the reception buffer register, the subsequent reception sequence was completed.)

- Notes**
1. Even when the stop bit length is set to 2 bits by setting bit 2 (SL20) of asynchronous serial interface mode register 20 (ASIM20), the stop bit detection in the case of reception is performed with 1 bit.
  2. Be sure to read receive buffer register 20 (RXB20) when an overflow error occurs. If not, every time the data is received an overflow error is generated.

**(4) Baud rate generator control register 20 (BRGC20)**

BRGC20 is used to specify the serial clock for serial interface.

BRGC20 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears BRGC20 to 00H.

**Figure 13-6. Baud Rate Generator Control Register 20 Format**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	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	External clock pulse input at the ASCK20 pin <sup>Note</sup>	—
Other than above				Setting prohibited	

**Note** An external clock can only be used in UART mode.

- Cautions**
1. When writing to BRGC20 is performed during a communication operation, the output of baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC20 during communication operations.
  2. Be sure not to select  $n = 1$  during operation at  $f_x = 5.0$  MHz because  $n = 1$  exceeds the baud rate limit.
  3. When the external input clock is selected, set port mode register 2 (PM2) in input mode.

- Remarks**
1.  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)
  2.  $n$ : Value specified in TPS200 to TPS203 ( $1 \leq n \leq 8$ )
  3. Values in parentheses apply to operation with  $f_x = 5.0$  MHz.

The baud rate transmit/receive clock to be generated is either a signal scaled from the system clock, or a signal scaled from the clock input from the ASCK20 pin.

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

The transmit/receive clock is generated by scaling the system clock. The baud rate generated from the system clock is 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 (ceramic/crystal oscillation)

$n$ : Values in Figure 13-6 specified by the setting in TPS200 to TPS203 ( $2 \leq n \leq 8$ )

**Table 13-3. Example of Relationship between System Clock and Baud Rate**

Baud Rate (bps)	n	BRGC20 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		

**(b) Generation of baud rate transmit/receive clock by means of external clock from ASCK20 pin**

The transmit/receive clock is generated by scaling the clock input from the ASCK20 pin. The baud rate generated from the clock input from the ASCK20 pin is estimated by using the following expression.

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

$f_{\text{ASCK}}$ : Frequency of clock pulse received at the ASCK20 pin

**Table 13-4. Relationship between ASCK20 Pin Input Frequency and Baud Rate (When BRGC20 Is Set to 80H)**

Baud Rate (bps)	ASCK20 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

## 13.4 Serial Interface 20 Operation

Serial interface 20 provides the following three types of modes.

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

### 13.4.1 Operation stop mode

In the operation stop mode, serial transfer is not executed, therefore, the power consumption can be reduced. The P20/SCK20/ASCK20, P21/SO20/TxD20, and P22/SI20/RxD20 pins can be used as normal I/O ports.

#### (1) Register setting

Operation stop mode is set by serial operating mode register 20 (CSIM20) and asynchronous serial interface mode register 20 (ASIM20).

##### (a) Serial operation mode register 20 (CSIM20)

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears CSIM20 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	SSE20	0	0	DAP20	DIR20	CCK20	CKP20	FF72H	00H	R/W

CSIE20	Operation control in 3-wire serial I/O mode
0	Operation disable
1	Operation enable

**Caution** Be sure to set bits 4 and 5 to 0.

**(b) Asynchronous serial interface mode register 20 (ASIM20)**

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears ASIM20 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stop
1	Transmit operation enable

RXE20	Receive operation control
0	Receive operation stop
1	Receive operation enable

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

### 13.4.2 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 communication at the desired transfer rate from many options. In addition, the baud rate can also be defined by dividing the input clock to the ASCK pin.

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

It is recommended that the ceramic/crystal oscillation be used for the system clock in the UART mode. Because the frequency deviation is large in RC oscillation, if an internal clock is selected as the source clock for the baud rate generator, there may be problems in send/receive operations.

#### (1) Register setting

The UART mode is set by serial operating mode register 20 (CSIM20), asynchronous serial interface mode register 20 (ASIM20), asynchronous serial interface status register 20 (ASIS20), and baud rate generator control register 20 (BRGC20).



**(a) Serial operating mode register 20 (CSIM20)**

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears CSIM20 to 00H.

Set 00H to CSIM20 when UART mode is selected.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	SSE20	0	0	DAP20	DIR20	CCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control		
0	Operation disabled		
1	Operation enabled		

SSE20	$\overline{\text{SS20}}$ -pin selection	Function of the $\overline{\text{SS20}}$ /P23 pin	Communication status
0	Not used	Port function	Communication enabled
1	Used	0	Communication enabled
		1	Communication disabled

DAP20	3-wire serial I/O mode data phase selection		
0	Outputs at the falling edge of $\overline{\text{SCK20}}$ .		
1	Outputs at the rising edge of $\overline{\text{SCK20}}$ .		

DIR20	First-bit specification		
0	MSB		
1	LSB		

CCK20	3-wire serial I/O mode clock selection		
0	External clock pulse input to the $\overline{\text{SCK20}}$ pin.		
1	Output of the dedicated baud rate generator		

CKP20	3-wire serial I/O mode clock phase selection		
0	Clock is low active, and $\overline{\text{SCK20}}$ is high level in the idle state		
1	Clock is high active, and $\overline{\text{SCK20}}$ is low level in the idle state		

**Caution** Bits 4 and 5 must be fixed to 0.

**(b) Asynchronous serial interface mode register 20 (ASIM20)**

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears ASIM20 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stop
1	Transmit operation enable

RXE20	Receive operation control
0	Receive operation stop
1	Receive operation enable

PS201	PS200	Parity bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception (No parity error is generated).
1	0	Odd parity
1	1	Even parity

CL20	Character length specification
0	7 bits
1	8 bits

SL20	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. Switch operating modes after halting the serial transmit/receive operation.

**(c) Asynchronous serial interface status register 20 (ASIS20)**

ASIS20 is read with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears ASIS20 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ASIS20	0	0	0	0	0	PE20	FE20	OVE20	FF71H	00H	R

PE20	Parity error flag
0	Parity error not generated
1	Parity error generated (when the parity of transmit data does not match.)

FE20	Framing error flag
0	Framing error not generated
1	Framing error generated (when stop bit is not detected.) <sup>Note 1</sup>

OVE20	Overrun error flag
0	Overrun error not generated
1	Overrun error generated <sup>Note 2</sup> (when the next receive operation is completed before the data is read from the receive buffer register.)

- Notes**
1. Even when the stop bit length is set to 2 bits by setting bit 2 (SL20) of asynchronous serial interface mode register 20 (ASIM20), the stop bit detection in the case of reception is performed with 1 bit.
  2. Be sure to read receive buffer register 20 (RXB20) when an overrun error occurs. If not, every time the data is received an overrun error is generated.

**(d) Baud rate generator control register 20 (BRGC20)**

BRGC20 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears BRGC20 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	3-bit counter source clock selection	n
0	0	0	0	$f_x/2$ (2.5 MHz)	1
0	0	0	1	$f_x/2^2$ (1.25 MHz)	2
0	0	1	0	$f_x/2^3$ (625 kHz)	3
0	0	1	1	$f_x/2^4$ (313 kHz)	4
0	1	0	0	$f_x/2^5$ (156 kHz)	5
0	1	0	1	$f_x/2^6$ (78.1 kHz)	6
0	1	1	0	$f_x/2^7$ (39.1 kHz)	7
0	1	1	1	$f_x/2^8$ (19.5 kHz)	8
1	0	0	0	Input clock from external to ASCK20 pin.	—
Other than above				Setting prohibited	

- Cautions**
1. When writing to BRGC20 is performed during a communication operation, the output of baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC20 during communication operation.
  2. Be sure not to select  $n = 1$  during an operation at  $f_x = 5.0$  MHz because  $n = 1$  exceeds the baud rate limit.
  3. When the external input clock is selected, set port mode register 2 (PM2) to input mode.

- Remarks**
1.  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)
  2.  $n$ : Values specified by the setting in TPS200 to TPS203 ( $1 \leq n \leq 8$ )
  3. Values in parentheses apply to operation with  $f_x = 5.0$  MHz.

The baud rate transmit/receive clock to be generated is either a signal scaled from the system clock, or a signal scaled from the clock input from the ASCK20 pin.

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

The transmit/receive clock is generated by scaling the system clock. The baud rate generated from the system clock is 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 (ceramic/crystal oscillation)

$n$ : Values in the above table specified by the setting in TPS200 to TPS203 ( $2 \leq n \leq 8$ )

**Table 13-5. Example of Relationship between System Clock and Baud Rate**

Baud Rate (bps)	n	BRGC20 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		

**(ii) Generation of baud rate transmit/receive clock by means of external clock from ASCK20 pin**

The transmit/receive clock is generated by scaling the clock input from the ASCK20 pin. The baud rate generated from the clock input from the ASCK20 pin is estimated by using the following expression.

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

$f_{\text{ASCK}}$ : Frequency of clock input to ASCK20 pin.

**Table 13-6. Relationship between ASCK20 Pin Input Frequency and Baud Rate (When BRGC20 Is Set to 80H)**

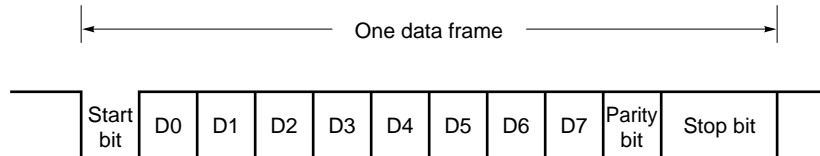
Baud Rate (bps)	ASCK20 Pin input Frequency (kHz)
75	1.2
150	2.4
300	4.8
600	9.6
1,200	19.2
2,400	38.4
4,800	76.8
9,600	153.6
19,200	307.2
31,250	500.0
38,400	614.4

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

The transmit/receive data format is as shown in Figure 13-7. One data frame consists of a start bit, character bits, parity bit and stop bit(s).

The specification of character bit length, parity selection, and specification of stop bit length for each data frame is carried out using asynchronous serial interface mode register 20 (ASIM20).

**Figure 13-7. Asynchronous Serial Interface Transmit/Receive Data Format**



- Start bits ..... 1 bit
- Character bits ..... 7 bits/8 bits
- Parity bits ..... Even parity/odd parity/0 parity/no parity
- Stop bits ..... 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 ASIM20 and baud rate generator control register 20 (BRGC20).

If a serial data receive error is generated, the receive error contents can be determined by reading the status of asynchronous serial interface status register 20 (ASIS20).

**(b) Parity types and operation**

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

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

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

The number of bits with a value of "1" is an odd number in transmit data: 1

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

**• At reception**

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

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

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

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

The number of bits with a value of "1" is an even number in transmit data: 1

**• At reception**

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

**(iii) 0 Parity**

When transmitting, the parity bit is set to "0" irrespective of the transmit data.

At reception, a parity bit check is not performed. Therefore, a parity error is not generated, irrespective of whether the parity bit is set to "0" or "1".

**(iv) No parity**

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

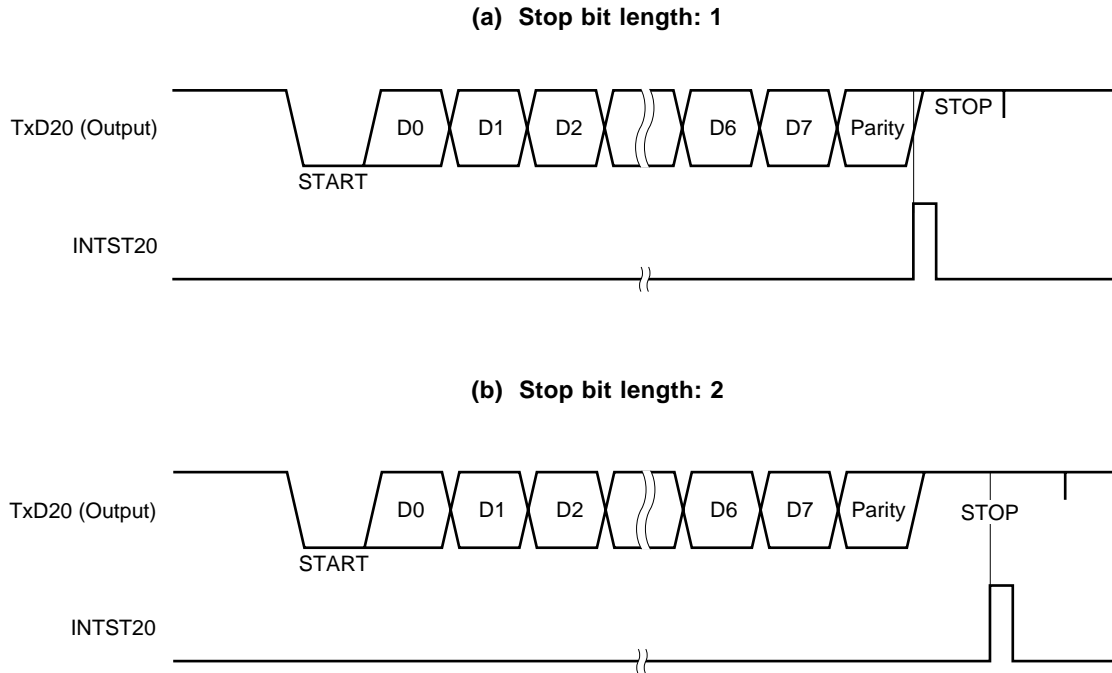


**(c) Transmission**

A transmit operation is started by writing transmit data to transmit shift register 20 (TXS20). The start bit, parity bit and stop bit are added automatically.

When the transmit operation starts, the data in TXS20 is shifted out, and when TXS20 is empty, a transmission completion interrupt (INTST20) is generated.

**Figure 13-8. Asynchronous Serial Interface Transmission Completion Interrupt Timing**



**Caution** Do not rewrite asynchronous serial interface mode register 20 (ASIM20) during a transmit operation. If ASIM20 register is rewritten during transmission, subsequent transmission may not be performed (the normal state is restored by  $\overline{\text{RESET}}$  input).

It is possible to determine whether transmission is in progress by software by using a transmission completion interrupt (INTST20) or the interrupt request flag (STIF20) set by INTST20.

**(d) Reception**

When bit 6 (RXE20) of asynchronous serial interface mode register 20 (ASIM20) is set (1), a receive operation is enabled and sampling of the RxD20 pin input is performed.

RxD20 pin input sampling is performed using the serial clock specified by ASIM20.

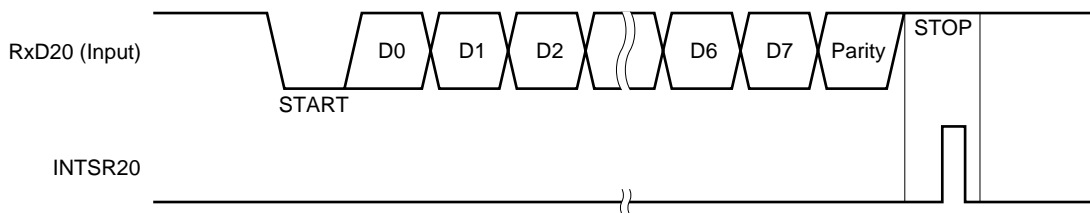
When the RxD20 pin input becomes low, the 3-bit counter starts counting, and when half the time determined by the specified baud rate has passed, the data sampling start timing signal is output. If the RxD20 pin input sampled again as a result of this start timing signal is low, it is identified as a 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 20 (RXB20), and a reception completion interrupt (INTSR20) is generated.

If an error is generated, the receive data in which the error was generated is still transferred to RXB20, and INTSR20 is generated.

If the RXE20 bit is reset (0) during the receive operation, the receive operation is stopped immediately. In this case, the contents of RXB20 and asynchronous serial interface status register 20 (ASIS20) are not changed, and INTSR20 is not generated.

**Figure 13-9. Asynchronous Serial Interface Reception Completion Interrupt Timing**



**Caution** Be sure to read receive buffer register 20 (RXB20) even if a receive error occurs. If RXB20 is not read, an overrun error will be generated when the next data is received, and the receive error state will continue indefinitely.

**(e) Receive errors**

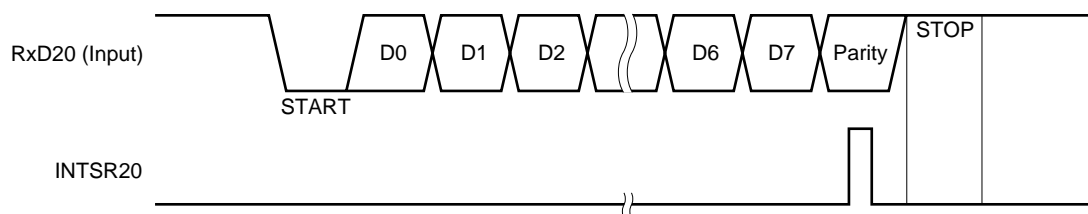
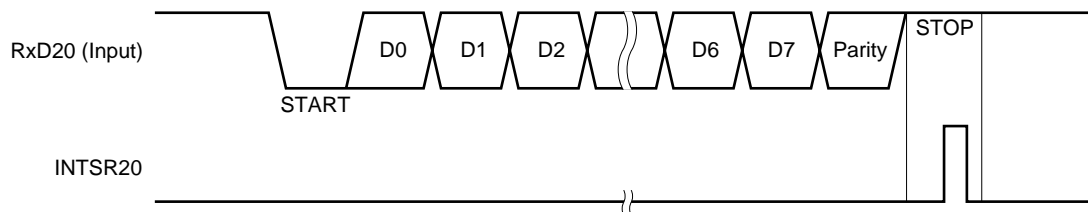
The following three errors may occur during a receive operation: a parity error, framing error, or overrun error. The data reception result error flag is set in asynchronous serial interface status register 20 (ASIS20). Receive error causes are shown in Table 13-7.

It is possible to determine what kind of error was generated during reception by reading the contents of ASIS20 in the reception error interrupt servicing (see Figures 13-9 and 13-10).

The contents of ASIS20 are reset (0) by reading receive buffer register 20 (RXB20) or receiving the next data (if there is an error in the next data, the corresponding error flag is set).

