

# User's Manual

## **V854™**

### **32/16-Bit Single-Chip Microcontroller**

#### **Hardware**

---

**μPD703006**

**μPD703008**

**μPD70F3008**

**μPD703008Y**

**μPD70F3008Y**

**[MEMO]**

## NOTES FOR CMOS DEVICES

### ① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

### ② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

### ③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

**V854, and V850 Family are trademarks of NEC Corporation.**

**Windows is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.**

**UNIX is a registered trademark licensed by X/Open Company Limited in the United States and other countries.**

Purchase of NEC I<sup>2</sup>C components conveys a license under the Philips I<sup>2</sup>C Patent Rights to use these components in an I<sup>2</sup>C system, provided that the system conforms to the I<sup>2</sup>C Standard Specification as defined by Philips.

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.

License not needed :  $\mu$ PD703006, 70F3008, 70F3008Y  
The customer must judge the need for license:  $\mu$ PD703008, 703008Y

**The information in this document is subject to change without notice.**

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.

NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.

While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.

NEC devices are classified into the following three quality grades:

"Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device before using it in a particular application.

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

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

Specific: Aircrafts, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.

The quality grade of NEC devices is "Standard" unless otherwise specified in NEC's Data Sheets or Data Books. If customers intend to use NEC devices for applications other than those specified for Standard quality grade, they should contact an NEC sales representative in advance.

Anti-radioactive design is not implemented in this product.

# Regional Information

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

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

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

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

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

## **NEC Electronics (Germany) GmbH**

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

## **NEC Electronics (UK) Ltd.**

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

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

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

## **NEC Electronics (Germany) GmbH**

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

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

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

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

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

## **NEC Electronics (Germany) GmbH**

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

## **NEC Electronics Hong Kong Ltd.**

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

## **NEC Electronics Hong Kong Ltd.**

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

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

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

## **NEC Electronics Taiwan Ltd.**

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

## **NEC do Brasil S.A.**

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

J99.1

## Major Revisions in This Edition

Page	Contents
Throughout	Deletion of $BV_{DD}$ and $BV_{SS}$ Modification of voltage of $V_{PP}$ Addition of $\mu$ PD703006 to the target devices
p.44	Modification of description in <b>2.3 (19) MODE0 to MODE2 (Mode 0 to 2)</b>
p.47	Modification of recommended connection method for pins P40/AD0 to P47/AD7, P50/AD8 to P57/AD15, P60/A16 to P67/A23, P90/ $\overline{LBEN}$ / $\overline{WRL}$ , P91/ $\overline{UBEN}$ , P92/ $\overline{R/W}$ / $\overline{WRH}$ , P93/ $\overline{DSTB/RD}$ , P94/ASTB, P95/HLDAK, P96/HLDRQ, $\overline{WAIT}$ , and MODE0 to MODE2 in <b>2.4 Pin I/O Circuit Type and Connection of Unused Pins</b>
p.53	Modification of description of EP flag in <b>Figure 3-3 Program Status Word (PSW)</b>
p.54	Modification of description in <b>3.3.2 Specifying operation mode</b>
p.140	Addition of <b>Caution</b> in <b>6.5.1 (2) IDLE mode</b>
p.140	Modification of description in <b>6.5.1 (3) (a) PLL mode</b>
p.140	Modification of description in <b>6.5.1 (3) (b) Direct mode</b>
p.142	Modification of description of CESEL bit in <b>6.5.2 (1) Power save control register (PSC)</b>
p.149	Modification of description in <b>6.6 (1) Securing time using internal time base counter (NMI pin input)</b>
p.150	Modification of description in <b>6.6 (2) Securing time by signal level width (RESET pin input)</b>
p.159	Addition of <b>Note</b> in <b>7.2 (1) Timer 0 (24-bit timer/event counter)</b>
p.161	Addition of description in <b>7.2.1 (1) Timers 0, 0L (TM0, TM0L)</b>
p.206	Addition to the item Transfer rate in <b>8.2.1 Features</b>
p.220	Addition to the item High transfer speed in <b>8.3.1 Features</b>
p.241	Addition of <b>Note</b> to bits 4 and 5 in <b>8.4.4 (3) IIC clock selection register (IICCL)</b>
p.242	Addition of <b>Caution</b> to <b>8.4.4 (4) IIC shift register (IIC)</b>
p.276	Addition of <b>8.4.6 (15) (b) Operation during communication reservation (when a multi-master is used), (c) Start operation after communication reservation (when a multi-master is used), and (d) STT setting timing (when a multi-master is used)</b>
p.299	Modification of description of function of bits FR2 to FR0 in the table in <b>9.3 (2) A/D converter mode register 1 (ADM1)</b>
p.301	Modification of setting value of ADM1 register in the table of operation and trigger modes in <b>9.4.2 Operation mode and trigger mode</b>
pp.308 to 310, 312 to 318	Addition of Tables 9-1 to 9-9 and Figures 9-6 to 9-14
p.319	Modification of description in <b>9.8.2 Interval of the external/timer trigger</b>
p.379	Addition of description in <b>13.2 (2) Power-ON reset</b>
p.380	Modification of initial values after reset of timer registers (TM0, TM1, TM0L, TM1L, TM20 to TM24, TM3) in <b>Table 13-2 Initial Values after Reset of Each Register (1/2)</b>
p.384	Deletion of CS12 in <b>14.3 Programming Environment</b>
p.385	Deletion of CS12, SO2, SI2, and $\overline{SCK2}$ in <b>14.4 Communication System</b>
p.388	Deletion of CS12 and the pin used by CS12 in the table of pins used by each serial interface in <b>14.5.2 Serial interface pin</b>
p.392	Deletion of CS12 in <b>Table 14-1 List of Communication Systems</b>
p.393	Modification of the table of flash memory control commands in <b>14.6.4 Communication command</b>

The mark ★ shows major revised points in this edition.

## INTRODUCTION

**Readers** This manual is intended for users who understand the functions of the V854 ( $\mu$ PD703006, 703008, 70F3008, 703008Y, 70F3008Y) and design application systems using the V854.

**Purpose** This manual is intended to enable users to understand the hardware functions described in the Organization below.

**Organization** The V854 User's Manual is divided into two parts: hardware (this manual) and architecture (V850 Family™ Architecture User's Manual) manuals.

Hardware	Architecture
<ul style="list-style-type: none"><li>• Pin function</li><li>• CPU function</li><li>• Internal peripheral function</li><li>• Flash memory programming mode</li></ul>	<ul style="list-style-type: none"><li>• Data type</li><li>• Register set</li><li>• Instruction format and instruction set</li><li>• Interrupt and exception</li><li>• Pipeline operation</li></ul>

**How to Read This Manual** It is assumed that the readers of this manual have general knowledge of electrical engineering, logic circuits, and microcontrollers.

- To find out the details of a register whose the name is known:  
→ Refer to **APPENDIX A REGISTER INDEX**.
- To find out the details of a function whose name is known:  
→ Refer to **APPENDIX C INDEX**.
- To understand the details of an instruction function:  
→ Refer to the **V850 Family Architecture User's Manual** available separately.
- To understand the overall functions of the V854:  
→ Read this manual according to the Table of Contents.

<b>Conventions</b>	Data significance:	Higher digits on the left and lower digits on the right
	Active low:	$\overline{\text{xxx}}$ (overscore over pin or signal name)
	Memory map address:	High order at high stage and low order at low stage
	<b>Note:</b>	Footnote for items marked with <b>Note</b> in the text
	<b>Caution:</b>	Information requiring particular attention
	<b>Remark:</b>	Supplementary information
	Number representation:	Binary ... xxxx or xxxxB Decimal ... xxxx Hexadecimal ... xxxxH
	Prefixes indicating power of 2 (address space, memory capacity):	K (kilo): $2^{10} = 1024$ M (mega): $2^{20} = 1024^2$ G (giga): $2^{30} = 1024^3$

**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.
V850 Family Architecture User's Manual	U10243E
V850 Family Instruction Table	U10229E
μPD703008 Data Sheet	To be prepared
μPD703008Y Data Sheet	To be prepared
μPD70F3008 Data Sheet	U12756E
μPD70F3008Y Data Sheet	U12755E
V854 Hardware User's Manual	This manual

- **Documents related to development tools (User's Manual)**

Document Name	Document No.	
IE-703002-MC (In-circuit emulator)	U11595E	
IE-703008-MC-EM1 (In-circuit emulator option board)	U12420E	
CA850 (C Compiler Package)	Operation (UNIX™ based)	U12839E
	Operation (Windows™ based)	U12827E
	C Language	U12840E
	Assembly Language	U10543E
ID850 (Ver.1.31) (Integrated Debugger)	Operation Windows based	U13716E
RX850 (Real Time OS)	Basics	U13430E
	Technical	U13431E
	Installation	U13410E
RX850 Pro (Real Time OS)	Fundamental	U13773E
	Technical	U13772E
	Installation	U13774E
RD850 (Task Debugger) <sup>Note</sup>		U11158E
RD850 (Ver.3.0) (Task Debugger)		U13737E
AZ850 (System Performance Analyzer)		U11181E
SM850 (Ver.2.0) (System Simulator) Operation Windows based.		Under preparation