**Table 13-7. Receive Error Causes**

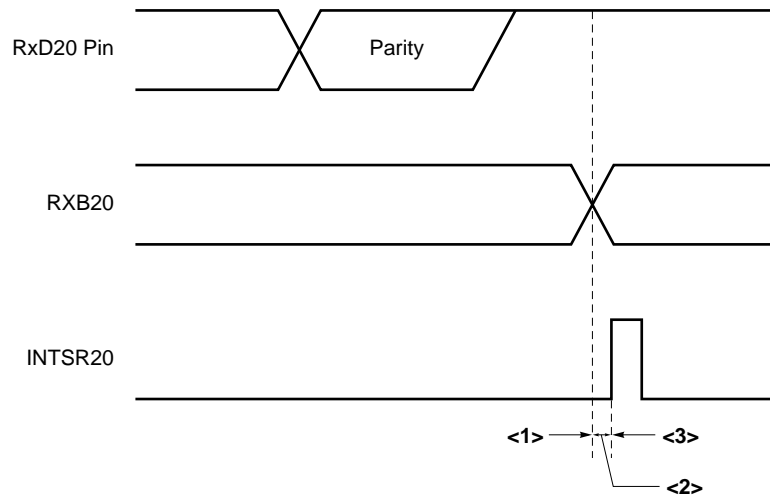
Receive Errors	Cause
Parity error	Transmission-time parity specification and reception data parity do not match
Framing error	Stop bit not detected
Overrun error	Reception of next data is completed before data is read from receive register buffer

**Figure 13-10. Receive Error Timing****(a) Parity error generated****(b) Framing error or overrun error generated**

- Cautions**
1. The contents of the ASIS20 register are reset (0) by reading receive buffer register 20 (RXB20) or receiving the next data. To ascertain the error contents, read ASIS20 before reading RXB20.
  2. Be sure to read receive buffer register 20 (RXB20) even if a receive error is generated. If RXB20 is not read, an overrun error will be generated when the next data is received, and the receive error state will continue indefinitely.

**(3) UART mode cautions**

- (a) When bit 7 (TXE20) of asynchronous serial interface mode register 20 (ASIM20) is cleared during transmission, be sure to set transmit shift register 20 (TXS20) to FFH, then set TXE20 to 1 before executing the next transmission.
- (b) When bit 6 (RXE20) of asynchronous serial interface mode register 20 (ASIM20) is cleared during reception, receive buffer register 20 (RXB20) and receive completion interrupt 20 (INTSR20) are as follows.



When RXE20 is set to 0 at a time indicated by **<1>**, RXB20 holds the previous data and does not generate INTSR20.

When RXE20 is set to 0 at a time indicated by **<2>**, RXB20 renews the data and does not generate INTSR20.

When RXE20 is set to 0 at a time indicated by **<3>**, RXB20 renews the data and generates INTSR20.

### 13.4.3 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. that incorporate a conventional synchronous clocked serial interface, such as the 75XL Series, 78K Series, 17K Series, etc.

Communication is performed using three lines: the serial clock ( $\overline{\text{SCK20}}$ ), serial output ( $\text{SO20}$ ), and serial input ( $\text{SI20}$ ).

#### (1) Register setting

3-wire serial I/O mode settings are performed using serial operating mode register 20 (CSIM20), asynchronous serial interface mode register 20 (ASIM20), and baud rate generator control register 20 (BRGC20).

##### (a) Serial operating mode register 20 (CSIM20)

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears CSIM20 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	SSE20	0	0	DAP20	DIR20	CSCCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control										
0	Operation disabled										
1	Operation enabled										

SSE20	$\overline{\text{SS20}}$ -pin selection			Function of the $\overline{\text{SS20}}$ /P23 pin		Communication status
0	Not used			Port function		Communication enabled
1	Used			0		Communication enabled
				1		Communication disabled

DAP20	3-wire serial I/O mode data phase selection										
0	Outputs at the falling edge of $\overline{\text{SCK20}}$ .										
1	Outputs at the rising edge of $\overline{\text{SCK20}}$ .										

DIR20	First-bit specification										
0	MSB										
1	LSB										

CSCCK20	3-wire serial I/O mode clock selection										
0	External clock pulse input to the $\overline{\text{SCK20}}$ pin.										
1	Output of the dedicated baud rate generator										

CKP20	3-wire serial I/O mode clock phase selection										
0	Clock is low active, and $\overline{\text{SCK20}}$ is at high level in the idle state										
1	Clock is high active, and $\overline{\text{SCK20}}$ is at low level in the idle state										

**Caution** Bits 4 and 5 must be fixed to 0.

**(b) Asynchronous serial interface mode register 20 (ASIM20)**

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears ASIM20 to 00H.

When the 3-wire serial I/O mode is selected, 00H must be set to ASIM20.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stop
1	Transmit operation enable

RXE20	Receive operation control
0	Receive operation stop
1	Receive operation enable

PS201	PS200	Parity bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception (No parity error is generated).
1	0	Odd parity
1	1	Even parity

CL20	Character length specification
0	7 bits
1	8 bits

SL20	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 operation modes must be performed after the serial transmit/receive operation is halted.

**(c) Baud rate generator control register 20 (BRGC20)**

BRGC20 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input clears BRGC20 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	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 BRGC20 is performed during a communication operation, the baud rate generator output is disrupted and communication cannot be performed normally. Be sure not to write to BRGC20 during communication operation.
  2. Be sure not to select  $n = 1$  during an operation at  $f_x = 5.0$  MHz because  $n = 1$  exceeds the baud rate limit.
  3. When the external input clock is selected, set port mode register 2 (PM2) in input mode.

- Remarks**
1.  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)
  2.  $n$ : Values specified by TPS200 to TPS203 ( $1 \leq n \leq 8$ )
  3. Values in parentheses apply to operation with  $f_x = 5.0$  MHz.

If the internal clock is used as the serial clock for the 3-wire serial I/O mode, set the TPS200 to TPS203 bits to set the frequency of the serial clock. To obtain the frequency to be set, use the following formula. When the serial clock is input from off-chip, setting BRGC20 is not necessary.

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

$f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)

$n$ : Values in the above table specified by the setting in TPS200 to TPS203 ( $1 \leq n \leq 8$ )

**(2) Communication operation**

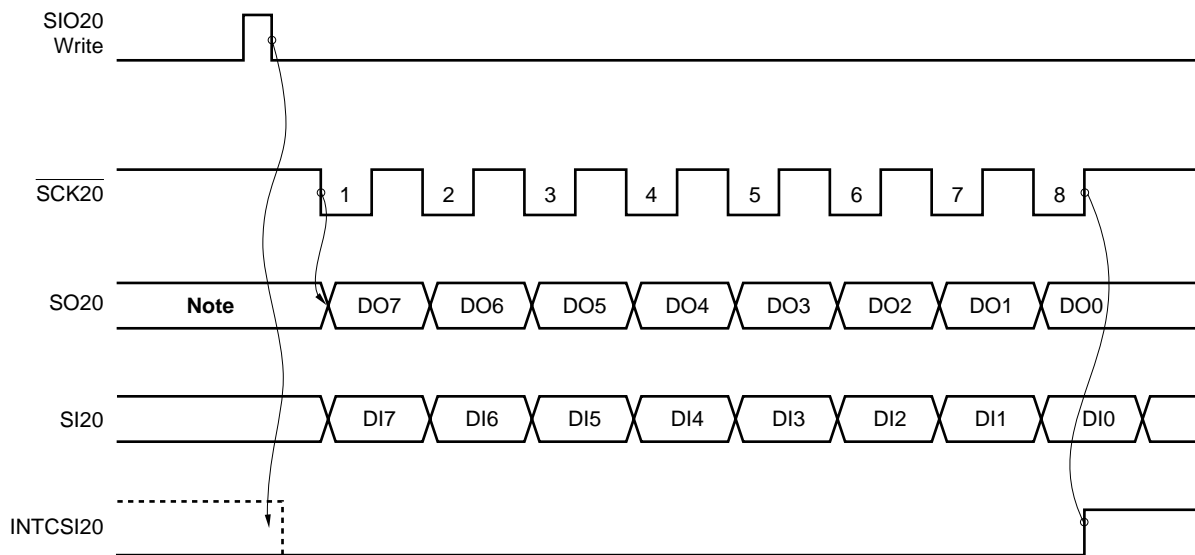
In the 3-wire serial I/O mode, data transmission/reception is performed in 8-bit units. Data is transmitted/received bit by bit in synchronization with the serial clock.

The transmit shift register (TXS20/SIO20) and receive shift register (RXS20) shift operations are performed in synchronization with the fall of the serial clock ( $\overline{\text{SCK20}}$ ). Then transmit data is held in the SO20 latch and output from the SO20 pin. Also, receive data input to the SI0 pin is latched in the receive buffer register (RXB20/SIO20) on the rise of  $\overline{\text{SCK20}}$ .

At the end of an 8-bit transfer, the operation of TXS20/SIO20 or RXS20 stops automatically, and the interrupt request signal (INTCSI20) is generated.

**Figure 13-11. 3-Wire Serial I/O Mode Timing (1/7)**

**(i) Master operation timing (when DAP20 = 0, CKP20 = 0, SSE20 = 0)**

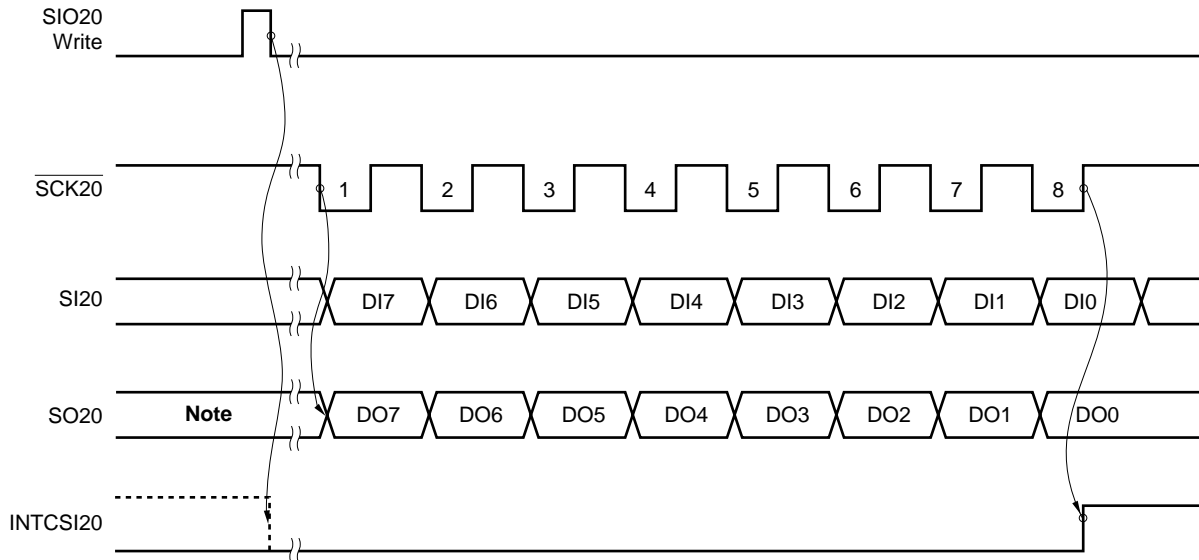


**Note** The value of the last bit previously output is output.



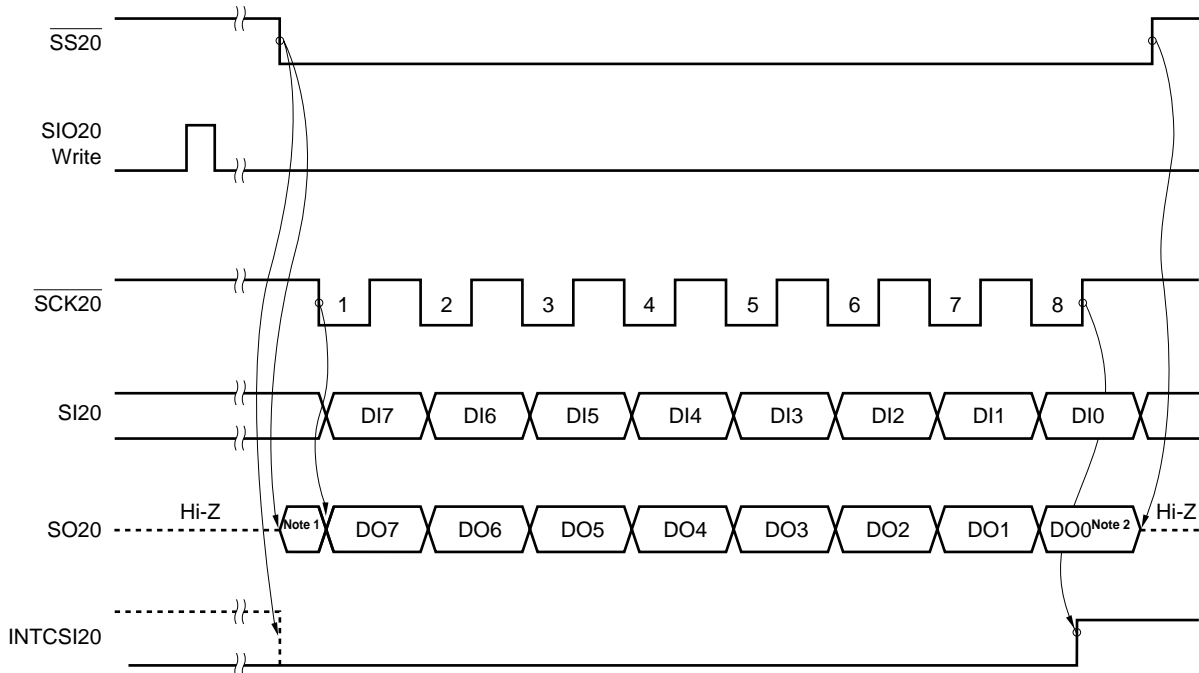
Figure 13-11. 3-Wire Serial I/O Mode Timing (2/7)

## (ii) Slave operation timing (when DAP20 = 0, CKP20 = 0, SSE20 = 0)



**Note** The value of the last bit previously output is output.

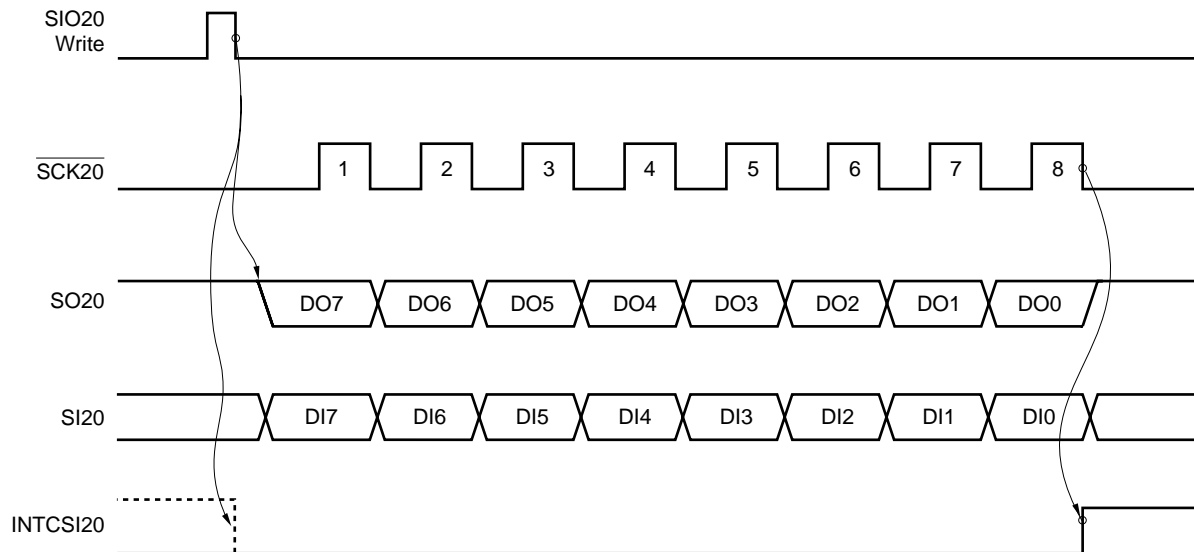
## (iii) Slave operation (when DAP20 = 0, CKP20 = 0, SSE20 = 1)



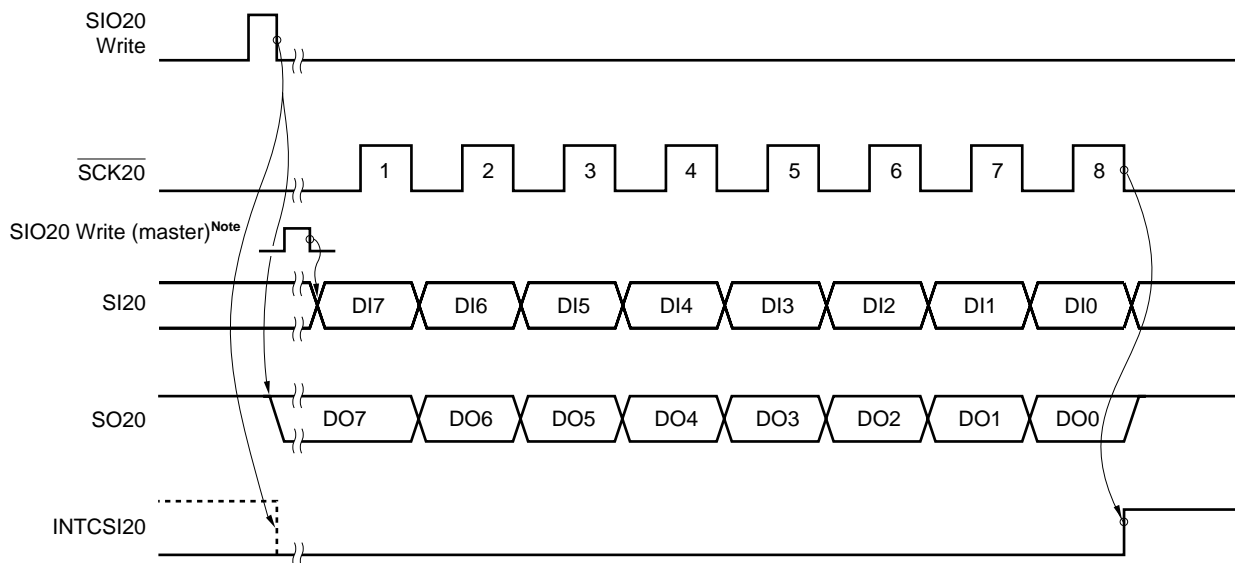
- Notes**
1. The value of the last bit previously output is output.
  2. DO0 is output until  $\overline{SS20}$  rises.  
When  $\overline{SS20}$  is high, SO20 is in a high-impedance state.

Figure 13-11. 3-Wire Serial I/O Mode Timing (3/7)

## (iv) Master operation (when DAP20 = 0, CKP20 = 1, SSE20 = 0)



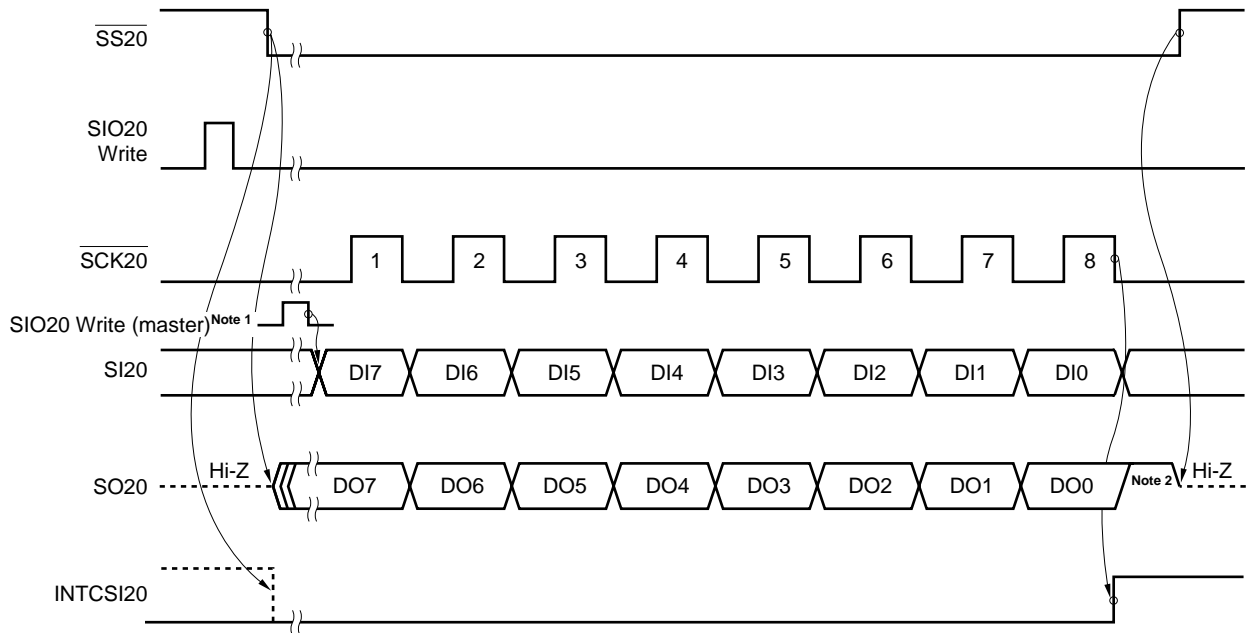
## (v) Slave operation (when DAP20 = 1, CKP20 = 1, SSE20 = 0)



**Note** The data of SI20 is loaded at the first rising edge of SCK20. Make sure that the master outputs the first bit before the first rising of SCK20.

Figure 13-11. 3-Wire Serial I/O Mode Timing (4/7)

## (vi) Slave operation (when DAP20 = 0, CKP20 = 1, SSE20 = 1)



- Notes**
1. The data of SI20 is loaded at the first rising edge of  $\overline{SCK20}$ . Make sure that the master outputs the first bit before the first rising of  $\overline{SCK20}$ .
  2. SO20 is high until  $\overline{SS20}$  rises after completion of DO0 output. When  $\overline{SS20}$  is high, SO20 is in a high-impedance state.

## (vii) Master operation (when DAP20 = 1, CKP20 = 0, SSE20 = 0)

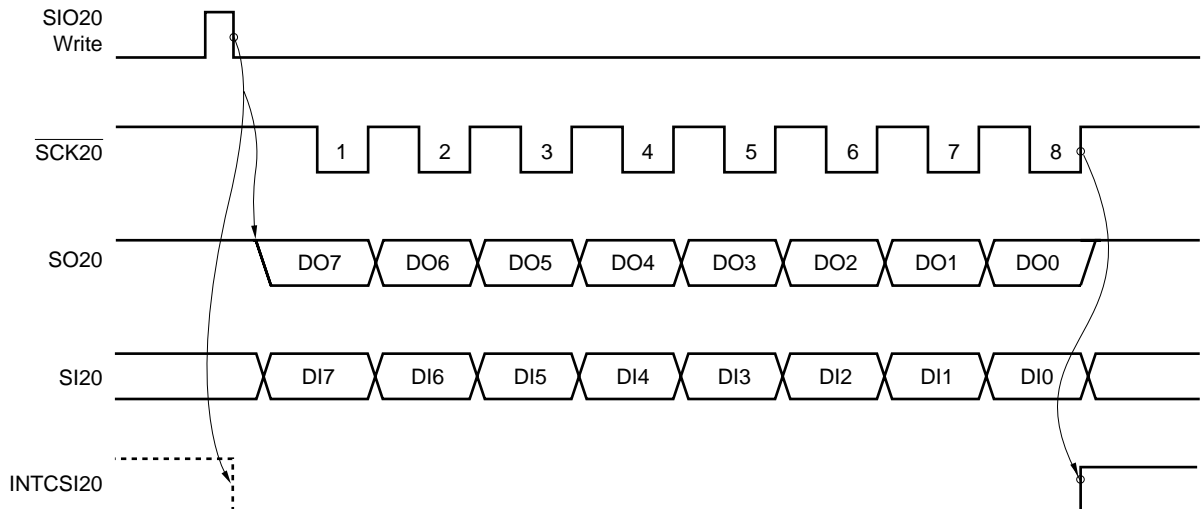
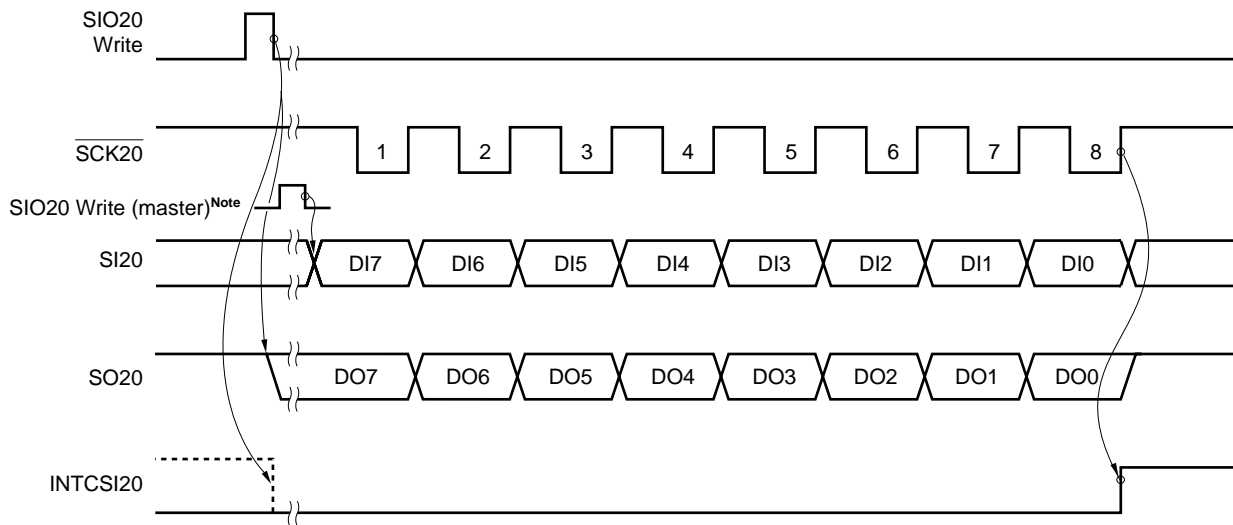


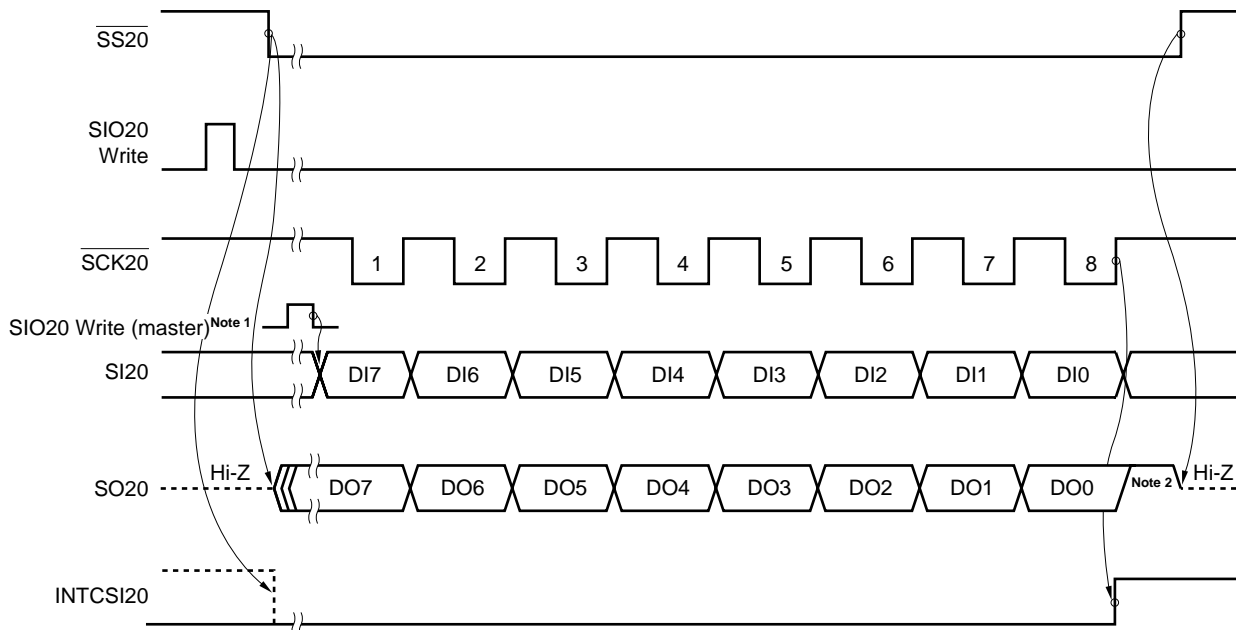
Figure 13-11. 3-Wire Serial I/O Mode Timing (5/7)

## (viii) Slave operation (when DAP20 = 1, CKP20 = 0, SSE20 = 0)



**Note** The data of SI20 is loaded at the first falling edge of  $\overline{\text{SCK20}}$ . Make sure that the master outputs the first bit before the first falling of  $\overline{\text{SCK20}}$ .

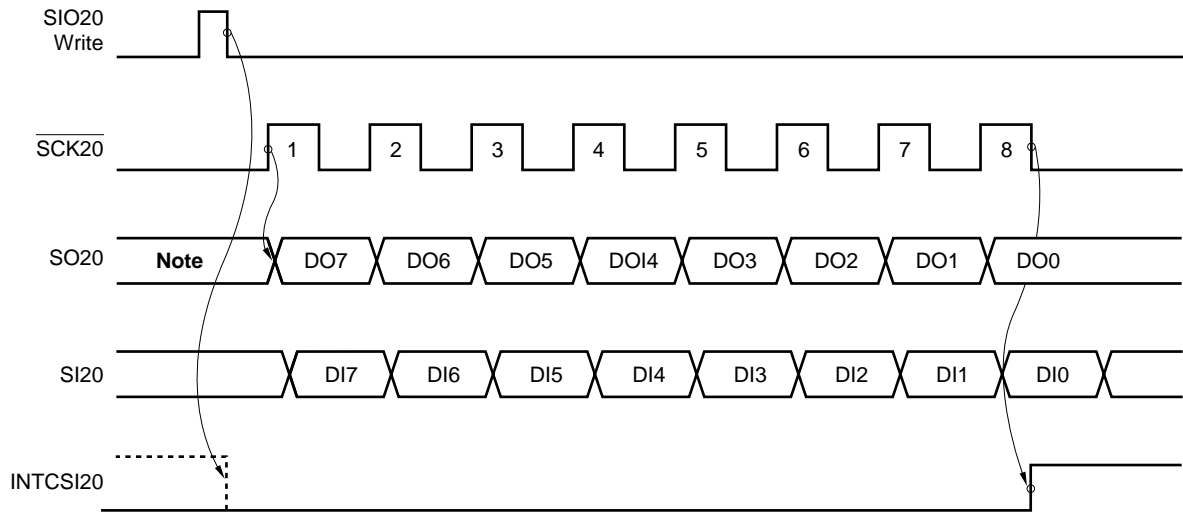
## (ix) Slave operation (when DAP20 = 1, CKP20 = 0, SSE20 = 1)



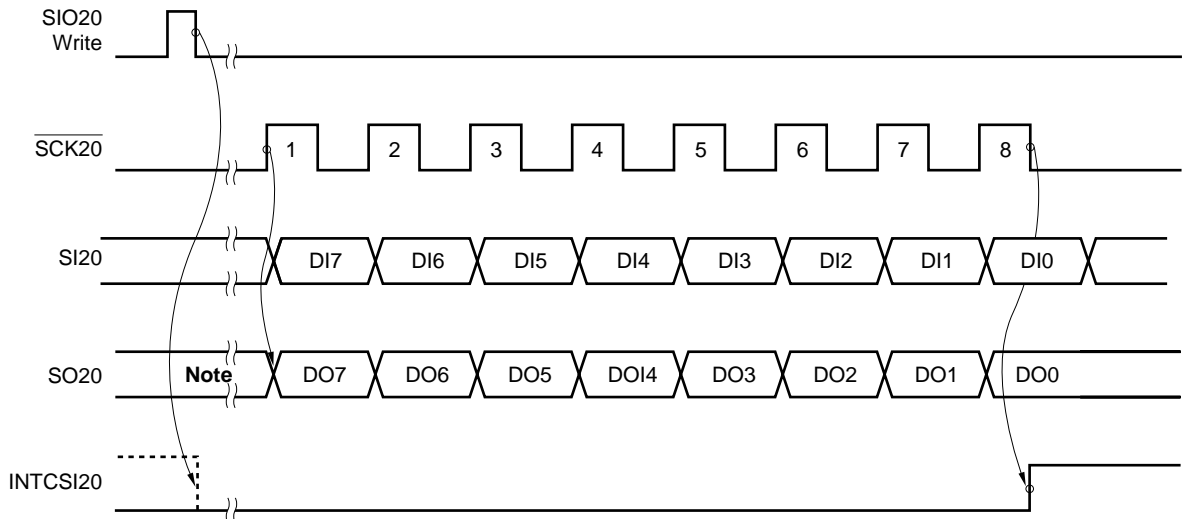
**Notes**

1. The data of SI20 is loaded at the first falling edge of  $\overline{\text{SCK20}}$ . Make sure that the master outputs the first bit before the first falling of  $\overline{\text{SCK20}}$ .
2. SO20 is high until  $\overline{\text{SS20}}$  rises after completion of DO0 output. When  $\overline{\text{SS20}}$  is high, SO20 is in a high-impedance state.

Figure 13-11. 3-Wire Serial I/O Mode Timing (6/7)

**(x) Master operation (when DAP20 = 1, CKP20 = 1, SSE20 = 0)**

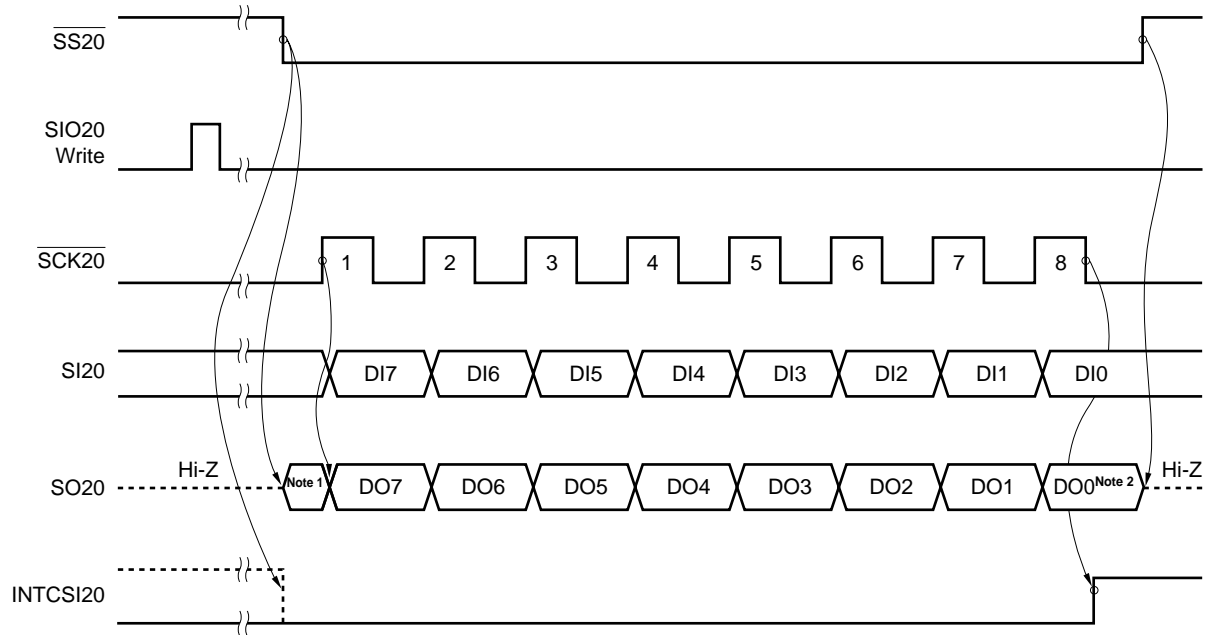
**Note** The value of the last bit previously output is output.

**(xi) Slave operation (when DAP20 = 1, CKP20 = 1, SSE20 = 0)**

**Note** The value of the last bit previously output is output.

Figure 13-11. 3-Wire Serial I/O Mode Timing (7/7)

## (xii) Slave operation (when DAP20 = 1, CKP20 = 1, SSE20 = 1)



- Notes**
1. The value of the last bit previously output is output.
  2. DO0 is output until  $\overline{\text{SS20}}$  rises.  
When  $\overline{\text{SS20}}$  is high, SO20 is in a high-impedance state.

**(3) Transfer start**

Serial transfer is started by setting transfer data to the transmission shift register (TXS20/SIO20) when the following two conditions are satisfied.

- Serial operation mode register 20 (CSIM20) bit 7 (CSIE20) = 1
- Internal serial clock is stopped or  $\overline{\text{SCK20}}$  is a high level after 8-bit serial transfer.