**Note** Supports ID850 (Ver.1.31)

## TABLE OF CONTENTS

<b>CHAPTER 1 INTRODUCTION .....</b>	<b>23</b>
<b>1.1 General .....</b>	<b>23</b>
<b>1.2 Features .....</b>	<b>24</b>
<b>1.3 Application Fields .....</b>	<b>25</b>
<b>1.4 Ordering Information .....</b>	<b>25</b>
<b>1.5 Pin Identification (Top View) .....</b>	<b>26</b>
<b>1.6 Function Block Configuration .....</b>	<b>28</b>
1.6.1 Internal block diagram .....	28
1.6.2 Internal units .....	29
<b>CHAPTER 2 PIN FUNCTIONS .....</b>	<b>31</b>
<b>2.1 Pin Function List .....</b>	<b>31</b>
<b>2.2 Pin Status .....</b>	<b>36</b>
<b>2.3 Pin Function .....</b>	<b>37</b>
<b>2.4 Pin I/O Circuit Type and Connection of Unused Pins .....</b>	<b>47</b>
<b>2.5 I/O Circuits of Pins .....</b>	<b>48</b>
<b>CHAPTER 3 CPU FUNCTIONS .....</b>	<b>49</b>
<b>3.1 Features .....</b>	<b>49</b>
<b>3.2 CPU Register Set .....</b>	<b>50</b>
3.2.1 Program register set .....	51
3.2.2 System register set .....	52
<b>3.3 Operation Modes .....</b>	<b>54</b>
3.3.1 Operation modes .....	54
3.3.2 Specifying operation mode .....	54
<b>3.4 Address Space .....</b>	<b>56</b>
3.4.1 CPU address space .....	56
3.4.2 Image (Virtual Address Space) .....	57
3.4.3 Wrap-around of CPU address space .....	58
3.4.4 Memory map .....	59
3.4.5 Area .....	60
3.4.6 External expansion mode .....	67
3.4.7 Recommended use of address space .....	69
3.4.8 Peripheral I/O registers .....	71
3.4.9 Specific registers .....	77
<b>CHAPTER 4 BUS CONTROL FUNCTION .....</b>	<b>81</b>
<b>4.1 Features .....</b>	<b>81</b>
<b>4.2 Bus Control Pins and Control Register .....</b>	<b>82</b>
4.2.1 Bus control pins .....	82
4.2.2 Control register .....	82
<b>4.3 Bus Access .....</b>	<b>83</b>
4.3.1 Number of access clocks .....	83

4.3.2	Bus width .....	84
<b>4.4</b>	<b>Memory Block Function .....</b>	<b>85</b>
<b>4.5</b>	<b>Wait Function .....</b>	<b>86</b>
4.5.1	Programmable wait function .....	86
4.5.2	External wait function .....	87
4.5.3	Relations between programmable wait and external wait .....	87
<b>4.6</b>	<b>Idle State Insertion Function .....</b>	<b>88</b>
<b>4.7</b>	<b>Bus Hold Function .....</b>	<b>89</b>
4.7.1	Outline of function .....	89
4.7.2	Bus hold procedure .....	89
4.7.3	Operation in power save mode .....	89
<b>4.8</b>	<b>Bus Timing .....</b>	<b>90</b>
<b>4.9</b>	<b>Bus Priority .....</b>	<b>97</b>
<b>4.10</b>	<b>Memory Boundary Operation Condition .....</b>	<b>97</b>
4.10.1	Program space .....	97
4.10.2	Data space .....	97
<b>4.11</b>	<b>Internal Peripheral I/O Interface .....</b>	<b>98</b>
<b>CHAPTER 5</b>	<b>INTERRUPT/EXCEPTION PROCESSING FUNCTION .....</b>	<b>99</b>
<b>5.1</b>	<b>Features .....</b>	<b>99</b>
<b>5.2</b>	<b>Non-Maskable Interrupt .....</b>	<b>102</b>
5.2.1	Operation .....	103
5.2.2	Restore .....	105
5.2.3	Non-maskable interrupt status flag (NP) .....	106
5.2.4	Noise elimination circuit of NMI pin .....	106
5.2.5	Edge detection function of NMI pin .....	106
<b>5.3</b>	<b>Maskable Interrupts .....</b>	<b>107</b>
5.3.1	Operation .....	109
5.3.2	Restore .....	111
5.3.3	Priorities of maskable interrupts .....	112
5.3.4	Interrupt control register (xxICn) .....	116
5.3.5	In-service priority register (ISPR) .....	118
5.3.6	Maskable interrupt status flag (ID) .....	118
5.3.7	Noise elimination .....	119
5.3.8	Edge detection function .....	120
5.3.9	Frequency divider .....	124
<b>5.4</b>	<b>Software Exception .....</b>	<b>126</b>
5.4.1	Operation .....	126
5.4.2	Restore .....	127
5.4.3	Exception status flag (EP) .....	128
<b>5.5</b>	<b>Exception Trap .....</b>	<b>129</b>
5.5.1	Illegal op code definition .....	129
5.5.2	Operation .....	129
5.5.3	Restore .....	130
<b>5.6</b>	<b>Multiple interrupt processing .....</b>	<b>131</b>
<b>5.7</b>	<b>Interrupt Response Time .....</b>	<b>133</b>

<b>5.8</b>	<b>Periods Where Interrupt is Not Acknowledged .....</b>	<b>133</b>
<b>CHAPTER 6</b>	<b>CLOCK GENERATOR FUNCTION .....</b>	<b>135</b>
<b>6.1</b>	<b>Features .....</b>	<b>135</b>
<b>6.2</b>	<b>Configuration .....</b>	<b>135</b>
<b>6.3</b>	<b>Selecting Input Clock .....</b>	<b>136</b>
6.3.1	Direct mode .....	136
6.3.2	PLL mode .....	136
6.3.3	Clock control register (CKC) .....	137
<b>6.4</b>	<b>PLL Lock-up .....</b>	<b>139</b>
<b>6.5</b>	<b>Power Save Control .....</b>	<b>140</b>
6.5.1	General .....	140
6.5.2	Control registers .....	142
6.5.3	HALT mode .....	143
6.5.4	IDLE mode .....	145
6.5.5	Software STOP mode .....	147
<b>6.6</b>	<b>Specifying Oscillation Stabilization Time .....</b>	<b>149</b>
<b>6.7</b>	<b>Clock Output Control .....</b>	<b>152</b>
6.7.1	Configuration .....	152
6.7.2	CLKOUT signal output control .....	152
6.7.3	CLO signal output control .....	153
<b>CHAPTER 7</b>	<b>TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT) .....</b>	<b>157</b>
<b>7.1</b>	<b>Features .....</b>	<b>157</b>
<b>7.2</b>	<b>Basic Configuration .....</b>	<b>158</b>
7.2.1	Timer 0 .....	161
7.2.2	Timer 1 .....	162
7.2.3	Timer 2 .....	164
7.2.4	Timer 3 .....	165
<b>7.3</b>	<b>Control Register .....</b>	<b>166</b>
<b>7.4</b>	<b>Timer 0 Operation .....</b>	<b>174</b>
7.4.1	Count operation .....	174
7.4.2	Count clock selection .....	175
7.4.3	Overflow .....	176
7.4.4	Clearing/starting timer .....	177
7.4.5	Capture operation .....	179
7.4.6	Compare operation .....	181
<b>7.5</b>	<b>Timer 1 Operation .....</b>	<b>183</b>
7.5.1	Count operation .....	183
7.5.2	Count clock selection .....	183
7.5.3	Overflow .....	184
7.5.4	Clearing/starting timer .....	185
7.5.5	Capture operation .....	186
7.5.6	Compare operation .....	187
<b>7.6</b>	<b>Timer 2 Operation .....</b>	<b>188</b>
7.6.1	Count operation .....	188

7.6.2	Count clock selection .....	188
7.6.3	Overflow .....	189
7.6.4	Clearing/starting timer .....	189
7.6.5	Compare operation .....	189
7.6.6	Toggle output .....	191
<b>7.7</b>	<b>Timer 3 Operation .....</b>	<b>192</b>
7.7.1	Count operation .....	192
7.7.2	Count clock selection .....	192
7.7.3	Overflow .....	192
7.7.4	Clearing/starting timer .....	193
7.7.5	Capture operation .....	193
7.7.6	Compare operation .....	194
<b>7.8</b>	<b>Application Examples .....</b>	<b>195</b>
<b>7.9</b>	<b>Note .....</b>	<b>203</b>
<b>CHAPTER 8 SERIAL INTERFACE FUNCTION .....</b>		<b>205</b>
<b>8.1</b>	<b>Features .....</b>	<b>205</b>
<b>8.2</b>	<b>Asynchronous Serial Interface (UART) .....</b>	<b>206</b>
8.2.1	Features .....	206
8.2.2	Configuration of asynchronous serial interface .....	207
8.2.3	Control registers .....	209
8.2.4	Interrupt request .....	215
8.2.5	Operation .....	216
<b>8.3</b>	<b>Clocked Serial Interface 0 to 3 (CSI0 to CSI3) .....</b>	<b>220</b>
8.3.1	Features .....	220
8.3.2	Configuration .....	220
8.3.3	Control registers .....	222
8.3.4	Basic operation .....	224
8.3.5	Transmission in CSI0 to CSI3 .....	226
8.3.6	Reception in CSI0 to CSI3 .....	227
8.3.7	Transmission/reception in CSI0 to CSI3 .....	228
8.3.8	System configuration example .....	230
<b>8.4</b>	<b>I<sup>2</sup>C Bus (<math>\mu</math>PD703008Y and 70F3008Y only) .....</b>	<b>231</b>
8.4.1	Features .....	231
8.4.2	Functions .....	232
8.4.3	Configuration .....	234
8.4.4	Serial interface control register .....	236
8.4.5	I <sup>2</sup> C bus functions .....	242
8.4.6	Definition and controls of I <sup>2</sup> C bus .....	243
8.4.7	Communication operation .....	277
8.4.8	Timing chart .....	279
<b>8.5</b>	<b>Baud Rate Generator 0 to 3 (BRG0 to BRG3) .....</b>	<b>286</b>
8.5.1	Configuration and function .....	286
8.5.2	Baud rate generator compare registers 0 to 3 (BRGC0 to BRGC3) .....	291
8.5.3	Baud rate generator prescaler mode registers 0 to 3 (BPRM0 to BPRM3) .....	292
<b>8.6</b>	<b>Selection of Operational Serial Interface .....</b>	<b>293</b>