**Caution** If CSIE20 is set to “1” after data write to TXS20/SIO20, transfer does not start.

A termination of 8-bit transfer stops the serial transfer automatically and generates the interrupt request signal (INTCSI20).

### 14.1 Multiplier Function

The multiplier has the following function:

- Calculation of 8 bits  $\times$  8 bits = 16 bits

### 14.2 Multiplier Configuration

#### (1) 16-bit multiplication result storage register 0 (MUL0)

This register stores the 16-bit result of multiplication.

This register holds the result of multiplication after 16 CPU clocks have elapsed.

MUL0 is set with a 16-bit memory manipulation instruction.

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

**Caution** Although this register is manipulated with a 16-bit memory manipulation instruction, it can be also manipulated with an 8-bit memory manipulation instruction. When using an 8-bit memory manipulation instruction, however, access the register by means of direct addressing.

#### (2) Multiplication data registers A and B (MRA0 and MRB0)

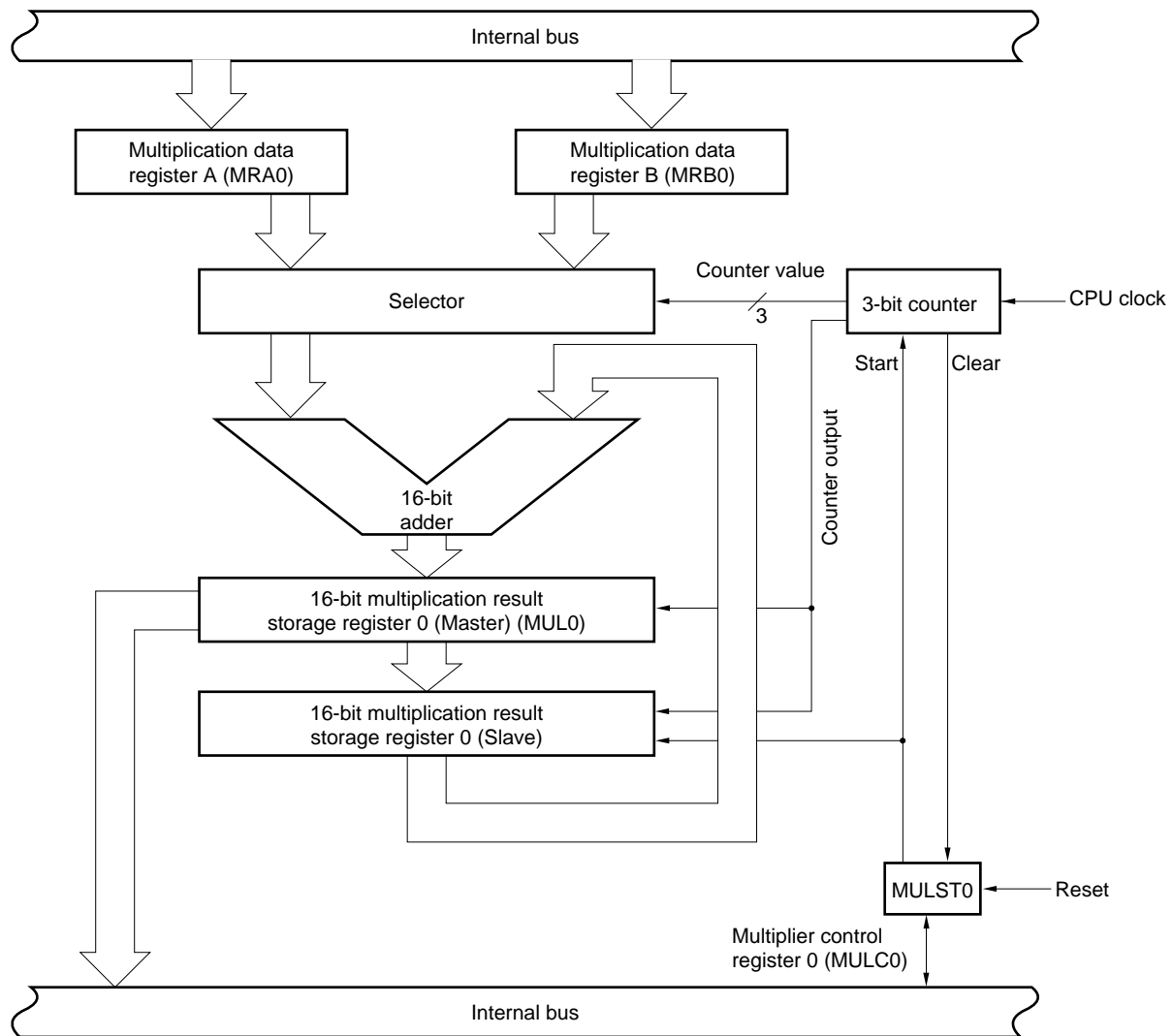
These are 8-bit multiplication data storage registers. The multiplier multiplies the values of MRA0 and MRB0.

MRA0 and MRB0 are set with a 1-bit or 8-bit memory manipulation instructions.

$\overline{\text{RESET}}$  input makes these registers undefined.

Figure 14-1 shows the block diagram of the multiplier.

Figure 14-1. Block Diagram of Multiplier





### 14.3 Multiplier Control Register

The multiplier is controlled by the following register:

- Multiplier control register (MULC0)

MULC0 indicates the operating status of the multiplier after operation, as well as controls the multiplier.

MULC0 is set with a 1-bit or 8-bit memory manipulation instruction.

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

**Figure 14-2. Multiplier Control Register 0 Format**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MULC0	0	0	0	0	0	0	0	MULST0	FFD2H	00H	R/W

MULST0	Multiplier operation start control bit	Operating status of multiplier
0	Stops operation after resetting counter to 0.	Operation stops
1	Enables operation	Operation in progress

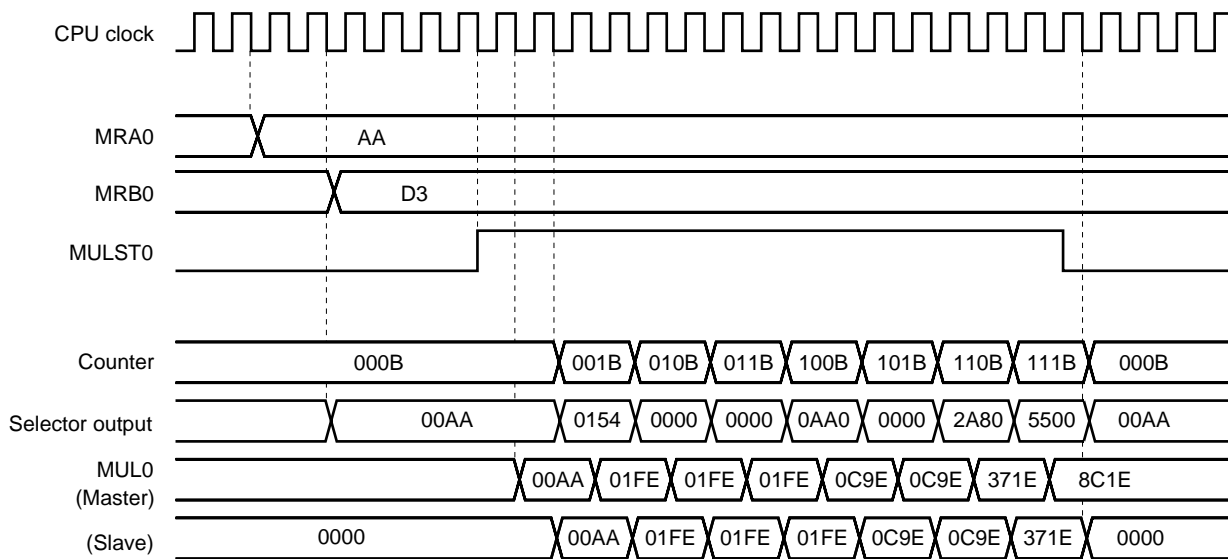
**Caution** Be sure to set bits 1 to 7 to 0.

## 14.4 Multiplier Operation

The multiplier of the  $\mu$ PD789104A/114A/124A/134A Subseries can execute the calculation of 8 bits  $\times$  8 bits = 16 bits. Figure 14-3 shows the operation timing of the multiplier where MRA0 is set to AAH and MRB0 is set to D3H.

- <1> Counting is started by setting MULST0.
- <2> The data generated by the selector is added to the data of MUL0 at each CPU clock, and the counter value is incremented by one.
- <3> If MULST0 is cleared when the counter value is 111B, the operation is stopped. At this time, MUL0 holds the data.
- <4> While MULST0 is low, the counter and slave are cleared.

**Figure 14-3. Multiplier Operation Timing**



## CHAPTER 15 INTERRUPT FUNCTIONS

### 15.1 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 non-maskable interrupt has one source of interrupt from the watchdog timer.

#### (2) Maskable interrupt

These interrupts undergo mask control. If two or more interrupts with the same priority are simultaneously generated, each interrupt has a predetermined priority as shown in Table 15-1.

A standby release signal is generated.

The maskable interrupt has three sources of external interrupts and six sources of internal interrupts.

### 15.2 Interrupt Sources and Configuration

There are total of 10 non-maskable and maskable interrupts in the interrupt sources (see **Table 15-1**).

Table 15-1. Interrupt Source List

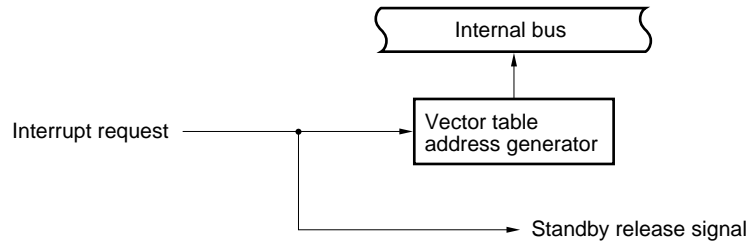
Interrupt Type	Priority <sup>Note 1</sup>	Interrupt Source		Internal /External	Vector Table Address	Basic Configuration Type <sup>Note 2</sup>
		Name	Trigger			
Non-maskable	—	INTWDT	Watchdog timer overflow (watchdog timer mode 1 selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (interval timer mode selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTSR20	End of serial interface 20 UART reception	Internal	000CH	(B)
		INTCSI20	End of serial interface 20 3-wire transfer			
	5	INTST20	End of serial interface 20 UART transmission		000EH	
	6	INTTM80	Generation of 8-bit timer/event counter 80 match signal		0010H	
	7	INTTM20	Generation of 16-bit timer 20 match signal		0012H	
	8	INTAD0	A/D conversion completion signal		0014H	

- Notes**
1. Priority is the priority applicable when two or more maskable interrupts are simultaneously generated. 0 is the highest priority and 8 is the lowest priority.
  2. Basic configuration types A to C correspond to A to C in Figure 15-1.

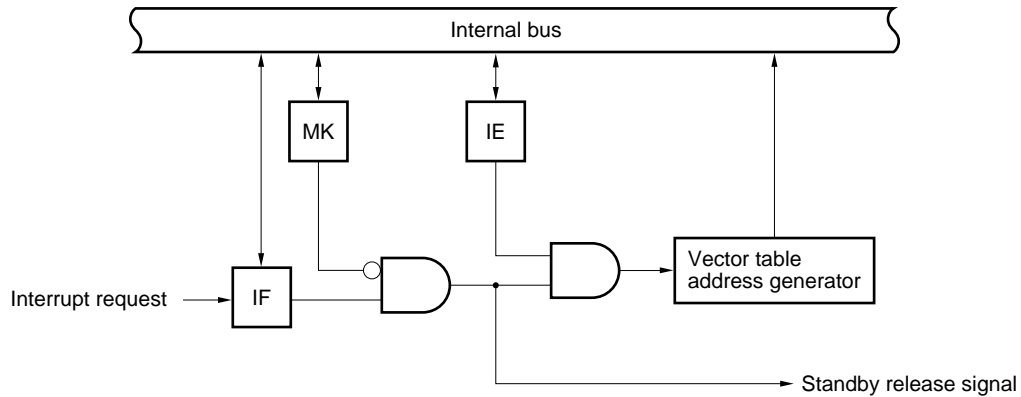
**Remark** As the interrupt source of the watchdog timer (INTWDT), either a non-maskable interrupt or a maskable interrupt (internal) can be selected.

Figure 15-1. Basic Configuration of Interrupt Function

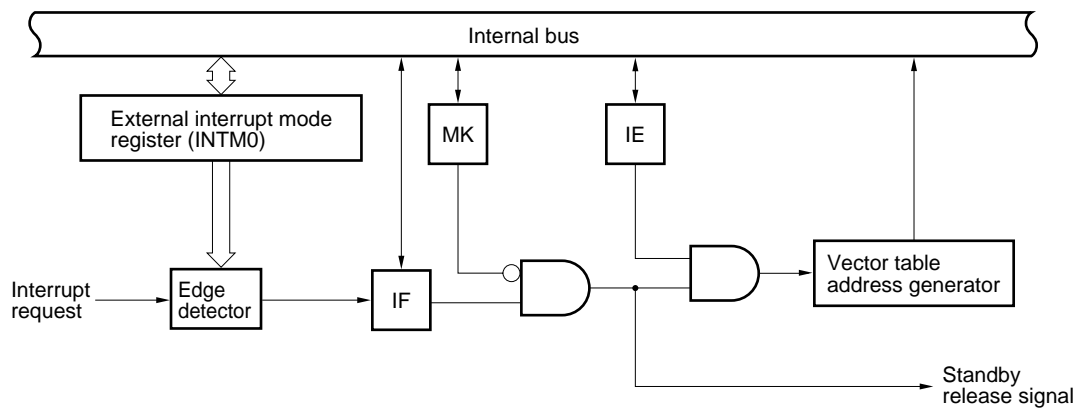
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt



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

### 15.3 Interrupt Function Control Registers

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

- Interrupt request flag registers (IF0, IF1)
- Interrupt mask flag registers (MK0, MK1)
- External interrupt mode register (INTM0)
- Program status word (PSW)

Table 15-2 gives a listing of interrupt request flag and interrupt mask flag names corresponding to interrupt requests.

**Table 15-2. Flags Corresponding to Interrupt Request Signal**

Interrupt Request Signal Name	Interrupt Request Flag	Interrupt Mask Flag
INTWDT	TMIF4	TMMK4
INTP0	PIF0	PMK0
INTP1	PIF1	PMK1
INTP2	PIF2	PMK2
INTSR20/INTCSI20	SRIF20	SRMK20
INTST20	STIF20	STMK20
INTTM80	TMIF80	TMMK80
INTTM20	TMIF20	TMMK20
INTAD0	ADIF0	ADMK0

### (1) Interrupt request flag registers (IF0, 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 when an instruction is executed upon acknowledgement of an interrupt request or upon  $\overline{\text{RESET}}$  input.

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

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

**Figure 15-2. Interrupt Request Flag Register Format**

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF0	TMIF20	TMIF80	STIF20	SRIF20	PIF2	PIF1	PIF0	TMIF4	FFE0H	00H	R/W

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
IF1	0	0	0	0	0	0	0	ADIF0	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. TMIF4 flag is R/W enabled only when the watchdog timer is used as an interval timer. If the watchdog timer mode 1 and 2 are used, set the TMIF4 flag to 0.
  2. Because port 2 has an alternate function as the external interrupt input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set. Therefore, the interrupt mask flag should be set to 1 before using the output mode.

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

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

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

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

**Figure 15-3. Interrupt Mask Flag Register Format**

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
MK0	TMMK20	TMMK80	STMK20	SRMK20	PMK2	PMK1	PMK0	TMMK4	FFE4H	FFH	R/W

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
MK1	1	1	1	1	1	1	1	ADMK0	FFE5H	FFH	R/W

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

- Cautions**
1. If the TMMK4 flag is read when the watchdog timer is used in watchdog timer mode 1 and 2, its value becomes undefined.
  2. Because port 2 has an alternate function as the external interrupt input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set. Therefore, 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.

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

**Figure 15-4. External Interrupt Mode Register 0 Format**

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

ES21	ES20	INTP2 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES11	ES10	INTP1 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES01	ES00	INTP0 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

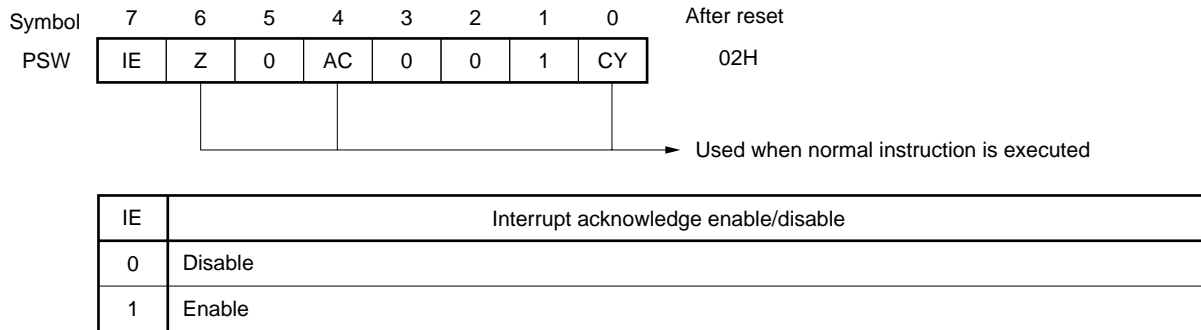
**Cautions** 1. Be sure to set bits 0 and 1 to 0.

- 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 ( $\times\times\text{MK}\times = 0$ ), which will enable interrupts.

**(4) Program status word (PSW)**

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

This register can be read/written in 8-bit units and can carry out operations using a bit manipulation and dedicated instructions (EI, DI). When a vectored interrupt request is acknowledged, PSW is automatically saved into a stack, and the IE flag is reset to 0. It is reset from the stack by the RETI and POP PSW instructions.  $\overline{\text{RESET}}$  input sets PSW to 02H.

**Figure 15-5. Program Status Word Configuration**

## 15.4 Interrupt Processing Operation

### 15.4.1 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, 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 15-6 shows the flowchart from non-maskable interrupt request generation to acknowledgement. Figure 15-7 shows the timing of non-maskable interrupt request acknowledgement. Figure 15-8 shows the acknowledgement operation if multiple non-maskable interrupts are generated.