<b>CHAPTER 9 A/D CONVERTER .....</b>	<b>295</b>
<b>9.1 Features .....</b>	<b>295</b>
<b>9.2 Configuration .....</b>	<b>295</b>
<b>9.3 Control Register .....</b>	<b>297</b>
<b>9.4 A/D Converter Operation .....</b>	<b>301</b>
9.4.1 Basic operation of A/D converter .....	301
9.4.2 Operation mode and trigger mode .....	301
<b>9.5 Operation in the A/D Trigger Mode .....</b>	<b>308</b>
9.5.1 Select mode operation .....	308
9.5.2 Scan mode operation .....	310
<b>9.6 Operation in the Timer Trigger Mode .....</b>	<b>311</b>
9.6.1 Select mode operation .....	311
9.6.2 Scan mode operation .....	314
<b>9.7 Operation in the External Trigger Mode .....</b>	<b>315</b>
9.7.1 Select mode operation (External trigger select) .....	315
9.7.2 Scan mode operation (External trigger scan) .....	318
<b>9.8 Precautions Regarding Operations .....</b>	<b>319</b>
9.8.1 Stop of conversion operations .....	319
9.8.2 Interval of the external/timer trigger .....	319
9.8.3 Operation in the standby mode .....	319
9.8.4 Compare coincide interrupt in the timer trigger mode .....	319
<b>CHAPTER 10 REAL-TIME OUTPUT FUNCTION .....</b>	<b>321</b>
<b>10.1 Configuration and Function .....</b>	<b>321</b>
<b>10.2 Control Register .....</b>	<b>322</b>
<b>10.3 Operation .....</b>	<b>322</b>
<b>10.4 Example .....</b>	<b>323</b>
<b>CHAPTER 11 PWM UNIT .....</b>	<b>325</b>
<b>11.1 Features .....</b>	<b>325</b>
<b>11.2 Configuration .....</b>	<b>326</b>
<b>11.3 Control Register .....</b>	<b>327</b>
<b>11.4 PWM Operations .....</b>	<b>330</b>
11.4.1 Basic operations of PWM .....	330
11.4.2 Enabling/disabling PWM operation .....	332
11.4.3 Specification of active level of PWM pulse .....	333
11.4.4 Specification of PWM pulse width rewrite cycle .....	333
11.4.5 Repetition frequency .....	335
<b>CHAPTER 12 PORT FUNCTION .....</b>	<b>337</b>
<b>12.1 Features .....</b>	<b>337</b>
<b>12.2 Basic Configuration of Ports .....</b>	<b>338</b>
<b>12.3 Port Pin Function .....</b>	<b>348</b>
12.3.1 Port 0 .....	348
12.3.2 Port 1 .....	350
12.3.3 Port 2 .....	352

12.3.4	Port 3 .....	354
12.3.5	Port 4 .....	357
12.3.6	Port 5 .....	359
12.3.7	Port 6 .....	361
12.3.8	Port 7, port 8 .....	363
12.3.9	Port 9 .....	364
12.3.10	Port 10 .....	366
12.3.11	Port 11 .....	368
12.3.12	Port 12 .....	371
12.3.13	Port 13 .....	373
12.3.14	Port 14 .....	375
<b>CHAPTER 13</b>	<b>RESET FUNCTION .....</b>	<b>377</b>
13.1	<b>Features .....</b>	<b>377</b>
13.2	<b>Pin Function .....</b>	<b>377</b>
13.3	<b>Initialize .....</b>	<b>379</b>
<b>CHAPTER 14</b>	<b>FLASH MEMORY (<math>\mu</math>PD70F3008 AND 70F3008Y ONLY) .....</b>	<b>383</b>
14.1	<b>Features .....</b>	<b>383</b>
14.2	<b>Writing by Flash Writer .....</b>	<b>383</b>
14.3	<b>Programming Environment .....</b>	<b>384</b>
14.4	<b>Communication System .....</b>	<b>385</b>
14.5	<b>Pin Handling .....</b>	<b>386</b>
14.5.1	V <sub>PP</sub> pin .....	387
14.5.2	Serial interface pin .....	388
14.5.3	Reset pin .....	390
14.5.4	NMI pin .....	390
14.5.5	Mode pin .....	390
14.5.6	Port pin .....	390
14.5.7	Other signal pin .....	390
14.5.8	Power supply .....	390
14.6	<b>Programming Method .....</b>	<b>391</b>
14.6.1	Flash memory control .....	391
14.6.2	Flash memory programming mode .....	392
14.6.3	Selection of communication mode .....	392
14.6.4	Communication command .....	393
14.6.5	Resources used .....	393
<b>CHAPTER 15</b>	<b>DIFFERENCES BETWEEN VERSIONS .....</b>	<b>395</b>
15.1	<b>Differences between Versions with I<sup>2</sup>C Function and Versions without I<sup>2</sup>C Function .....</b>	<b>395</b>
15.2	<b>Differences between On-chip Flash Memory Versions, On-chip Mask ROM Versions, and ROM-less Versions .....</b>	<b>395</b>

<b>APPENDIX A REGISTER INDEX .....</b>	<b>397</b>
<b>APPENDIX B INSTRUCTION SET LIST .....</b>	<b>403</b>
<b>APPENDIX C INDEX .....</b>	<b>409</b>

## LIST OF FIGURES (1/4)

Figure No.	Title	Page
3-1.	Program Counter (PC) .....	51
3-2.	Interrupt Source Register (ECR) .....	52
3-3.	Program Status Word (PSW) .....	53
3-4.	CPU Address Space .....	56
3-5.	Image on Address Space .....	57
3-6.	External Memory Area (when expanded to 64 K, 256 K, or 1 Mbytes) .....	64
3-7.	External Memory Area (when expanded to 4 Mbytes) .....	65
3-8.	External Memory Area (when fully expanded) .....	66
3-9.	Recommended Memory Map .....	70
4-1.	Example of Inserting Wait States .....	87
5-1.	Non-Maskable Interrupt Processing .....	103
5-2.	Accepting Non-Maskable Interrupt Request .....	104
5-3.	RETI Instruction Processing .....	105
5-4.	Block Diagram of Maskable Interrupt .....	108
5-5.	Maskable Interrupt Processing .....	110
5-6.	RETI Instruction Processing .....	111
5-7.	Example of Interrupt Nesting Process .....	113
5-8.	Example of Processing Interrupt Requests Simultaneously Generated .....	115
5-9.	Example of Noise Elimination Timing .....	119
5-10.	Software Exception Processing .....	126
5-11.	RETI Instruction Processing .....	127
5-12.	Exception Trap Processing .....	129
5-13.	RETI Instruction Processing .....	130
5-14.	Pipeline Operation at Interrupt Request Acknowledge (General Description) .....	133
6-1.	Block Configuration .....	151
6-2.	CLO Signal Output Timing .....	153
7-1.	Basic Operation of Timer 0 .....	174
7-2.	Operation after Occurrence of Overflow (when ECLR0 = 0, OST0 = 1) .....	176
7-3.	Clearing/Starting Timer by TCLR0 Signal Input (when ECLR0 = 1, CCLR0 = 0, OST0 = 0) .....	177
7-4.	Relations between Clear/Start by TCLR0 Signal Input and Overflow (when ECLR0 = 1, OST0 = 1) .....	178
7-5.	Clearing/Starting Timer by CC03 Coincidence (when CCLR0 =1, OST0 = 0) .....	178
7-6.	Relations between Clear/Start by CC03 Coincidence and Overflow Operation (when CCLR0 = 1, OST0 = 1) .....	179
7-7.	Example of TM0 Capture Operation .....	180
7-8.	Example of TM0 Capture Operation (when both edges are specified) .....	180
7-9.	Example of Compare Operation .....	181

## LIST OF FIGURES (2/4)

Figure No.	Title	Page
7-10.	Example of TM0 Compare Operation (set/reset output mode).....	182
7-11.	Basic Operation of Timer 1 .....	183
7-12.	Operation after Occurrence of Overflow (OST1 = 1) .....	185
7-13.	Clearing/Starting Timer by Software (when OST1 = 1) .....	185
7-14.	Example of TM1 Capture Operation .....	186
7-15.	Example of Compare Operation .....	187
7-16.	Basic Operation of Timer 2 .....	188
7-17.	Operation with CM2n at 1 to FFFFH .....	190
7-18.	When CM2n is Set to 0 .....	190
7-19.	Example of Toggle Output Operation .....	191
7-20.	Basic Operation of Timer 3 .....	192
7-21.	Example of TM3 Capture Operation (when ES301 = 0, ES300 = 0, CMS3 = 0, CE3 = 1) .....	194
7-22.	Example of Timing of Interval Timer Operation (timer 2) .....	195
7-23.	Setting Procedure of Interval Timer Operation (timer 2) .....	195
7-24.	Pulse Width Measurement Timing (timer 0) .....	196
7-25.	Setting Procedure for Pulse Width Measurement (timer 0) .....	197
7-26.	Interrupt Request Processing Routine Calculating Pulse Width (timer 0) .....	197
7-27.	Example of PWM Output Timing .....	198
7-28.	Example of PWM Output Programming Procedure .....	199
7-29.	Example of Interrupt Request Processing Routine, Modifying Compare Value .....	200
7-30.	Example of Frequency Measurement Timing .....	201
7-31.	Example of Set-up Procedure for Frequency Measurement .....	202
7-32.	Example of Interrupt Request Processing Routine Calculating Cycle .....	202
8-1.	Block Diagram of Asynchronous Serial Interface .....	208
8-2.	Format of Transmit/Receive Data of Asynchronous Serial Interface .....	216
8-3.	Asynchronous Serial Interface Transmission Completion Interrupt Timing .....	217
8-4.	Asynchronous Serial Interface Reception Completion Interrupt Timing .....	219
8-5.	Receive Error Timing .....	219
8-6.	Block Diagram of Clocked Serial Interface .....	221
8-7.	Timing of 3-Wire Serial I/O Mode (transmission) .....	226
8-8.	Timing of 3-Wire Serial I/O Mode (reception) .....	227
8-9.	Timing of 3-Wire Serial I/O Mode (transmission/reception) .....	229
8-10.	Example of CSI System Configuration .....	230
8-11.	Example of Serial Bus Configuration Using I <sup>2</sup> C Bus .....	232
8-12.	Block Diagram of I <sup>2</sup> C Bus .....	233
8-13.	Pin Configuration .....	242
8-14.	Serial Data Transfer Timing of I <sup>2</sup> C Bus .....	243
8-15.	Start Condition .....	243
8-16.	Address .....	244
8-17.	Transfer Direction Specification .....	245

## LIST OF FIGURES (3/4)

Figure No.	Title	Page
8-18.	Acknowledge Signal .....	246
8-19.	Stop Condition .....	247
8-20.	Wait Signal .....	248
8-21.	Example of Arbitration Timing .....	272
8-22.	Communication Reservation Timing .....	274
8-23.	Communication Reservation Procedure.....	275
8-24.	STT Setting Timing Procedure .....	276
8-25.	Master Operation Procedure .....	277
8-26.	Slave Operation Procedure .....	278
8-27.	Example of Master → Slave Communication (9-clock wait is selected both for master and slave) .....	280
8-28.	Example of Slave → Master Communication (9-clock wait is selected both for master and slave) .....	283
8-29.	Block Diagram of Baud Rate Generator .....	287
9-1.	Block Diagram of A/D Converter .....	296
9-2.	Relation between Analog Input Voltage and A/D Conversion Result .....	300
9-3.	Operation Timing Example of Select Mode: 1-Buffer Mode (ANI1) .....	303
9-4.	Operation Timing Example of Select Mode: 4-Buffer Mode (ANI6) .....	304
9-5.	Operation Timing Example of Scan Mode: 4-Channel Scan (ANI0 to ANI3) .....	306
9-6.	Example of 1-Buffer Mode (A/D trigger select 1-buffer) Operation .....	308
9-7.	Example of 4-Buffer Mode (A/D trigger select 4-buffer) Operation .....	309
9-8.	Example of Scan Mode (A/D trigger scan) Operation .....	310
9-9.	Example of 1-Buffer Mode (timer trigger select 1-buffer) Operation .....	312
9-10.	Example of Operation in 4-Buffer Mode (timer trigger select 4-buffer) .....	313
9-11.	Example of Scan Mode (timer trigger scan) Operation.....	314
9-12.	Example of 1-Buffer Mode (external trigger select 1-buffer) Operation .....	316
9-13.	Example of 4-Buffer Mode (external trigger select 4-buffer) Operation .....	317
9-14.	Example of Scan Mode (external trigger scan) Operation.....	318
10-1.	Block Diagram of Real-Time Output Port .....	321
10-2.	Operational Timings of Real-Time Output Port .....	323
11-1.	Configuration of PWM Unit .....	326
11-2.	Basic Operations of PWM.....	330
11-3.	Example of PWM Output by Main Pulse and Additional Pulse .....	331
11-4.	Example of PWM Output Operation .....	331
11-5.	Operation Timing of PWM .....	332
11-6.	Setting of Active Level of PWM Output .....	333
11-7.	Example 1 of PWM Output Timing (PWM pulse width rewrite cycle $2^{(x+8)}/f_{PWMC}$ ) .....	334
11-8.	Example 2 of PWM Output Timing (PWM pulse width rewrite cycle $2^x/f_{PWMC}$ ) .....	334

## LIST OF FIGURES (4/4)

Figure No.	Title	Page
12-1.	Block Diagram of Type A .....	342
12-2.	Block Diagram of Type B .....	342
12-3.	Block Diagram of Type C .....	343
12-4.	Block Diagram of Type D .....	343
12-5.	Block Diagram of Type E .....	344
12-6.	Block Diagram of Type F .....	344
12-7.	Block Diagram of Type G .....	345
12-8.	Block Diagram of Type H.....	345
12-9.	Block Diagram of Type I .....	346
12-10.	Block Diagram of Type J .....	346
12-11.	Block Diagram of Type K.....	347

## LIST OF TABLES (1/2)

Table No.	Title	Page
3-1.	Program Registers .....	51
3-2.	System Register Numbers .....	52
3-3.	Interrupt/Exception Table .....	61
4-1.	Bus Priority .....	97
5-1.	Interrupt List .....	100
6-1.	Operation of Clock Generator by Power Save Control .....	141
6-2.	Operating Status in HALT Mode .....	143
6-3.	Operating Status in IDLE Mode .....	145
6-4.	Operating Status in Software STOP Mode .....	147
6-5.	Example of Count Time .....	151
7-1.	List of Real-Time Pulse Unit (RPU) Configuration .....	158
7-2.	Capture Trigger Signal to 24-Bit Capture Register .....	179
7-3.	Interrupt Request Signal from 24-Bit Compare Register .....	181
7-4.	Capture Trigger Signal to 24-Bit Capture Register .....	186
7-5.	Interrupt Request Signal from 24-Bit Compare Register .....	187
7-6.	Capture Trigger Signal to 16-Bit Capture Register .....	193
8-1.	Default Priority of Interrupts .....	215
8-2.	I <sup>2</sup> C Bus Configuration .....	234
8-3.	INTIIC Generation Timing and Wait Control .....	270
8-4.	Definition of Extension Code Bit .....	271
8-5.	Wait Time .....	273
8-6.	Baud Rate Generators 0 to 3 Set-up Values (when typical clocks are used) .....	289
9-1.	Correspondence between Analog Input Pin and ADCRn Register (1-buffer mode (A/D trigger select 1-buffer)) .....	308
9-2.	Correspondence between Analog Input Pin and ADCRn Register (4-buffer mode (A/D trigger select 4-buffer)) .....	309
9-3.	Correspondence between Analog Input Pin and ADCRn Register (scan mode (A/D trigger scan)) .....	310
9-4.	Correspondence between Analog Input Pin and ADCRn Register (1-buffer mode (timer trigger select 1-buffer)) .....	312
9-5.	Correspondence between Analog Input Pin and ADCRn Register (4-buffer mode (timer trigger select 4-buffer)) .....	313
9-6.	Correspondence between Analog Input Pin and ADCRn Register (scan mode (timer trigger scan)) .....	314
9-7.	Correspondence between Analog Input Pin and ADCRn Register (1-buffer mode (external trigger select 1-buffer)) .....	315

## LIST OF TABLES (2/2)

Table No.	Title	Page
9-8.	Correspondence between Analog Input Pin and ADCRn Register (4-buffer mode (external trigger select 4-buffer)) .....	317
9-9.	Correspondence between Analog Input Pin and ADCRn Register (scan mode (external trigger scan)) .....	318
13-1.	Operating Status of I/O and Output Pins During Reset Period .....	378
13-2.	Initial Values after Reset of Each Register .....	380
14-1.	List of Communication Systems .....	392

**[MEMO]**

## CHAPTER 1 INTRODUCTION

The V854 is a product of NEC's V850 Family single-chip microcontrollers for real-time control applications. This chapter briefly outlines the V854.