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

Figure 15-6. Flowchart from Non-Maskable Interrupt Request Generation to Acknowledgement

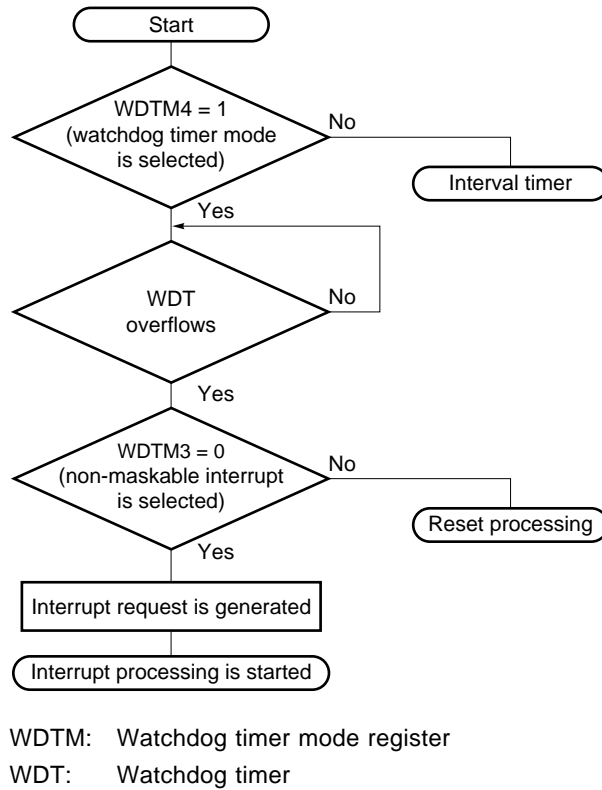


Figure 15-7. Timing of Non-Maskable Interrupt Request Acknowledgement

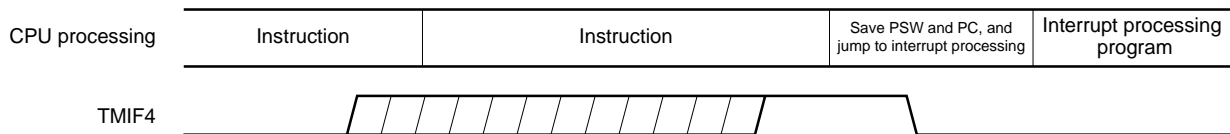
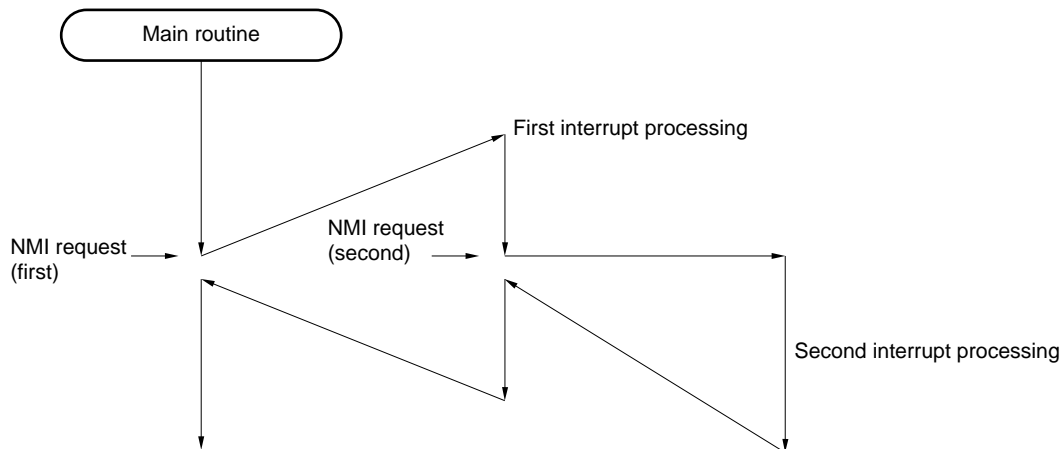


Figure 15-8. Acknowledging Non-Maskable Interrupt Request



### 15.4.2 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 processing after a maskable interrupt request has been generated is shown in Table 15-3.

Refer to Figures 15-10 and 15-11 for the interrupt request acknowledgement timing.

**Table 15-3. Time from Generation of Maskable Interrupt Request to Processing**

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 BT and BF instruction.

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

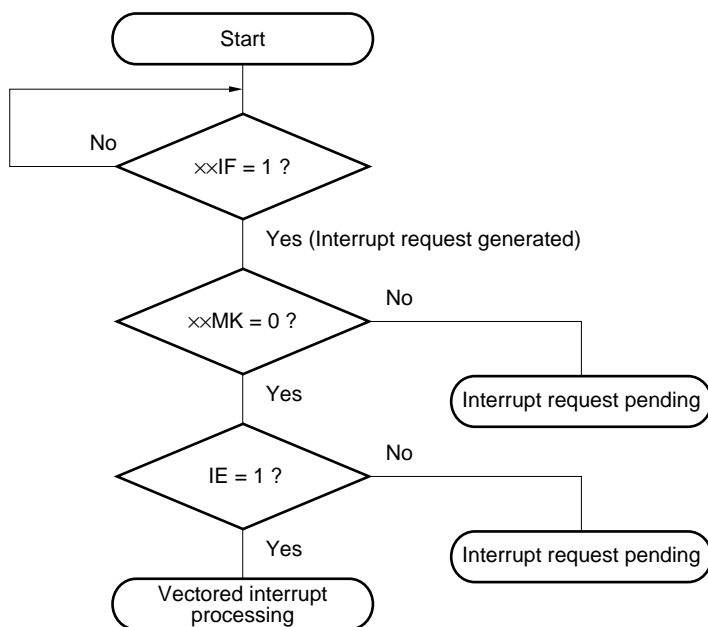
When two or more maskable interrupt requests are generated at the same time, they are acknowledged starting from the interrupt request assigned the highest priority.

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

Figure 15-9 shows the algorithm of acknowledging interrupt requests.

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

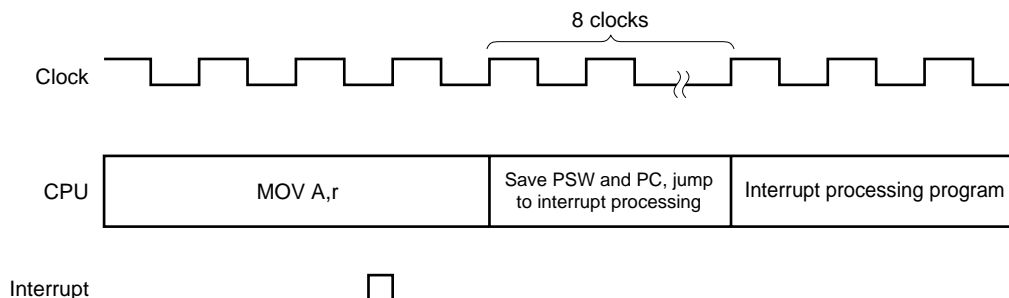
To return from interrupt processing, use the RETI instruction.

**Figure 15-9. Interrupt Acknowledgement Program Algorithm**

xxIF: Interrupt request flag

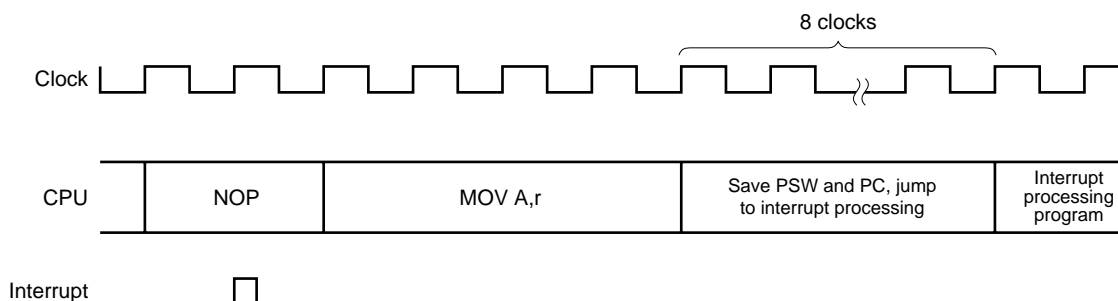
xxMK: Interrupt mask flag

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

**Figure 15-10. Interrupt Request Acknowledgement Timing (Example of MOV A,r)**

If an interrupt request flag ( $\times\times IF$ ) is set before an instruction clock  $n$  ( $n = 4$  to  $10$ ) under execution becomes  $n-1$ , the interrupt is acknowledged after the instruction under execution is complete. Figure 15-10 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 for 3 clocks after the execution starts, the interrupt acknowledgement processing is performed after the MOV A,r instruction is completed.

**Figure 15-11. Interrupt Request Acknowledgement Timing  
(When Interrupt Request Flag Is Generated at the  
Last Clock During Instruction Execution)**



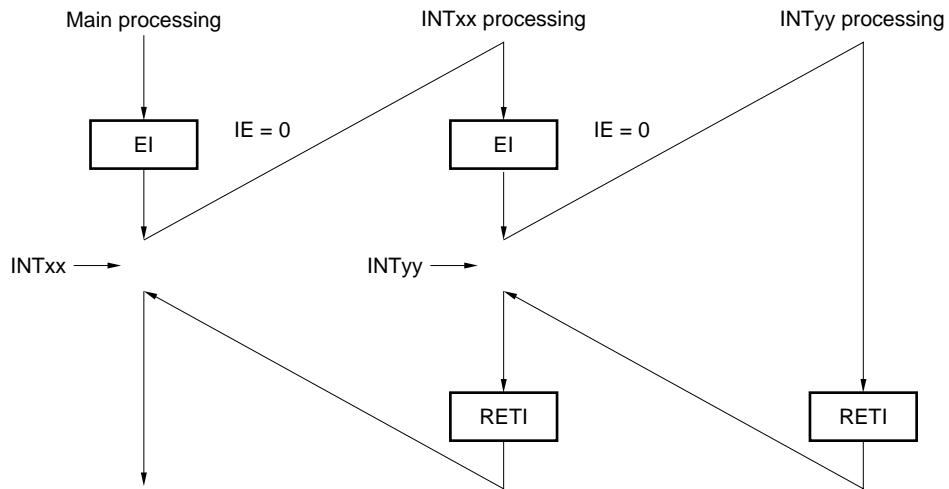
If an interrupt request flag ( $\times\times IF$ ) is set at the last clock of the instruction, the interrupt acknowledgement processing starts after the next instruction is executed.

Figure 15-11 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.

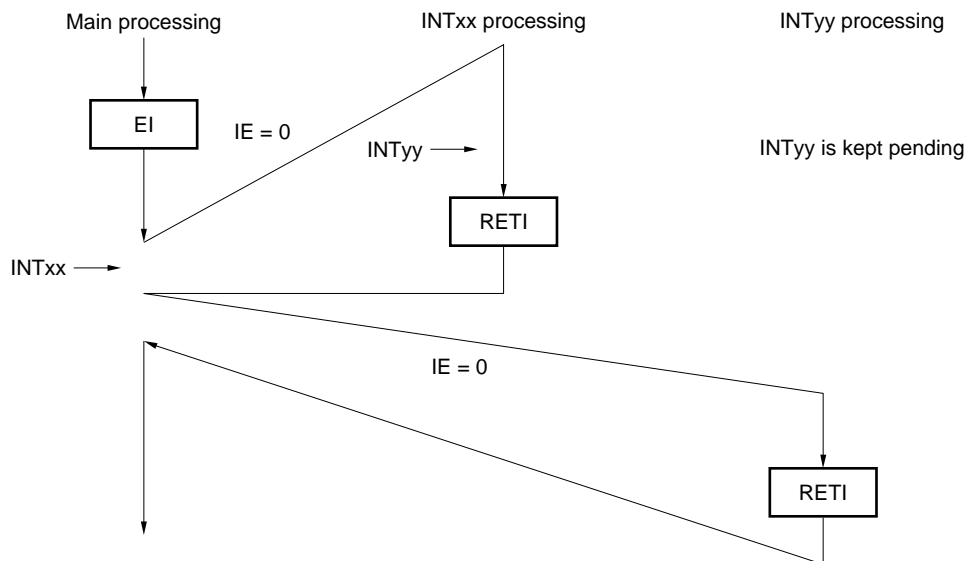
**Caution** Interrupt requests are reserved while the interrupt request flag register (IF0, IF1) or the interrupt mask flag register (MK0, MK1) is being accessed.

### 15.4.3 Multiple interrupt processing

Multiple interrupt processing in which another interrupt is acknowledged while an interrupt is being processed can be processed by priority. When the priority is controlled by the default priority and two or more interrupts are generated at once, interrupt processing is performed according to the priority assigned to each interrupt request in advance (refer to **Table 15-1**).

**Figure 15-12. Example of Multiple Interrupts****Example 1. Multiple interrupts are acknowledged**

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

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

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

IE = 0: Interrupt request acknowledgement disabled



#### 15.4.4 Interrupt request reserve

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

- Manipulation instruction for the interrupt request flag register (IF0, IF1)
- Manipulation instruction for the interrupt mask flag register (MK0, MK1)

**[MEMO]**

## CHAPTER 16 STANDBY FUNCTION

### 16.1 Standby Function and Configuration

#### 16.1.1 Standby function

The standby function is 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 oscillation circuit continues oscillating. This mode does not reduce the power 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 main system clock oscillation circuit and stops the entire system. The power consumption of the CPU can be substantially reduced in this mode.

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

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

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

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

### 16.1.2 Standby function control register ( $\mu$ PD789104A, 789114A Subseries)

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

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  $2^{17}/f_x$ .

**Note**  $\mu$ PD789104A and 789114A Subseries only.

The  $\mu$ PD789124A and 789134A Subseries do not provide an oscillation stabilization time select register.

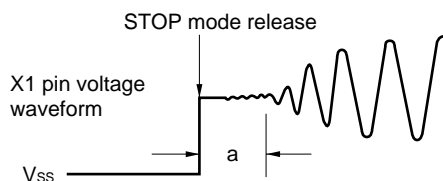
The oscillation stabilization time of the  $\mu$ PD789124A and 789134A Subseries is fixed to  $2^7/f_{cc}$ .

**Figure 16-1. Oscillation Stabilization Time Select Register Format**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection
0	0	0	$2^{12}/f_x$ (819 $\mu$ 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 in a ceramic/crystal oscillator 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 (ceramic/crystal oscillation)
  2. Values in parentheses apply to operation with  $f_x = 5.0$  MHz.

## 16.2 Operation of Standby Function

### 16.2.1 HALT mode

#### (1) HALT mode

The HALT mode is set by executing the HALT instruction.

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

**Table 16-1. HALT Mode Operating Status**

Item	HALT Mode Operating Status
Clock generator	System clock can be oscillated. Clock supply to CPU stops.
CPU	Operation stopped
Port (Output latch)	Retains the status before setting the HALT mode.
16-bit timer 20	Operable
8-bit timer/event counter 80	Operable
Watchdog timer	Operable
Serial interface 20	Operable
A/D converter	Operation stopped
Multiplier	Operation stopped
External interrupt	Operable <b>Note</b>

**Note** Maskable interrupt that is not masked

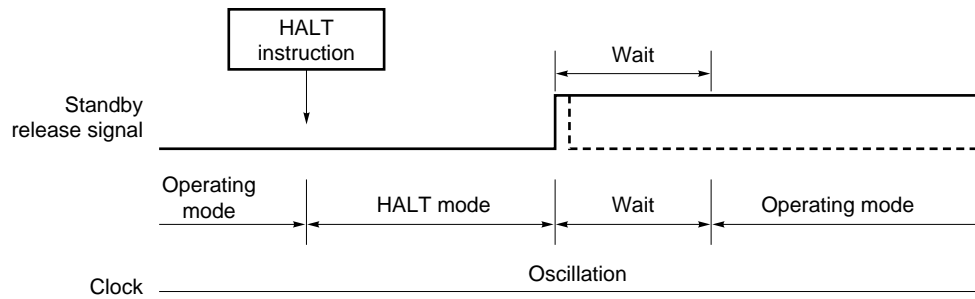
**(2) Releasing HALT mode**

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

**(a) Releasing by unmasked interrupt request**

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

**Figure 16-2. 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:

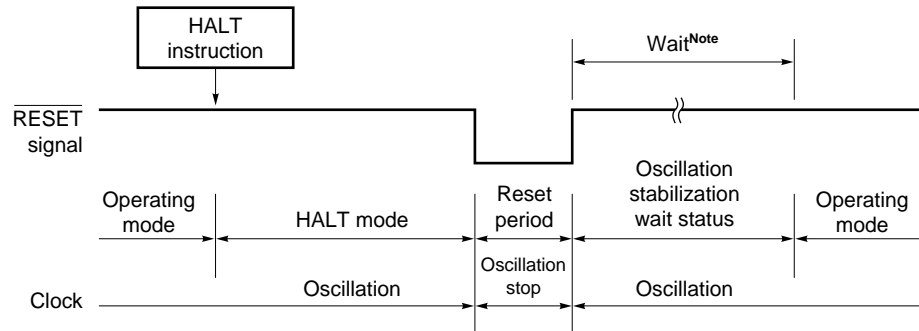
- When vectored interrupt processing is performed: 9 to 10 clocks
- When vectored interrupt processing is not performed: 1 to 2 clocks

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

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

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

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

**Figure 16-3. Releasing HALT Mode by  $\overline{\text{RESET}}$  Input**

**Note** In the  $\mu\text{PD789104A}$  and 789114A Subseries,  $2^{15}/f_x$ : 6.55 ms (at  $f_x = 5.0\text{-MHz}$  operation)  
 In the  $\mu\text{PD789124A}$  and 789134A Subseries,  $2^7/f_{cc}$ : 32  $\mu\text{s}$  (at  $f_{cc} = 4.0\text{-MHz}$  operation)

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)  
 $f_{cc}$ : System clock oscillation frequency (RC oscillation)

**Table 16-2. Operation after Release of HALT Mode**

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

$\times$ : don't care

## 16.2.2 STOP mode

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

The STOP mode is set by executing the STOP instruction.

- Cautions**
1. When the STOP mode is set, the X2 or CL2 pin is internally pulled up to  $V_{DD}$  to suppress the current leakage of the oscillation circuit block. Therefore, do not use the STOP mode in a system where the external clock is used as the system clock.
  2. Because the standby mode can be released by an interrupt request signal, the standby mode is released as soon as it is set if there is an interrupt source whose interrupt request flag is set and interrupt mask flag is reset. When the STOP mode is set, therefore, the HALT mode is set immediately after the STOP instruction has been executed, the wait time set by the oscillation stabilization time select register (OSTS) elapses, and then an operation mode is set.