### 1.1 General

The V854 is a 32-/16-bit single-chip microcontroller that employs the CPU core of the V850 Family of high-performance 32-bit single-chip microcontrollers for real-time control applications, and integrates peripheral functions such as ROM/RAM, real-time pulse unit, serial interface, A/D converter, and PWM.

The V854 is provided with multiplication instructions that are executed with a hardware multiplier, saturated operation instructions, and bit manipulation instructions that are ideal for digital servo control applications, in addition to the basic instructions that have a high real-time response speed and can be executed in 1 clock cycle. This microcontroller can be employed for many applications including real-time control systems such as AV applications including digital still cameras and camera built-in VCRs; communication applications including portable telephones and portable information terminals. In any of these applications, the V854 demonstrates an extremely high cost effectiveness. Especially, the real-time pulse unit that can realize VCR software servo control is provided; therefore, the system/servo/camera control of camera built-in VCR is realized by one chip.

1.2 Features

- Number of instructions : 74
- Minimum instruction execution time : 30 ns (at internal 33 MHz)
- General register : 32 bits x 32
- Instruction set : Signed multiply (16 bits x 16 bits → 32 bits): 1 to 2 clocks  
Saturated operation instructions (with overflow/underflow detection function)  
32-bit shift instructions: 1 clock  
Bit manipulation instructions  
Load/store instructions with long/short format
- Memory space : 16 Mbytes linear address space (common program/data)  
Memory block division function: 2 Mbytes/block  
Programmable wait function  
Idle state insertion function
- External bus interface : 16-bit data bus (address/data multiplexed)  
Bus hold function  
External wait function

★ ○ Internal memory

Part Number	Internal ROM	Internal RAM
μPD703006	None	4 Kbytes
μPD703008, 703008Y	128 K (Mask ROM)	4 Kbytes
μPD70F3008, 70F3008Y	128 K (Flash memory)	4 Kbytes

- Interrupt/exception : External interrupt: 22 (including NMI)  
Internal interrupt : 31 sources  
Exception : 1 source  
Eight levels of priorities can be set.
- I/O line : Input port: 16  
I/O port : 96
- Real-time pulse unit : 24-bit timer/event counter: 2 ch  
16-bit interval timer: 6 ch
- Serial interface : Asynchronous serial interface (UART)  
Synchronous or clocked serial interface (CSI)  
I<sup>2</sup>C bus interface (I<sup>2</sup>C) (μPD703008Y and 70F3008Y only)  
UART/CSI: 1 ch  
CSI: 2 ch  
CSI/I<sup>2</sup>C: 1 ch  
Dedicated baud rate generator: 4 ch

- PWM (Pulse Width Modulation) : 12- to 16-bit resolution PWM: 4 ch
- A/D converter : 8-bit resolution A/D converter: 16 ch
- Clock generator : Multiplication function by PLL clock synthesizer (multiplication by one or five)  
2-frequency division function by external clock
- Power save function : HALT/IDLE/software STOP mode  
Clock output stop function
- Package : 144-pin plastic LQFP: pin pitch: 0.5 mm
- CMOS technology : Complete static circuit

### 1.3 Application Fields

- System/servo/camera control of camera built-in VCRs, etc.
- Portable cameras such as digital still cameras, etc.
- Portable telephones and portable information terminals, etc.

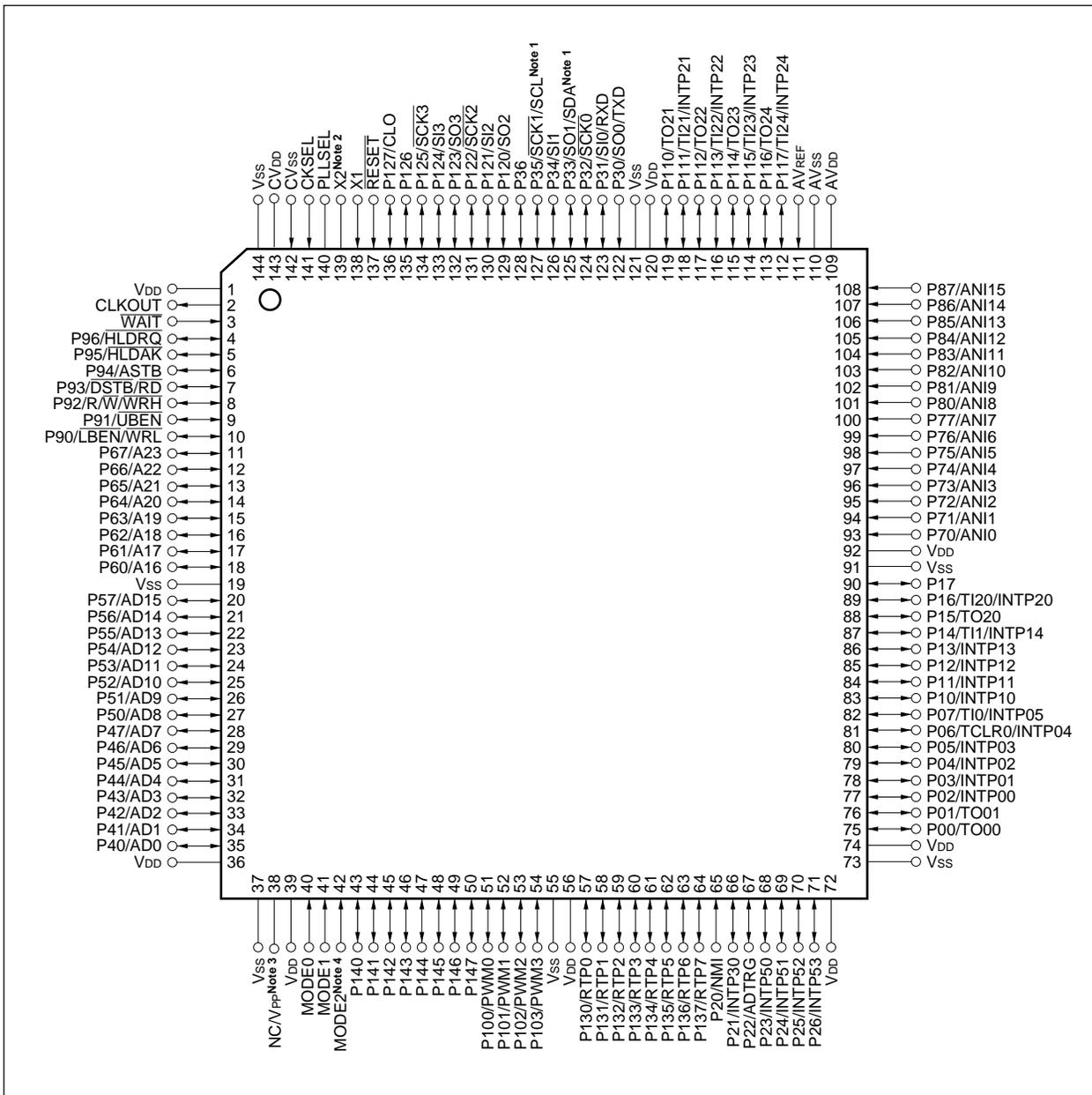
### ★ 1.4 Ordering Information

Part number	Package	Internal ROM
μPD703006GJ-33-8EU	144-pin plastic LQFP (fine pitch) (20 × 20 mm)	None
μPD703008GJ-25-xxx-8EU	144-pin plastic LQFP (fine pitch) (20 × 20 mm)	Mask ROM
μPD703008YGJ-25-xxx-8EU	144-pin plastic LQFP (fine pitch) (20 × 20 mm)	Mask ROM
μPD703008YGJ-33-xxx-8EU	144-pin plastic LQFP (fine pitch) (20 × 20 mm)	Mask ROM
μPD70F3008GJ-16-8EU <sup>Note</sup>	144-pin plastic LQFP (fine pitch) (20 × 20 mm)	Flash memory
μPD70F3008YGJ-16-8EU <sup>Note</sup>	144-pin plastic LQFP (fine pitch) (20 × 20 mm)	Flash memory

**Note** Under development

**Remark** xxx indicates ROM code suffix.

★ 1.5 Pin Identification (Top View)

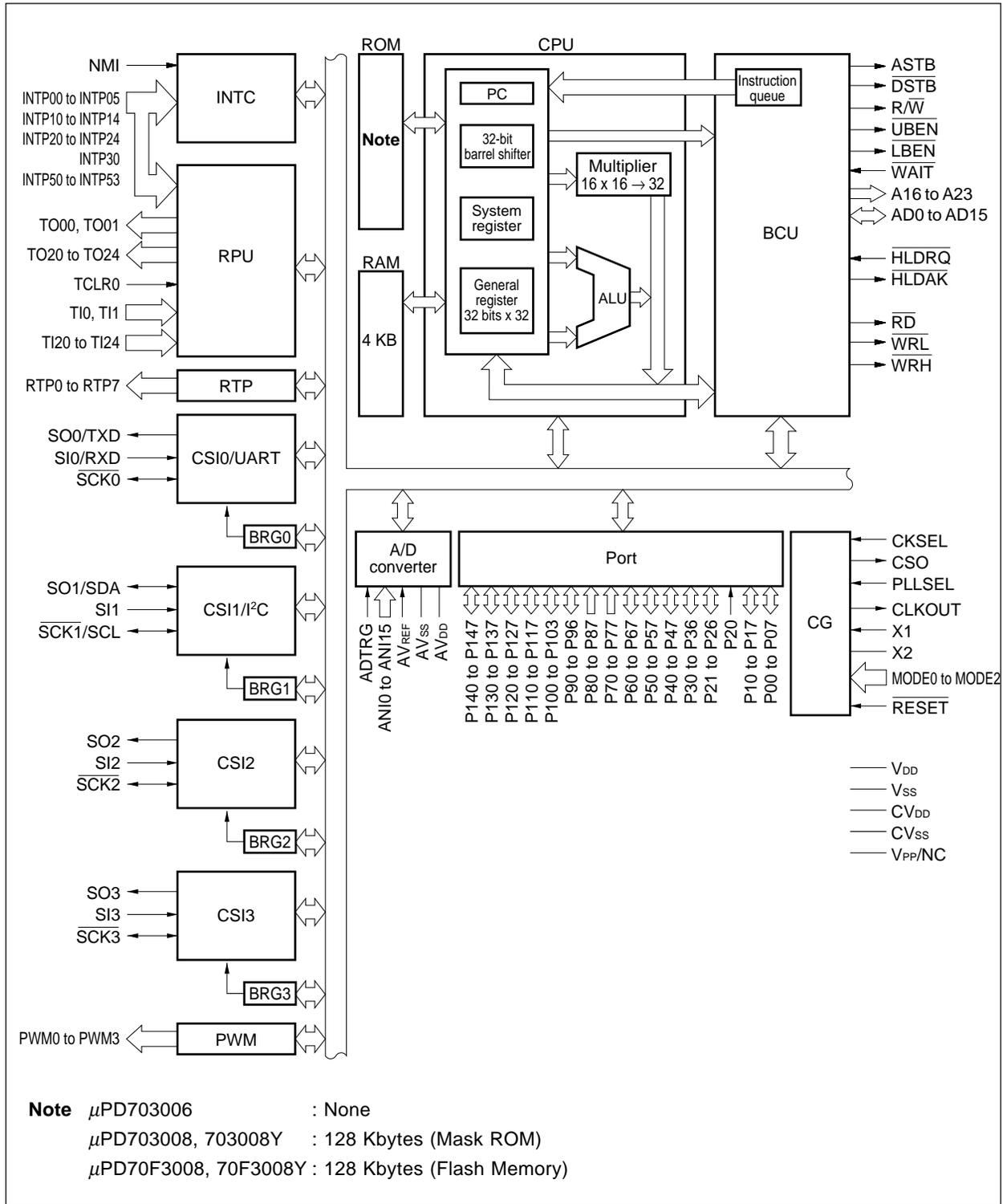


- Notes**
1. SCL and SDA are available only for  $\mu$ PD703008Y and 70F3008Y.
  2. Leave open when external clock is connected to X1 pin.
  3.  $\mu$ PD703006, 703008, 703008Y : NC  
 $\mu$ PD70F3008, 70F3008Y :  $V_{PP}$  (Connect to  $V_{SS}$  via a resistor ( $R_{VPP}$ ) in normal operating mode)
  4. Connect directly to  $V_{SS}$  in the normal operation mode.

<b>Pin name</b>		P00 to P07	: Port0
		P10 to P17	: Port1
A16 to A23	: Address Bus	P20 to P26	: Port2
AD0 to AD15	: Address/Data Bus	P30 to P36	: Port3
ADTRG	: AD Trigger Input	P40 to P47	: Port4
ANI0 to ANI15	: Analog Input	P50 to P57	: Port5
ASTB	: Address Strobe	P60 to P67	: Port6
AV <sub>DD</sub>	: Analog V <sub>DD</sub>	P70 to P77	: Port7
AV <sub>REF</sub>	: Analog Reference Voltage	P80 to P87	: Port8
AV <sub>SS</sub>	: Analog V <sub>SS</sub>	P90 to P96	: Port9
CKSEL	: Clock Select	P100 to P103	: Port10
CLKOUT	: Clock Output	P110 to P117	: Port11
CLO	: Clock Output (Divided)	P120 to P127	: Port12
CV <sub>DD</sub>	: Power Supply for Clock Generator	P130 to P137	: Port13
CV <sub>SS</sub>	: Ground for Clock Generator	P140 to P147	: Port14
$\overline{\text{DSTB}}$	: Data Strobe	PLLSEL	: PLL Select
$\overline{\text{HLD\!A\!K}}$	: Hold Acknowledge	PWM0 to PWM3	: Pulse Width Modulation
$\overline{\text{HLDR\!Q}}$	: Hold Request	$\overline{\text{RD}}$	: Read
INTP00 to	: Interrupt Request from Peripherals	$\overline{\text{RESET}}$	: Reset
INTP05,		RTP0 to RTP7	: Real-time Port
INTP10 to		$\overline{\text{R\!W}}$	: Read/Write Status
INTP14,		RXD	: Receive Data
INTP20 to		$\overline{\text{SCK0}}$ to $\overline{\text{SCK3}}$	: Serial Clock
INTP24,		SCL	: Serial Clock
INTP30,		SDA	: Serial Data
INTP50 to		SI0 to SI3	: Serial Input
INTP53		SO0 to SO3	: Serial Output
$\overline{\text{LBEN}}$	: Lower Byte Enable	TCLR0	: Timer Clear
MODE0 to	: Mode	TI0, TI1,	: Timer Input
MODE2		TI20 to TI24	
NC	: No Connection	TO00, TO01,	: Timer Output
NMI	: Non-maskable Interrupt Request	TO20 to TO24	
		TXD	: Transmit Data
		$\overline{\text{UBEN}}$	: Upper Byte Enable
		V <sub>DD</sub>	: Power Supply
		V <sub>PP</sub>	: Programming Power Supply
		V <sub>SS</sub>	: Ground
		$\overline{\text{WAIT}}$	: Wait
		$\overline{\text{WRH}}$	: Write Strobe High Level Data
		$\overline{\text{WRL}}$	: Write Strobe Low Level Data
		X1, X2	: Crystal

## 1.6 Function Block Configuration

### ★ 1.6.1 Internal block diagram



## 1.6.2 Internal units

### (1) CPU

Executes almost all instruction processing such as address calculation, arithmetic/logic operation, and data transfer in 1 clock by using a 5-stage pipeline.

Dedicated hardware devices such as a multiplier (16 bits × 16 bits → 32 bits) and a barrel shifter (32 bits) are provided to increase the speed of processing complicated instructions.

- ★
- ### (2) Bus control unit (BCU)
- Initiates the necessary number of external bus cycles based on the physical address obtained by the CPU. If the CPU does not issue a request to start a bus cycle when an instruction is fetched from external memory area, it generates a prefetch address to prefetch an instruction code. The prefetched instruction code is loaded into the internal instruction queue.
- ★
- ### (3) Internal ROM
- The  $\mu$ PD703008 and 703008Y incorporate mask ROM (128 Kbytes) and the  $\mu$ PD70F3008 and 70F3008Y incorporate flash memory (128 Kbytes). They are each mapped starting from address 00000000H. The  $\mu$ PD703006 does not contain internal ROM.
- Access is enabled/disabled by the MODE0 to MODE2 pins. With the internal flash memory device, the programming mode is specified by these two pins.
- This internal ROM is accessed in 1 clock by the CPU when an instruction is fetched.
- ### (4) Internal RAM
- 4 Kbytes RAM is mapped starting from address FFFFE000H. This RAM can be accessed in 1 clock by the CPU when data is accessed.
- ### (5) Interrupt controller (INTC)
- Processes interrupt requests (NMI, INTP00 to INTP05, INTP10 to INTP14, INTP20 to INTP24, INTP30, and INTP50 to INTP53) from the internal peripheral hardware and external sources. Eight levels of priorities can be specified for these interrupt requests, and multiplexed processing control can be performed on an interrupt source.
- ### (6) Clock generator (CG)
- By the internal PLL, supplies the CPU clock whose frequency is five times, one time (when internal PLL is used), or 1/2 times (when internal PLL is not used) the frequency of the oscillator connected across the X1 and X2 pins. Input from an external clock source can also be referenced instead of using the oscillator.
- ### (7) Real-time pulse unit (RPU)
- Provides two 24-bit timer/event counter channels, six 16-bit interval timer channels, and capabilities for measuring pulse width and generation of programmable pulse outputs.
- ### (8) Serial interface (SIO)
- The serial interface consists of 4 channels in total of asynchronous serial interfaces (UART) and synchronous or clocked serial interfaces (CSI). One of these channels can be switched between UART and CSI, one channel can be switched between CSI and I<sup>2</sup>C ( $\mu$ PD703008Y and 70F3008Y only), and the other two channels are fixed to CSI.
- UART transfers data by using the TXD and RXD pins.
- CSI transfers data by using the SO, SI, and  $\overline{\text{SCK}}$  pins.
- I<sup>2</sup>C transfers data by using the SDA and SCL pins.
- The serial clock source can be selected from the baud rate generator output and system clock.