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

**Table 16-3. STOP Mode Operating Status**

Item	STOP Mode Operating Status
Clock generator	System clock oscillation stopped
CPU	Operation stopped
Port (Output latch)	Retains the status before setting the STOP mode
16-bit timer 20	Operation stopped
8-bit timer/event counter 80	Operable <b>Note 1</b>
Watchdog timer	Operation stopped
Serial interface 20	Operable <b>Note 2</b>
A/D converter	Operation stopped
Multiplier	Operation stopped
External interrupt	Operable <b>Note 3</b>

- Notes**
1. Operation is possible only when TI80 is selected as the count clock.
  2. Operation is possible in both 3-wire serial I/O and UART modes while an external clock is being used.
  3. Maskable interrupt that is not masked



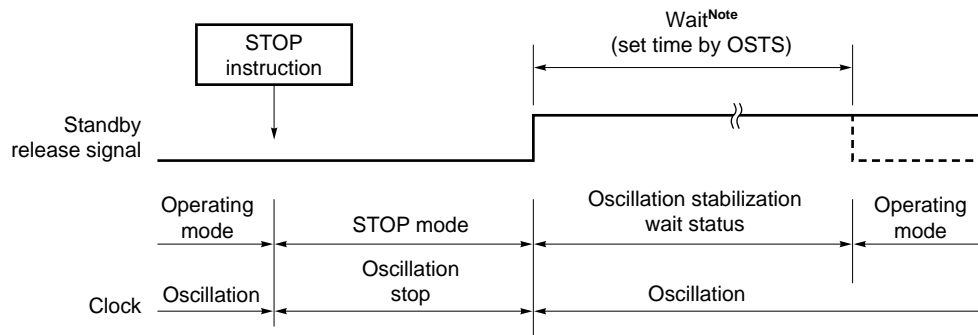
**(2) Releasing STOP mode**

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 able to be acknowledged, vectored interrupt processing is performed, after the oscillation stabilization time has elapsed. If the interrupt is disabled, the instruction at the next address is executed.

**Figure 16-4. Releasing STOP Mode by Interrupt**

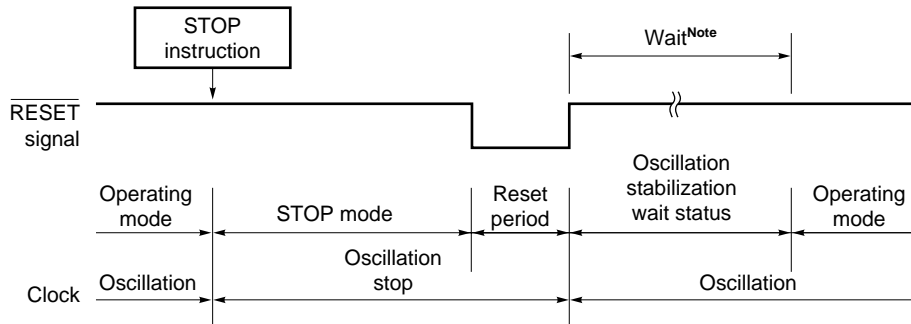


**Note** OSTs is not provided in the  $\mu$ PD789124A and 789134A Subseries, and the wait time is fixed to  $2^7/f_{cc}$ .

**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 16-5. Releasing STOP Mode by  $\overline{\text{RESET}}$  Input**

**Note** In the  $\mu\text{PD789104A}$  and 789114A Subseries,  $2^{15}/f_x$ : 6.55 ms (at  $f_x = 5.0\text{-MHz}$  operation)  
 In the  $\mu\text{PD789124A}$  and 789134A Subseries,  $2^7/f_{cc}$ : 32  $\mu\text{s}$  (at  $f_{cc} = 4.0\text{-MHz}$  operation)

**Remark**  $f_x$ : System clock oscillation frequency (ceramic/crystal oscillation)  
 $f_{cc}$ : System clock oscillation frequency (RC oscillation)

**Table 16-4. 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 processing
	1	$\times$	Retains STOP mode
$\overline{\text{RESET}}$ input	—	—	Reset processing

$\times$ : don't care

## CHAPTER 17 RESET FUNCTION

The following two operations are available to generate reset signals.

- (1) External reset input with  $\overline{\text{RESET}}$  pin
- (2) Internal reset by program runaway time detection with watchdog timer

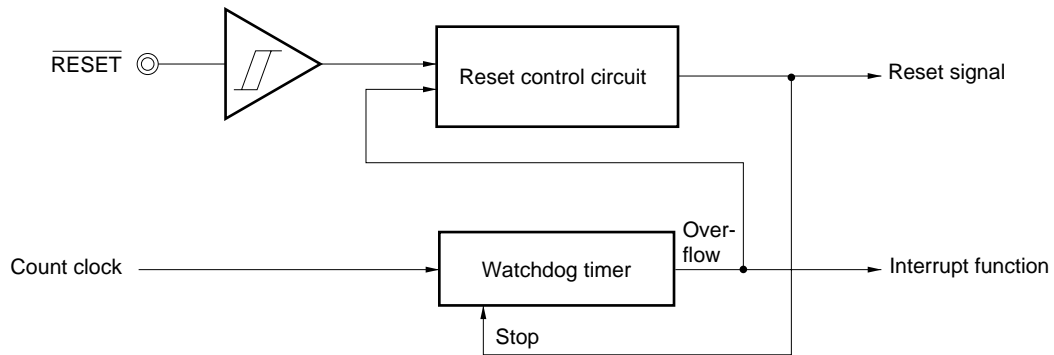
External and internal reset have no functional differences. In both cases, program execution starts at the addresses 0000H and 0001H by reset signal input.

When a low level is input to the  $\overline{\text{RESET}}$  pin or the watchdog timer overflows, a reset is applied and each hardware item is set to the status shown in Table 17-1. Each pin has a high impedance during reset input or during the oscillation stabilization time just after reset clear.

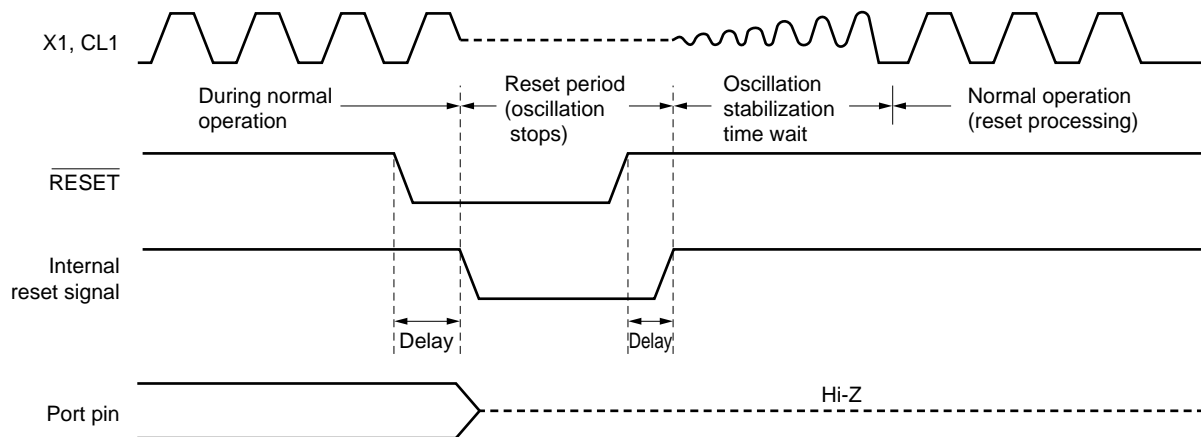
When a high level is input to the  $\overline{\text{RESET}}$  pin, the reset is cleared and program execution is started after the oscillation stabilization time has elapsed. The reset applied by the watchdog timer overflow is automatically cleared after reset, and program execution is started after the oscillation stabilization time has elapsed (see **Figures 17-2** through **17-4**).

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

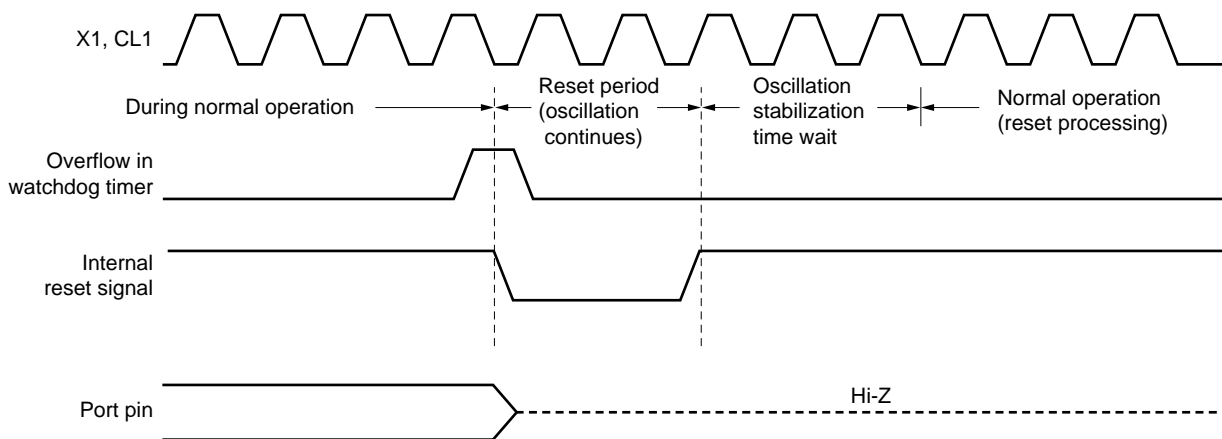
Figure 17-1. Block Diagram of Reset Function



**Figure 17-2. Reset Timing by  $\overline{\text{RESET}}$  Input**



**Figure 17-3. Reset Timing by Overflow in Watchdog Timer**



**Figure 17-4. Reset Timing by  $\overline{\text{RESET}}$  Input in STOP Mode**

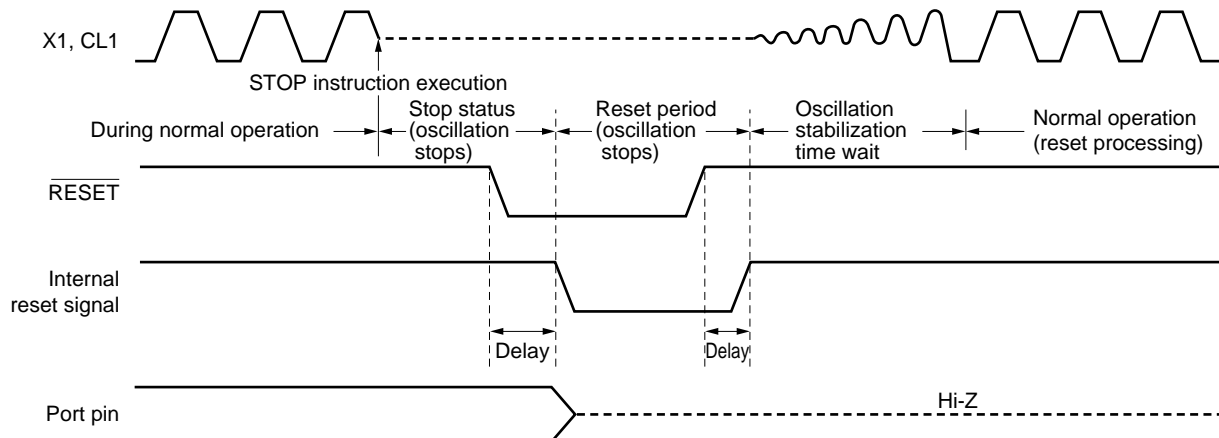


Table 17-1. Hardware Status after Reset (1/2)

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 register	Undefined <sup>Note 2</sup>
Port (P0 to P2, P5) (Output latch)		00H
Port mode register (PM0 to PM2, PM5)		FFH
Pull-up resistor option register 0 (PU0)		00H
Pull-up resistor option register B2 (PUB2)		00H
Processor clock control register (PCC)		02H
Oscillation stabilization time select register (OSTS) <sup>Note 3</sup>		04H
16-bit timer 20	Timer counter (TM20)	0000H
	Compare register (CR20)	FFFFH
	Mode control register (TMC20)	00H
	Capture register (TPC20)	Undefined
8-bit timer/event counter 80	Timer counter (TM80)	00H
	Compare register (CR80)	Undefined
	Mode control register (TMC80)	00H
Watchdog timer	Timer clock select register (TCL2)	00H
	Mode register (WDTM)	00H
A/D converter	Mode register (ADM0)	00H
	Input channel specification register (ADS0)	00H
	Conversion result register (ADCR0)	Undefined
Serial interface 20	Mode register (CSIM20)	00H
	Asynchronous serial interface mode register (ASIM20)	00H
	Asynchronous serial interface status register (ASIS20)	00H
	Baud rate generator control register (BRGC20)	00H
	Transmit shift register (TXS20)	FFH
	Receive buffer register (RXB20)	Undefined

- 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. If the reset signal is input in the standby mode, the status before reset is retained even after reset.
  3.  $\mu$ PD789104A, 789114A Subseries only

**Table 17-1. Hardware Status after Reset (2/2)**

Hardware		Status after Reset
Multiplier	16-bit multiplication result storage register (MUL0)	Undefined
	Data register A (MRA0)	Undefined
	Data register B (MRB0)	Undefined
	Control register (MULC0)	00H
Interrupt	Request flag register (IF0, IF1)	00H
	Mask flag register (MK0, MK1)	FFH
	External interrupt mode register (INTM0)	00H

## CHAPTER 18 $\mu$ PD78F9116A, 78F9136A

The  $\mu$ PD78F9116A is a version with flash memory instead of the internal ROM of the mask ROM version in the  $\mu$ PD789104A and 789114A Subseries. The  $\mu$ PD78F9136A is a version with flash memory instead of the internal ROM of the mask ROM version in the  $\mu$ PD789124A and 789134A Subseries. The differences between the flash memory and the mask ROM versions are shown in Table 18-1.

**Table 18-1. Differences between Flash Memory and Mask ROM Versions**

Item		Flash Memory	Mask ROM		
		$\mu$ PD78F9116A	$\mu$ PD789101A $\mu$ PD789111A	$\mu$ PD789102A $\mu$ PD789112A	$\mu$ PD789104A $\mu$ PD789114A
		$\mu$ PD78F9136A	$\mu$ PD789121A $\mu$ PD789131A	$\mu$ PD789122A $\mu$ PD789132A	$\mu$ PD789124A $\mu$ PD789134A
Internal memory	ROM	16 Kbytes (flash memory)	2 Kbytes	4 Kbytes	8 Kbytes
	High-speed RAM	256 bytes			
Pull-up resistor		12 (software control only)	16 (software control: 12, mask option specification: 4)		
$V_{PP}$ pin		Provided	Not provided		
Electrical specifications		Refer to the individual data sheets.			

- Cautions**
- There are differences in noise immunity and noise radiation between the flash memory versions 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 versions.
  - A/D conversion result register 0 (ADCR0) is manipulated by an 8-bit memory manipulation instruction or a 16-bit memory manipulation instruction, when used as an 8-bit A/D converter ( $\mu$ PD789104A, 789124A Subseries) or 10-bit A/D converter ( $\mu$ PD789114A, 789134A Subseries), respectively.  
However, if the  $\mu$ PD78F9116A is used as the flash memory version of the  $\mu$ PD789101A, 789102A, 789104A, ADCR0 can be manipulated by an 8-bit memory manipulation instruction, providing an object file has been assembled in the  $\mu$ PD789101A, 789102A, 789104A. If the  $\mu$ PD78F9136A is used as the flash memory version of the  $\mu$ PD789121A, 789122A, 789124A, ADCR0 can be manipulated by an 8-bit memory manipulation instruction, providing an object file has been assembled in the  $\mu$ PD789121A, 789122A, 789124A.

## 18.1 Flash Memory Programming

The on-chip program memory in the  $\mu$ PD78F9116A, 78F9136A is flash memory.

The flash memory can be written with the  $\mu$ PD78F9116A mounted on the target system (on-board). Connect the dedicated flash programmer (Flashpro III (part number: FL-PR3, PG-FP3)) to the host machine and target system to write the flash memory.

**Remark** FL-PR3 is made by Naito Densai Machida Mfg. Co., Ltd.

### 18.1.1 Selecting communication mode

The flash memory is written by using Flashpro III and by means of serial communication. Select a communication mode from those listed in Table 18-2. To select a communication mode, the format shown in Figure 18-1 is used. Each communication mode is selected by the number of  $V_{PP}$  pulses shown in Table 18-2.

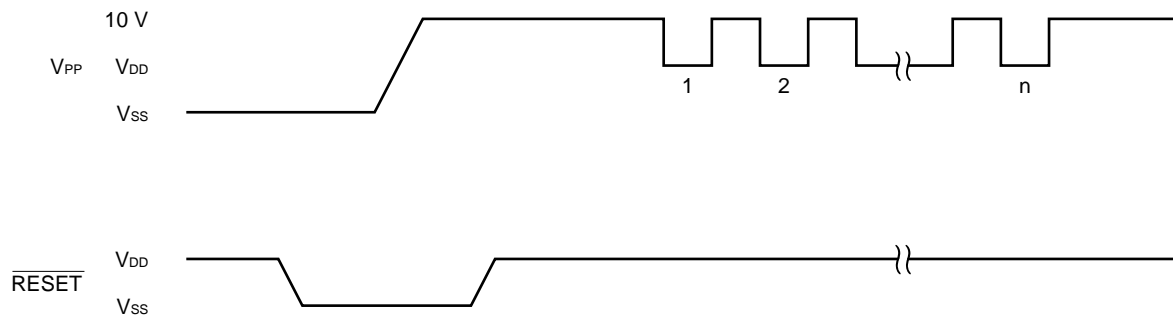
**Table 18-2. Communication Mode**

Communication Mode	Pins Used	Number of $V_{PP}$ Pulses
3-wire serial I/O	SCK20/ASCK20/P20 SO20/TxD20/P21 SI20/RxD20/P22	0
UART	TxD20/SO20/P21 RxD20/SI20/P22	8
Pseudo 3-wire mode <b>Note</b>	P00 (Serial clock input) P01 (Serial data output) P02 (Serial data input)	12

**Note** Serial transfer is performed by controlling a port by software.

**Caution** Be sure to select a communication mode based on the  $V_{PP}$  pulse number shown in Table 18-2.

**Figure 18-1. Communication Mode Selection Format**





### 18.1.2 Function of flash memory programming

By transmitting/receiving commands and data in the selected communication mode, operations such as writing to the flash memory are performed. Table 18-3 shows the major functions of flash memory programming.

**Table 18-3. Functions of Flash Memory Programming**

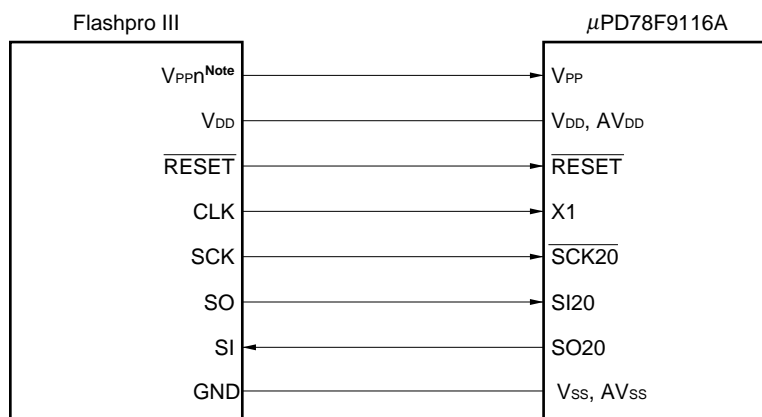
Function	Description
Batch erase	Erases all contents of memory
Batch blank check	Checks erased state of entire memory
Data write	Write to flash memory based on write start address and number of data written (number of bytes)
Batch verify	Compares all contents of memory with input data

### 18.1.3 Flashpro III connection example

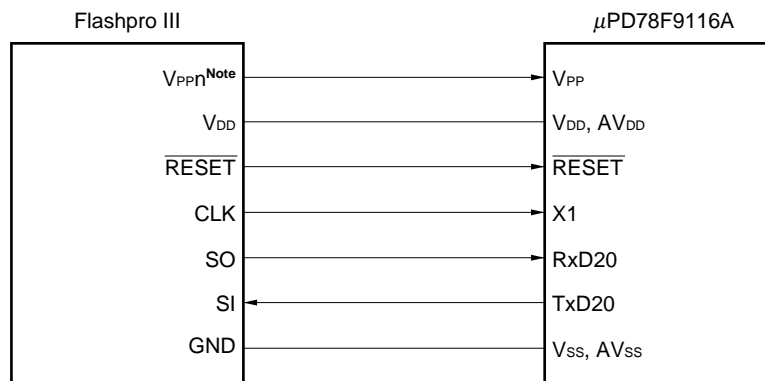
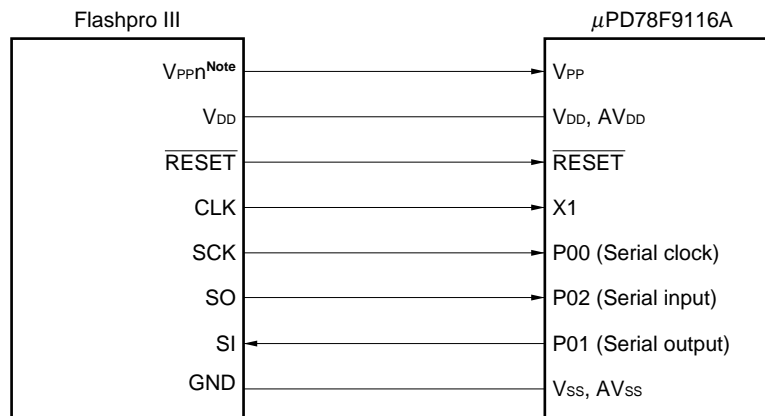
Connection between the Flashpro III and the  $\mu$ PD78F9116A and 78F9136A differs depending on the communication mode (3-wire serial I/O, UART, or pseudo 3-wire mode). Figures 18-2 to 18-4 show the connection examples in the respective modes.

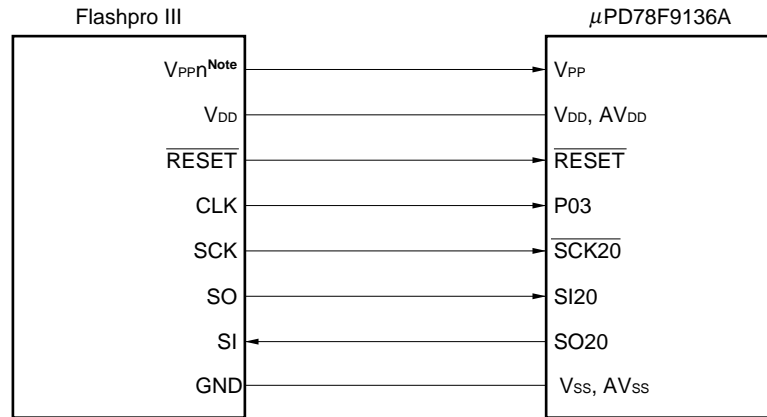
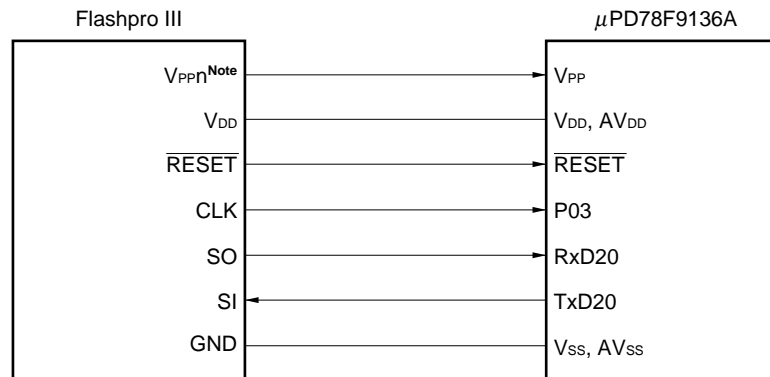
#### (a) Connection between $\mu$ PD78F9116A and Flashpro III

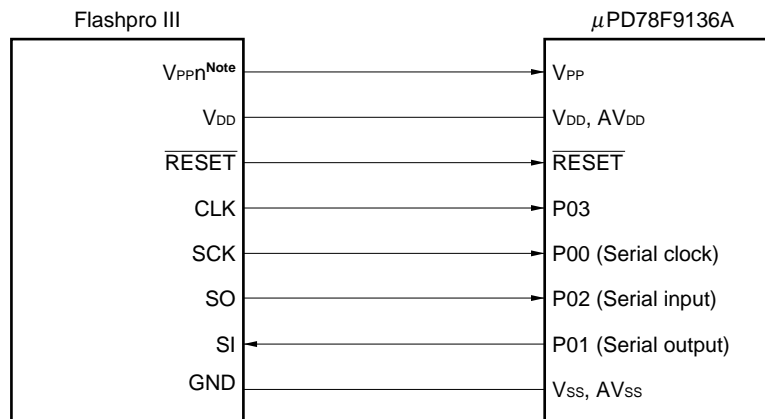
**Figure 18-2. Flashpro III Connection in 3-Wire Serial I/O Mode**



**Note**  $n = 1, 2$

**Figure 18-3. Flashpro III Connection in UART Mode****Note**  $n = 1, 2$ **Figure 18-4. Flashpro III Connection in Pseudo 3-Wire Mode (When P0 Is Used)****Note**  $n = 1, 2$

(b) Connection between  $\mu$ PD78F9136A and Flashpro III**Figure 18-5. Flashpro III Connection in 3-Wire Serial I/O Mode****Note**  $n = 1, 2$ **Figure 18-6. Flashpro III Connection in UART Mode****Note**  $n = 1, 2$

**Figure 18-7. Flashpro III Connection in Pseudo 3-Wire Mode (When P0 Is Used)**

**Note**  $n = 1, 2$

**18.1.4 Example of settings for Flashpro III (PG-FP3)**

Make the following setting when writing to flash memory using Flashpro III (PG-FP3).

- <1> Load the parameter file.
- <2> Select serial mode and serial clock using the type command.
- <3> An example of the settings for the PG-FP3 is shown below.

**Table 18-4. Example of Settings for PG-FP3**

Communication Mode	Example of Setting for PG-FP3		Number of V <sub>PP</sub> Pulses <sup>Note 1</sup>
3-wire serial I/O	COMM PORT	SIO-ch0	0
	CPU CLK	On Target Board ----- In Flashpro	
	On Target Board	4.1943 MHz	
	SIO CLK	1.0 MHz	
	In Flashpro	4.0 MHz	
	SIO CLK	1.0 MHz	
UART	COMM PORT	UART-ch0	8
	CPU CLK	On Target Board	
	On Target Board	4.1943 MHz	
	UART BPS	9600 bps <sup>Note 2</sup>	
Pseudo 3-wire	COMM PORT	Port A	12
	CPU CLK	On Target Board ----- In Flashpro	
	On Target Board	4.1943 MHz	
	SIO CLK	1.0 kHz	
	In Flashpro	4.0 MHz	
	SIO CLK	1.0 kHz	

- Notes**
1. The number of V<sub>PP</sub> pulses supplied from Flashpro III when serial communication is initialized. The pins to be used for communication are determined according to the number of these pulses.
  2. Select one of 9600 bps, 19200 bps, 38400 bps, or 76800 bps.

**Remark**

COMM PORT: Selection of serial port  
 SIO CLK: Selection of serial clock frequency  
 CPU CLK: Selection of source of CPU clock to be input

[MEMO]

## CHAPTER 19 MASK OPTION (MASK ROM VERSION)

**Table 19-1. Selection of Mask Option for Pins**

Pin	Mask Option
P50 to P53	On-chip pull-up resistor can be specified in 1-bit units.

For P50 to P53 (port 5), an on-chip pull-up resistor can be specified by the mask option. The mask option is specified in 1-bit units.

**Caution** The flash memory versions do not provide the on-chip pull-up resistor function.

[MEMO]



## CHAPTER 20 INSTRUCTION SET

This chapter lists the instruction set of the  $\mu$ PD789104A/114A/124A/134A Subseries. For the details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series User's Manual Instruction (U11047E)**.

### 20.1 Operation

#### 20.1.1 Operand identifiers and description methods

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

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

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

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

**Table 20-1. Operand Identifiers and Description Methods**

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

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

**20.1.2 Description of “operation” column**

A:	A register; 8-bit accumulator
X:	X register
B:	B register
C:	C register
D:	D register
E:	E register
H:	H register
L:	L register
AX:	AX register pair; 16-bit accumulator
BC:	BC register pair
DE:	DE register pair
HL:	HL register pair
PC:	Program counter
SP:	Stack pointer
PSW:	Program status word
CY:	Carry flag
AC:	Auxiliary carry flag
Z:	Zero flag
IE:	Interrupt request enable flag
NMIS:	Flag indicating non-maskable interrupt servicing in progress
( ):	Memory contents indicated by address or register contents in parenthesis
x <sub>H</sub> , x <sub>L</sub> :	Higher 8 bits and lower 8 bits of 16-bit register
∧:	Logical product (AND)
∨:	Logical sum (OR)
⊕:	Exclusive logical sum (exclusive OR)
—:	Inverted data
addr16:	16-bit immediate data or label
jdisp8:	Signed 8-bit data (displacement value)

**20.1.3 Description of “flag operation” column**

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

## 20.2 Operation List

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

**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	Byte	Clock	Operation	Flag
					Z AC CY
<b>SUBC</b>	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,laddr16	3	8	$A, CY \leftarrow A - (\text{laddr16}) - CY$	× × ×
	A,[HL]	1	6	$A, CY \leftarrow A - (HL) - CY$	× × ×
	A,[HL+byte]	2	6	$A, CY \leftarrow A - (HL+\text{byte}) - CY$	× × ×
<b>AND</b>	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,laddr16	3	8	$A \leftarrow A \wedge (\text{laddr16})$	×
	A,[HL]	1	6	$A \leftarrow A \wedge (HL)$	×
	A,[HL+byte]	2	6	$A \leftarrow A \wedge (HL+\text{byte})$	×
<b>OR</b>	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,laddr16	3	8	$A \leftarrow A \vee (\text{laddr16})$	×
	A,[HL]	1	6	$A \leftarrow A \vee (HL)$	×
	A,[HL+byte]	2	6	$A \leftarrow A \vee (HL+\text{byte})$	×
<b>XOR</b>	A,#byte	2	4	$A \leftarrow A \veebar \text{byte}$	×
	saddr,#byte	3	6	$(saddr) \leftarrow (saddr) \veebar \text{byte}$	×
	A,r	2	4	$A \leftarrow A \veebar r$	×
	A,saddr	2	4	$A \leftarrow A \veebar (saddr)$	×
	A,laddr16	3	8	$A \leftarrow A \veebar (\text{laddr16})$	×
	A,[HL]	1	6	$A \leftarrow A \veebar (HL)$	×
	A,[HL+byte]	2	6	$A \leftarrow A \veebar (HL+\text{byte})$	×

**Remark** One instruction clock cycle is one CPU clock cycle ( $f_{CPU}$ ) selected by the Processor Clock Control Register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag
					Z AC CY
<b>CMP</b>	A,#byte	2	4	$A - \text{byte}$	× × ×
	saddr,#byte	3	6	$(\text{saddr}) - \text{byte}$	× × ×
	A,r	2	4	$A - r$	× × ×
	A,saddr	2	4	$A - (\text{saddr})$	× × ×
	A,!addr16	3	8	$A - (\text{addr16})$	× × ×
	A,[HL]	1	6	$A - (\text{HL})$	× × ×
	A,[HL+byte]	2	6	$A - (\text{HL} + \text{byte})$	× × ×
<b>ADDW</b>	AX,#word	3	6	$\text{AX}, \text{CY} \leftarrow \text{AX} + \text{word}$	× × ×
<b>SUBW</b>	AX,#word	3	6	$\text{AX}, \text{CY} \leftarrow \text{AX} - \text{word}$	× × ×
<b>CMPW</b>	AX,#word	3	6	$\text{AX} - \text{word}$	× × ×
<b>INC</b>	r	2	4	$r \leftarrow r + 1$	× ×
	saddr	2	4	$(\text{saddr}) \leftarrow (\text{saddr}) + 1$	× ×
<b>DEC</b>	r	2	4	$r \leftarrow r - 1$	× ×
	saddr	2	4	$(\text{saddr}) \leftarrow (\text{saddr}) - 1$	× ×
<b>INCW</b>	rp	1	4	$\text{rp} \leftarrow \text{rp} + 1$	
<b>DECW</b>	rp	1	4	$\text{rp} \leftarrow \text{rp} - 1$	
<b>ROR</b>	A,1	1	2	$(\text{CY}, \text{A}_7 \leftarrow \text{A}_0, \text{A}_{m-1} \leftarrow \text{A}_m) \times 1$	×
<b>ROL</b>	A,1	1	2	$(\text{CY}, \text{A}_0 \leftarrow \text{A}_7, \text{A}_{m+1} \leftarrow \text{A}_m) \times 1$	×
<b>RORC</b>	A,1	1	2	$(\text{CY} \leftarrow \text{A}_0, \text{A}_7 \leftarrow \text{CY}, \text{A}_{m-1} \leftarrow \text{A}_m) \times 1$	×
<b>ROLC</b>	A,1	1	2	$(\text{CY} \leftarrow \text{A}_7, \text{A}_0 \leftarrow \text{CY}, \text{A}_{m+1} \leftarrow \text{A}_m) \times 1$	×
<b>SET1</b>	saddr.bit	3	6	$(\text{saddr.bit}) \leftarrow 1$	
	sfr.bit	3	6	$\text{sfr.bit} \leftarrow 1$	
	A.bit	2	4	$\text{A.bit} \leftarrow 1$	
	PSW.bit	3	6	$\text{PSW.bit} \leftarrow 1$	× × ×
	[HL].bit	2	10	$(\text{HL}).\text{bit} \leftarrow 1$	
<b>CLR1</b>	saddr.bit	3	6	$(\text{saddr.bit}) \leftarrow 0$	
	sfr.bit	3	6	$\text{sfr.bit} \leftarrow 0$	
	A.bit	2	4	$\text{A.bit} \leftarrow 0$	
	PSW.bit	3	6	$\text{PSW.bit} \leftarrow 0$	× × ×
	[HL].bit	2	10	$(\text{HL}).\text{bit} \leftarrow 0$	
<b>SET1</b>	CY	1	2	$\text{CY} \leftarrow 1$	1
<b>CLR1</b>	CY	1	2	$\text{CY} \leftarrow 0$	0
<b>NOT1</b>	CY	1	2	$\text{CY} \leftarrow \overline{\text{CY}}$	×

**Remark** One instruction clock cycle is one CPU clock cycle ( $f_{\text{CPU}}$ ) selected by the Processor Clock Control Register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag
					Z AC CY
<b>CALL</b>	!addr16	3	6	$(SP - 1) \leftarrow (PC + 3)_H, (SP - 2) \leftarrow (PC + 3)_L,$ $PC \leftarrow \text{addr16}, SP \leftarrow SP - 2$	
<b>CALLT</b>	[addr5]	1	8	$(SP - 1) \leftarrow (PC + 1)_H, (SP - 2) \leftarrow (PC + 1)_L,$ $PC_H \leftarrow (00000000, \text{addr5} + 1),$ $PC_L \leftarrow (00000000, \text{addr5}), SP \leftarrow SP - 2$	
<b>RET</b>		1	6	$PC_H \leftarrow (SP + 1), PC_L \leftarrow (SP), SP \leftarrow SP + 2$	
<b>RETI</b>		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
<b>PUSH</b>	PSW	1	2	$(SP - 1) \leftarrow PSW, SP \leftarrow SP - 1$	
	rp	1	4	$(SP - 1) \leftarrow rp_H, (SP - 2) \leftarrow rp_L, SP \leftarrow SP - 2$	
<b>POP</b>	PSW	1	4	$PSW \leftarrow (SP), SP \leftarrow SP + 1$	R R R
	rp	1	6	$rp_H \leftarrow (SP + 1), rp_L \leftarrow (SP), SP \leftarrow SP + 2$	
<b>MOVW</b>	SP, AX	2	8	$SP \leftarrow AX$	
	AX, SP	2	6	$AX \leftarrow SP$	
<b>BR</b>	!addr16	3	6	$PC \leftarrow \text{addr16}$	
	\$addr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$	
	AX	1	6	$PC_H \leftarrow A, PC_L \leftarrow X$	
<b>BC</b>	\$addr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$ if CY = 1	
<b>BNC</b>	\$addr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$ if CY = 0	
<b>BZ</b>	\$saddr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$ if Z = 1	
<b>BNZ</b>	\$saddr16	2	6	$PC \leftarrow PC + 2 + \text{jdisp8}$ if Z = 0	
<b>BT</b>	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	
<b>BF</b>	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	
<b>DBNZ</b>	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 $(\text{saddr}) \neq 0$	
<b>NOP</b>		1	2	No Operation	
<b>EI</b>		3	6	$IE \leftarrow 1$ (Enable Interrupt)	
<b>DI</b>		3	6	$IE \leftarrow 0$ (Disable Interrupt)	
<b>HALT</b>		1	2	Set HALT Mode	
<b>STOP</b>		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).