**(9) Ports**

The ports have functions as general ports and functions as control pins as shown below.

Port	I/O	Port Function	Control Function
Port0	8-bit I/O	General port	Timer I/O, external interrupt
Port1			
Port2	1-bit input, 6-bit I/O		NMI, A/D converter trigger, external interrupt
Port3	7-bit I/O		Serial interface
Port4	8-bit I/O		External address/data bus
Port5			
Port6			External address bus
Port7	8-bit input		A/D converter analog input
Port8			
Port9	7-bit I/O		External bus interface control signal I/O
Port10	4-bit I/O		PWM output
Port11	8-bit I/O		Timer I/O, external interrupt
Port12			Serial interface
Port13			Real time output port
Port14		—	

**(10) PWM (Pulse Width Modulation)**

The V854 is provided with four channels of PWM signal output for which the 12- to 16-bit resolutions can be selected. The PWM output can be used as a D/A converter output by connecting an external low pass filter. It is suitable for controlling the actuator of motors, etc.

**(11) A/D converter**

This is a high speed, high resolution 8-bit A/D converter with 16 analog input pins. Converts with a sequential conversion method.

**(12) RTP**

Real time output function which transfers the 8-bit data previously set to output latch with the coincidence signal of compare register. RTP can output data to port at the accurate timing specified with timer.

## CHAPTER 2 PIN FUNCTIONS

The following table shows the names and functions of the V854's pins. These pins can be divided by function into port pins and other pins.

### 2.1 Pin Function List

#### (1) Port pins

(1/2)

Pin Name	I/O	Function	Alternate Function
P00	I/O	Port 0. 8-bit I/O port. Can be specified in input/output mode in 1-bit units.	TO00
P01			TO01
P02			INTP00
P03			INTP01
P04			INTP02
P05			INTP03
P06			TCLR0/INTP04
P07			T10/INTP05
P10	I/O	Port 1. 8-bit I/O port. Can be specified in input/output mode in 1-bit units.	INTP10
P11			INTP11
P12			INTP12
P13			INTP13
P14			T11/INTP14
P15			TO20
P16			T120/INTP20
P17			-
P20	Input	Port 2. P20 is input-only port. This pin operates as NMI input when valid edge is input. Bit 0 in P2 register signifies NMI input state. P21 to P26 are 6-bit input/output port pins. Can be specified in input/output mode in 1-bit units.	NMI
P21	I/O		INTP30
P22			ADTRG
P23			INTP50
P24			INTP51
P25			INTP52
P26			INTP53
P30	I/O	Port 3. 7-bit I/O port. Can be specified in input/output mode in 1-bit units.	SO0/TXD
P31			SI0/RXD
P32			$\overline{\text{SCK0}}$
P33			SO1/SDA
P34			SI1
P35			$\overline{\text{SCK1/SCL}}$
P36			-
P40 to P47	I/O	Port 4. 8-bit I/O port. Can be specified in input/output mode in 1-bit units.	AD0 to AD7

Pin Name	I/O	Function	Alternate Function
P50 to P57	I/O	Port 5. 8-bit I/O port. Can be specified in input/output mode in 1-bit units.	AD8 to AD15
P60 to P67	I/O	Port 6. 8-bit I/O port. Can be specified in input/output mode in 1-bit units.	A16 to A23
P70 to P77	Input	Port 7. 8-bit input only port.	ANI0 to ANI7
P80 to P87	Input	Port 8. 8-bit input only port.	ANI8 to ANI15
P90	I/O	Port 9. 7-bit I/O port. Can be specified in input/output mode in 1-bit units.	$\overline{\text{LBEN}}/\overline{\text{WRL}}$
P91			$\overline{\text{UBEN}}$
P92			$\overline{\text{R/W}}/\overline{\text{WRH}}$
P93			$\overline{\text{DSTB}}/\overline{\text{RD}}$
P94			ASTB
P95			$\overline{\text{HLDK}}$
P96			$\overline{\text{HLDRQ}}$
P100	I/O	Port 10. 4-bit I/O port. Can be specified in input/output mode in 1-bit units.	PWM0
P101			PWM1
P102			PWM2
P103			PWM3
P110	I/O	Port 11. 8-bit I/O port. Can be specified in input/output mode in 1-bit units.	TO21
P111			TI21/INTP21
P112			TO22
P113			TI22/INTP22
P114			TO23
P115			TI23/INTP23
P116			TO24
P117			TI24/INTP24
P120	I/O	Port 12. 8-bit I/O port. Can be specified in input/output mode in 1-bit units.	SO2
P121			SI2
P122			$\overline{\text{SCK2}}$
P123			SO3
P124			SI3
P125			$\overline{\text{SCK3}}$
P126			–
P127			CLO
P130 to P137			I/O
P140 to P147	I/O	Port 14. 8-bit I/O port. Can be specified in input/output mode in 1-bit units.	–

(2) Pins other than port pins

(1/3)

Pin Name	I/O	Function	Alternate Function
TO00	Output	Pulse signal output from timer 0 and 2.	P00
TO01			P01
TO20			P15
TO21			P110
TO22			P112
TO23			P114
TO24			P116
TCLR0	Input	External clear signal input to timer 0.	P06/INTP04
TI0	Input	External count clock input to timer 0, 1, and 2.	P07/INTP05
TI1			P14/INTP14
TI20			P16/INTP20
TI21			P111/INTP21
TI22			P113/INTP22
TI23			P115/INTP23
TI24			P117/INTP24
INTP00 to INTP03	Input	External capture trigger input to timer 0. Also used to input external maskable interrupt request.	P02 to P05
INTP04	Input	External maskable interrupt request input.	P06/TCLR0
INTP05			P07/TI0
INTP10 to INTP13	Input	External capture trigger input to timer 1. Also used to input external maskable interrupt request.	P10 to P13
INTP14	Input	External maskable interrupt request input.	P14/TI1
INTP20	Input	External maskable interrupt request input.	P16/TI20
INTP21			P111/TI21
INTP22			P113/TI22
INTP23			P115/TI23
INTP24			P117/TI24
INTP30	Input	External capture trigger input to timer 3. Also used to input external maskable interrupt request.	P21
INTP50 to INTP53	Input	External maskable interrupt request input.	P23 to P26
NMI	Input	Non-maskable interrupt request input.	P20

Pin Name	I/O	Function	Alternate Function
SO0	Output	Serial transmit data output from CSI0 to CSI3 (3-wire).	P30/TXD
SO1			P33/SDA
SO2			P120
SO3			P123
SI0	Input	Serial receive data input to CSI0 to CSI3 (3-wire).	P31/RXD
SI1			P34
SI2			P121
SI3			P124
$\overline{\text{SCK0}}$	I/O	Serial clock I/O from/to CSI0 to CSI3 (3-wire).	P32
$\overline{\text{SCK1}}$			P35/SCL
$\overline{\text{SCK2}}$			P122
$\overline{\text{SCK3}}$			P125
SDA	I/O	Serial transmit/receive data I/O from/to I <sup>2</sup> C.	P33/SO1
SCL		Serial clock I/O from/to I <sup>2</sup> C.	P35/ $\overline{\text{SCK1}}$
TXD	Output	Serial transmit data output from UART.	P30/SO0
RXD	Input	Serial receive data input to UART.	P31/SI0
PWM0 to PWM3	Output	Pulse signal output from PWM.	P100 to P103
AD0 to AD7	I/O	16-bit multiplexed address/data bus when external memory is used.	P40 to P47
AD8 to AD15			P50 to P57
A16 to A19	Output	Higher address bus when external memory is used.	P60 to P67
$\overline{\text{LBEN}}$	Output	Lower byte enable signal output of external data bus.	P90/ $\overline{\text{WRL}}$
$\overline{\text{UBEN}}$		Higher byte enable signal output of external data bus.	P91
R/ $\overline{\text{W}}$		External read/write status output.	P92/ $\overline{\text{WRH}}$
$\overline{\text{DSTB}}$		External data strobe signal output.	P93/ $\overline{\text{RD}}$
ASTB		External address strobe signal output.	P94
$\overline{\text{HLDAK}}$	Output	Bus hold acknowledge output.	P95
$\overline{\text{HLDRQ}}$	Input	Bus hold request input.	P96
$\overline{\text{WRL}}$	Output	Lower byte write strobe signal output to external data bus.	P90/ $\overline{\text{LBEN}}$
$\overline{\text{WRH}}$		Higher byte write strobe signal output to external data bus.	P92/R/ $\overline{\text{W}}$
$\overline{\text{RD}}$	Output	Read strobe signal output to external data bus.	P93/ $\overline{\text{DSTB}}$
ANI0 to ANI7	Input	Analog input to A/D converter.	P70 to P77
ANI8 to ANI15			P80 to P87
RTP0 to RTP7	Output	Real time output port.	P130 to P137
CLO	Output	System clock output (with frequency division function).	P127
CKSEL	Input	Input to specify clock generator operation mode.	–
PLLSEL	Input	Input to specify the number of PLL multiplication.	–
CLKOUT	Output	System clock output.	–