## 20.3 Instructions Listed by Addressing Type

### (1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, INC, DEC, ROR, ROL, RORC, ROLC, PUSH, POP, DBNZ

2nd Operand 1st Operand	#byte	A	r	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL+byte]	\$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV <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
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											
[HL+byte]		MOV											

**Note** Except r = A.



**(2) 16-bit instructions**

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

2nd Operand 1st Operand	#word	AX	rp <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

## APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for the development of systems that employ the  $\mu$ PD789104A/114A/124A/134A Subseries.

Figure A-1 shows the development tool configuration.

- **Support of the PC98-NX Series**

Unless otherwise specified, the  $\mu$ PD789104A/114A/124A/134A Subseries supported by IBM PC/AT™ and compatibles can be used for the PC98-NX Series. When using the PC98-NX Series, refer to the descriptions of the IBM PC/AT and compatibles.

- **Windows**

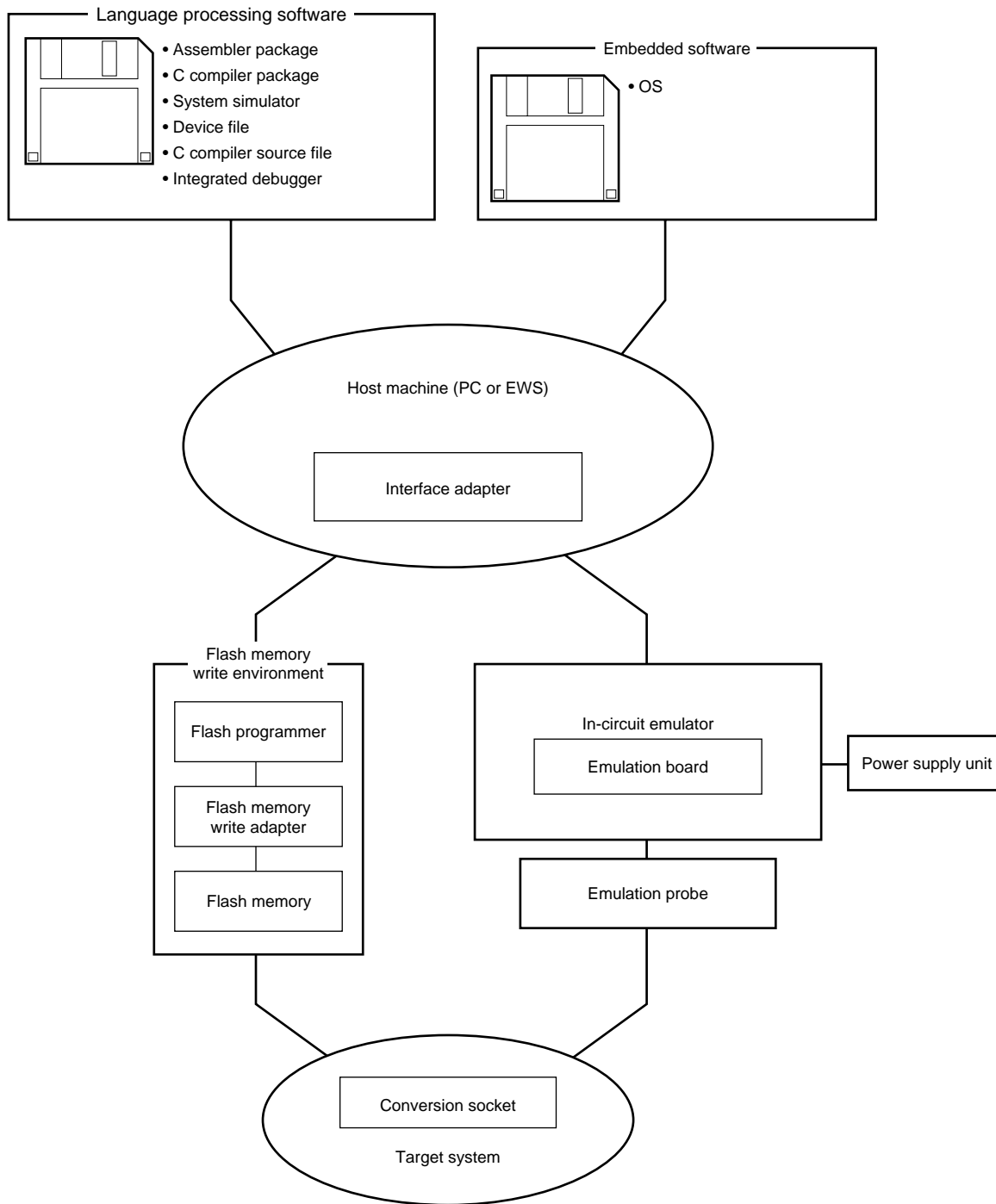
Unless otherwise specified, "Windows" indicates the following OSs.

Windows 3.1

Windows 95

Windows NT™ Ver. 4.0

Figure A-1. Development Tool Configuration



## A.1 Language Processing Software

RA78K0S Assembler package	A program that converts a program written in mnemonic into object codes that can be executed by microcontrollers. In addition, automatic functions to generate symbol tables and optimize branch instructions are also provided. Used in combination with a device file (DF789136) (sold separately). <Caution when used in PC environment> 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	A program that converts a program written in C language into object codes that can be executed by microcontrollers. Used in combination with an assembler package (RA78K0S) and device file (DF789136) (both sold separately). <Caution when used in PC environment> 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
DF789136 <sup>Note</sup> Device file	File containing the information inherent to the device. Used in combination with RA78K0S, CC78K0S, and SM78K0S (all sold separately).
	Part number: $\mu$ SxxxxDF789136
CC78K0S-L C compiler source file	Source file of functions for generating the object library included in the C compiler package. Necessary for changing the object library included in the 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

**Note** DF789136 is a common file that can be used with RA78K0S, CC78K0S, and SM78K0S.

**Remark** xxxx in the part number differs depending on the host machine and operating system to be used.

$\mu$ SxxxxRA78K0S  
 $\mu$ SxxxxCC78K0S  
 $\mu$ SxxxxDF789136  
 $\mu$ SxxxxCC78K0S-L

xxxx	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows <sup>Note</sup>	3.5" 2HD FD
AB13	IBM PC/AT and compatibles	Japanese Windows <sup>Note</sup>	3.5" 2HC FD
3P16	HP9000 Series 700™	HP-UX™ (Rel.10.10)	DAT (DDS)
3K13	SPARCstation™	SunOS™ (Rel.4.1.1)	3.5" 2HC FD
3K15		Solaris™ (Rel.2.5.1)	1/4" CGMT
3R13	NEWS™ (RISC)	NEWS-OS™ (Rel.6.1)	3.5" 2HC FD

**Note** Also operates in the DOS environment

## A.2 Flash Memory Writing Tools

Flashpro III (Part No. FL-PR3, PG-FP3) Flash programmer	Dedicated flash programmer for microcontrollers incorporating flash memory
FA-30MC Flash memory writing adapter	Adapter for writing to flash memory and connected to Flashpro III. • FA-30MC: for 30-pin plastic SSOP (MC-5A4 type)

**Remark** The FL-PR3 and FA-30MC are products made by Naito Densei Machida Mfg. Co., Ltd.  
(TEL +81-44-822-3813).

## A.3 Debugging Tools

### A.3.1 Hardware

IE-78K0S-NS In-circuit emulator	In-circuit emulator for debugging the hardware and software of an application system using the 78K/0S Series. Supports an integrated debugger (ID78K0S-NS). Used in combination with an AC adapter, emulation probe, and interface adapter for connecting the host machine.
IE-70000-MC-PS-B AC adapter	Adapter for supplying power from an AC 100 to 240 V outlet.
IE-70000-98-IF-C Interface adapter	Adapter necessary when using a PC-9800 series PC (except notebook type) as the host machine of the IE-78K0S-NS (C bus supported)
IE-70000-CD-IF-A PC card interface	PC card and interface cable necessary when using a notebook PC as the host machine of the IE-78K0S-NS (PCMCIA socket supported)
IE-70000-PC-IF-C Interface adapter	Interface adapter necessary when using an IBM PC/AT or compatible as the host machine of the IE-78K0S-NS (ISA bus supported)
IE-70000-PCI-IF Interface adapter	Adapter necessary when using a personal computer incorporating the PCI bus as the host machine of the IE-78K0S-NS
IE-789136-NS-EM1 Emulation board	Board for emulating the peripheral hardware inherent to the device. Used in combination with an in-circuit emulator.
NP-36GS Emulation probe	Probe for connecting the in-circuit emulator and target system. This is for a 30-pin plastic SSOP (MC-5A4 type).
NGS-30 Conversion socket	Conversion socket to connect the NP-36GS and a target system board on which a 30-pin plastic SSOP (MC-5A4 type) can be mounted.

**Remark** The NP-36GS, and NGS-30 are products made by Naito Densei Machida Mfg. Co., Ltd. For details of these products, contact Naito Densei Machida Mfg. Co., Ltd. (TEL +81-44-822-3813).

### A.3.2 Software

ID78K0S-NS Integrated debugger (Supports in-circuit emulator IE-78K0S-NS)	Control program for debugging the 78K/0S Series. This program provides a graphical user interface. It runs on Windows for personal computer users and on OSF/Motif™ for engineering work station users, and has visual designs and operationability that comply with these operating systems. In addition, it has a powerful debug function that supports C language. Therefore, trace results can be displayed at a C language level by the window integration function that links the source program, disassembled display, and memory display, to the trace result. This software also allows users to add other function extension modules such as a task debugger and system performance analyzer to improve the debug efficiency for programs using a real-time operating system. Used in combination with a device file (DF789136) (sold separately).
	Part number: $\mu$ SxxxxID78K0S-NS

**Remark** xxxx in the part number differs depending on the host machine and operating system to be used.

$\mu$ SxxxxID78K0S-NS

xxxx	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows <sup>Note</sup>	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows <sup>Note</sup>	3.5" 2HC FD

**Note** Also operates in the DOS environment.

SM78K0S System simulator	Debugs the program at C source level or assembler level while simulating operation of the target system on the host machine. SM78K0S runs in Windows. By using SM78K0S, the logic and performance of an application can be verified independently of hardware development even when the in-circuit emulator is not used. This enhances development efficiency and improves software quality. Used in combination with a device file (DF789136) (sold separately).
	Part number: $\mu$ SxxxxSM78K0S
DF789136 <sup>Note</sup> Device file	File containing the information inherent to the device. Used in combination with the RA78K0S, CC78K0S, and SM78K0S (all sold separately).
	Part number: $\mu$ SxxxxDF789136

**Note** DF789136 is a common file that can be used with the RA78K0S, CC78K0S, and SM78K0S.

**Remark** xxxx in the part number differs depending on the host machine and operating system to be used.

$\mu$ SxxxxSM78K0S

xxxx	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows <sup>Note</sup>	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows <sup>Note</sup>	3.5" 2HC FD

**Note** Also operates in the DOS environment.

**[MEMO]**



## APPENDIX B EMBEDDED SOFTWARE

The following embedded software products are available for efficient program development and maintenance of the  $\mu$ PD789104A/114A/124A/134A Subseries.

MX78K0S OS	MX78K0S is a subset OS that is based on the $\mu$ ITRON specification. Supplied with the MX78K0S nucleus. The MX78K0S OS controls tasks, events, and time. In task control, the MX78K0S OS controls task execution order, and then perform the switching process to a task to be executed. <Caution when used in a PC environment> The MX78K0S is a DOS-based application. Use this software in the DOS pane when running it on Windows.
	Part number: $\mu$ SxxxxMX78K0S

**Remark** xxxx in the part number differ depending on the host machine and OS used.

$\mu$ SxxxxMX78K0S

xxxx	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows <sup>Note</sup>	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows <sup>Note</sup>	3.5" 2HD FD
BB13		English Windows <sup>Note</sup>	

**Note** Can also be operated in the DOS environment.

**[MEMO]**

## APPENDIX C REGISTER INDEX

### C.1 Register Name Index (Alphabetic Order)

8-bit compare register 80 (CR80) .....	117
8-bit timer mode control register 80 (TMC80) .....	118
8-bit timer counter 80 (TM80) .....	117
16-bit compare register 20 (CR20) .....	106
16-bit multiplication result storage register 0 (MUL0) .....	199
16-bit timer capture register 20 (TCP20) .....	106
16-bit timer mode control register 20 (TMC20) .....	107
16-bit timer counter 20 (TM20) .....	106
<b>[A]</b>	
Asynchronous serial interface mode register 20 (ASIM20) .....	168, 175, 178, 190
Asynchronous serial interface status register 20 (ASIS20) .....	170, 179
A/D conversion result register 0 (ADCR0) .....	136
A/D converter mode register 0 (ADM0) .....	138
Analog input channel specification register 0 (ADS0) .....	139
<b>[B]</b>	
Baud rate generator control register 20 (BRGC20) .....	171, 180, 191
<b>[E]</b>	
External interrupt mode register 0 (INTM0) .....	209
<b>[I]</b>	
Interrupt mask flag register 0 (MK0) .....	208
Interrupt mask flag register 1 (MK1) .....	208
Interrupt request flag register 0 (IF0) .....	207
Interrupt request flag register 1 (IF1) .....	207
<b>[M]</b>	
Multiplication data register A0 (MRA0) .....	199
Multiplication data register B0 (MRB0) .....	199
Multiplier control register 0 (MULC0) .....	201
<b>[O]</b>	
Oscillation stabilization time select register (OSTS) .....	220
<b>[P]</b>	
Port 0 (P0) .....	75
Port 1 (P1) .....	76
Port 2 (P2) .....	77
Port 5 (P5) .....	81
Port 6 (P6) .....	82

Port mode register 0 (PM0) .....	83
Port mode register 1 (PM1) .....	83
Port mode register 2 (PM2) .....	83, 109, 119
Port mode register 5 (PM5) .....	83
Processor clock control register (PCC) .....	88, 96
Pull-up resistor option register 0 (PU0) .....	84
Pull-up resistor option register B2 (PUB2) .....	84
<b>[R]</b>	
Receive buffer register 20 (RXB20) .....	166
<b>[S]</b>	
Serial operating mode register 20 (CSIM20) .....	167, 174, 177, 189
<b>[T]</b>	
Timer clock select register 2 (TCL2) .....	131
Transmit shift register 20 (TXS20) .....	166
<b>[W]</b>	
Watchdog timer mode register (WDTM) .....	132

**C.2 Register Symbol Index (Alphabetic Order)****[A]**

ADCR0:	A/D conversion result register 0 .....	136
ADM0:	A/D converter mode register 0 .....	138
ADS0:	Analog input channel specification register 0 .....	139
ASIM20:	Asynchronous serial interface mode register 20 .....	168, 175, 178, 190
ASIS20:	Asynchronous serial interface status register 20 .....	170, 179

**[B]**

BRGC20:	Baud rate generator control register 20 .....	171, 180, 191
---------	---	---------------

**[C]**

CR20:	16-bit compare register 20 .....	106
CR80:	8-bit compare register 80 .....	117
CSIM20:	Serial operating mode register 20 .....	167, 174, 177, 189

**[I]**

IF0:	Interrupt request flag register 0 .....	207
IF1:	Interrupt request flag register 1 .....	207
INTM0:	External interrupt mode register 0 .....	209

**[M]**

MK0:	Interrupt mask flag register 0 .....	208
MK1:	Interrupt mask flag register 1 .....	208
MRA0:	Multiplication data register A0 .....	199
MRB0:	Multiplication data register B0 .....	199
MUL0:	16-bit multiplication result storage register 0 .....	199
MULC0:	Multiplier control register 0 .....	201

**[O]**

OSTS:	Oscillation stabilization time select register .....	220
-------	--	-----

**[P]**

P0:	Port 0 .....	75
P1:	Port 1 .....	76
P2:	Port 2 .....	77
P5:	Port 5 .....	81
P6:	Port 6 .....	82
PCC:	Processor clock control register .....	88, 96
PM0:	Port mode register 0 .....	83
PM1:	Port mode register 1 .....	83
PM2:	Port mode register 2 .....	83, 109, 119
PM5:	Port mode register 5 .....	83
PU0:	Pull-up resistor option register 0 .....	84
PUB2:	Pull-up resistor option register B2 .....	84

**[R]**

RXB20:	Receive buffer register 20 .....	166
--------	----------------------------------	-----

**[T]**

TCL2:	Timer clock select register 2 .....	131
TCP20:	16-bit timer capture register 20 .....	106
TM20:	16-bit timer counter 20 .....	106
TM80:	8-bit timer counter 80 .....	117
TMC20:	16-bit timer mode control register 20 .....	116
TMC80:	8-bit timer mode control register 80 .....	118
TXS20:	Transmit shift register 20 .....	166

**[W]**

WDTM:	Watchdog timer mode register .....	132
-------	------------------------------------	-----

## Facsimile Message

From:

Name

Company

Tel.

FAX

Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

*Thank you for your kind support.*

### North America

NEC Electronics Inc.  
Corporate Communications Dept.  
Fax: 1-800-729-9288  
1-408-588-6130

### Europe

NEC Electronics (Europe) GmbH  
Technical Documentation Dept.  
Fax: +49-211-6503-274

### South America

NEC do Brasil S.A.  
Fax: +55-11-6465-6829

### Hong Kong, Philippines, Oceania

NEC Electronics Hong Kong Ltd.  
Fax: +852-2886-9022/9044

### Korea

NEC Electronics Hong Kong Ltd.  
Seoul Branch  
Fax: 02-528-4411

### Taiwan

NEC Electronics Taiwan Ltd.  
Fax: 02-2719-5951

### Asian Nations except Philippines

NEC Electronics Singapore Pte. Ltd.  
Fax: +65-250-3583

### Japan

NEC Semiconductor Technical Hotline  
Fax: 044-548-7900

I would like to report the following error/make the following suggestion:

Document title: \_\_\_\_\_

Document number: \_\_\_\_\_ Page number: \_\_\_\_\_

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Technical Accuracy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>