Pin Name	I/O	Function	Alternate Function
$\overline{\text{WAIT}}$	Input	Control signal input inserting wait state to bus cycle.	–
MODE0 to MODE2	Input	Specifies operation mode.	–
$\overline{\text{RESET}}$	Input	System reset input.	–
X1	Input	System clock oscillator connecting pins. Supply external clock to X1.	–
X2	–		–
ADTRG	Input	A/D converter external trigger input.	P22
AV <sub>REF</sub>	Input	Reference voltage input for A/D converter.	–
AV <sub>DD</sub>	–	Positive power supply for A/D converter.	–
AV <sub>SS</sub>	–	Ground for A/D converter.	–
CV <sub>DD</sub>	–	Positive power supply for clock generator.	–
CV <sub>SS</sub>	–	Ground for clock generator.	–
V <sub>DD</sub>	–	Positive power supply.	–
V <sub>SS</sub>	–	Ground.	–
V <sub>PP</sub>	–	High voltage applying pin for program write/verify. ( $\mu$ PD70F3008, 70F3008Y only)	–
NC	–	Internally unconnected ( $\mu$ PD703006, 703008, 703008Y only)	–

★

## 2.2 Pin Status

The operating status of each pin in each operation mode is as follows:

Operating Status Pin	Reset	Software STOP Mode	IDLE Mode	Bus Hold	Idle State	HALT Mode
AD0 to AD15	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z
A16 to A23	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Retained <sup>Note 1</sup>	Retained
$\overline{\text{LBEN}}$ , $\overline{\text{UBEN}}$	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Retained <sup>Note 1</sup>	Retained
$\overline{\text{R/W}}$	Hi-Z	Hi-Z	Hi-Z	Hi-Z	H	H
$\overline{\text{DSTB}}$ , $\overline{\text{WRL}}$ , $\overline{\text{WRH}}$ , $\overline{\text{RD}}$	Hi-Z	Hi-Z	Hi-Z	Hi-Z	H	H
$\overline{\text{ASTB}}$	Hi-Z	Hi-Z	Hi-Z	Hi-Z	H	H
$\overline{\text{HLDRQ}}$	–	–	–	Operates	Operates	Operates
$\overline{\text{HLDK}}$	Hi-Z	Hi-Z	Hi-Z	L	Operates	Operates
$\overline{\text{WAIT}}$	–	–	–	–	–	–
CLKOUT	Operates <sup>Note 2</sup>	L	–	Operates <sup>Note 2</sup>	Operates <sup>Note 2</sup>	Operates <sup>Note 2</sup>

Hi-Z : high-impedance

Retained : Retains status in external bus cycle immediately before

L : low-level output

H : high-level output

– : input not sampled

**Notes** 1. Undefined immediately after the bus hold end.

2. Low-level output in single chip mode 1, flash memory programming mode, and clock output inhibit mode.

## 2.3 Pin Function

### (1) P00 to P07 (Port0) ... 3-state I/O

These pins constitute an 8-bit I/O port, port 0. They also serve as control signal pins.

P00 to P07 function not only as I/O port pins, but also as the I/O pins of the real-time pulse unit (RPU) and external interrupt request input pins. Each bit of port 0 can be specified in the port or control mode, by using port 0 mode control register (PMC0).

#### (a) Port mode

P00 to P07 can be set in the input or output mode in 1-bit units by using port 0 mode register (PM0).

#### (b) Control mode

P00 to P07 can be set in the port or control mode in 1-bit units by the PMC0 register.

#### (i) TO00, TO01 (Timer Output) ... output

These are pulse signals output pins for timer 0.

#### (ii) TCLR0 (Timer Clear) ... input

This pin inputs an external clear signal to timer 0.

#### (iii) TI0 (Timer Input) ... input

This pin inputs an external count clock to timer 0.

#### (iv) INTP00 to INTP05 (Interrupt Request from Peripherals) ... input

These pins are the external interrupt request input pins.

### (2) P10 to P17 (Port 1) ... 3-state I/O

These pins constitute an 8-bit I/O port, port 1, which can be set in the input or output mode in 1-bit units.

P10 to P17 function as ports, as well as RPU inputs/outputs and external interrupt request inputs. In the operation mode, port control can be selected in 1-bit units, and is specified by the port 1 mode control register (PMC1).

#### (a) Port mode

P10 to P17 can be set in the input or output mode in 1-bit units by using port 1 mode register (PM1).

#### (b) Control mode

P10 to P17 can be set in the port or control mode in 1-bit units by the PMC1 register.

#### (i) TO20 (Timer Output) ... output

This is pulse signal output pin for timer 2.

#### (ii) TI1, TI20 (Timer Input) ... input

These pins are external count clock input pins of timer 1 and timer 2.

#### (iii) INTP10 to INTP14 (Interrupt Request from Peripherals) ... input

These pins are the external interrupt request input pins and capture trigger input pins of timer 1.

#### (iv) INTP20 (Interrupt Request from Peripherals) ... input

This pin is the external interrupt request input pin.

**(3) P20 to P26 (Port 2) ... 3-state I/O**

These pins constitute an I/O port, port 2, which can be set in the input or output mode in 1-bit units, except P20, which is an input only pin. These pins function not only as port pins but also as NMI input, external interrupt request input, and A/D converter external trigger.

Each bit of this port can be specified in the port or control mode by using port 2 mode control register (PMC2).

**(a) Port mode**

P21 to P26 can be set in the input or output mode in 1-bit units by port 2 mode register (PM2).

P20 is the input-only port and operates as NMI input when a valid edge is input.

**(b) Control mode**

P21 to P26 can be set in the port or control mode in 1-bit units by the PMC2 register.

P20 is fixed to control mode.

**(i) NMI (Non-maskable Interrupt Request) ... input**

This is an input pin for the non-maskable interrupt request signal.

**(ii) INTP30, INTP50 to INTP53 (Interrupt Request from Peripherals) ... input**

These pins are the external interrupt request input pins.

**(iii) ADTRG (AD Trigger Input) ... input**

This pin is external trigger input pin of A/D converter.

**(4) P30 to P36 (Port 3) ... 3-state I/O**

These pins constitute an 7-bit I/O port, port 3. They also function as control signal pins. P30 to P36 function not only as I/O port pins but also as serial interface I/O pins in the control mode.

Each bit of port 3 can be specified in the port or control mode in 1-bit units by using port 3 mode control register (PMC3).

**(a) Port mode**

P30 to P36 can be set in the input or output mode in 1-bit units by port 3 mode register (PM3).

**(b) Control mode**

P30 to P36 can be set in the port or control mode in 1-bit units by the PMC3 register.

**(i) TXD (Transmit Data) ... output**

This is a serial transmit data output pin for the UART.

When transmission is disabled: High impedance

When transmission is enabled: High level

**(ii) RXD (Receive Data) ... input**

This is a serial receive data input pin for the UART.

**(iii) SDA (Serial Data)...I/O**

This pin is I<sup>2</sup>C serial receive data I/O pin. Since this pin is open drain output, connect external pull-up resistor. ( $\mu$ PD703008Y and 70F3008Y only)

**(iv) SCL (Serial Clock) ... I/O**

This pin is I<sup>2</sup>C serial clock I/O pin. Since this pin is the open-drain output, connect external pull-up resistor. ( $\mu$ PD703008Y and 70F3008Y only)

**(v) SO0, SO1 (Serial Output 0, 1) ... output**

These are serial transmit data output pin for the CSI.

Since SO1 is open drain output, connect external pull-up resistor.

**(vi) SI0, SI1 (Serial Input 0, 1) ... input**

These are serial receive data input pin for the CSI.

**(vii)  $\overline{\text{SCK0}}$ ,  $\overline{\text{SCK1}}$  (Serial Clock 0, 1) ... 3-state I/O**

These pins input/output the serial clock of CSI.

Since  $\overline{\text{SCK1}}$  is open drain output, connect external pull-up resistor to output.

★ **(5) P40 to P47 (Port 4) ... 3-state I/O**  
 These pins constitute an 8-bit I/O port, port 4. They also form a portion of the address/data bus connected to external memory.

P40 to P47 function not only as I/O port pins but also as time sharing address/data bus pins (AD0 to AD7) in the control mode (external expansion mode) when an external memory is connected.

Operation mode is specified by the mode specification pin (MODE) and the memory expansion mode register (MM).

**(a) Port mode**

P40 to P47 can be set in the input or output port mode in 1-bit units by using port 4 mode register (PM4).

**(b) Control mode (External Expansion Mode)**

P40 to P47 can be specified as AD0 to AD7 by using the MODE pin and MM register.

**(i) AD0 to AD7 (Address/Data 0 to 7) ... 3-state I/O**

These pins constitute a multiplexed address/data bus when the external memory is accessed. They function as the A0 to A7 output pins of a 24-bit address in the address timing (T1 state), and as the lower 8-bit data I/O bus pins of 16-bit data in the data timing (T2, TW, T3).

The output status of these pins changes in synchronization with the rising edge of the clock in each state of the bus cycle. AD0 to AD7 go into a high-impedance state in the idle state (T1).

Since SO1 is open drain output, connect external pull-up resistor to output.

★ **(6) P50 to P57 (Port 5) ... 3-state I/O**  
 These pins constitute an 8-bit I/O port, port 5. They also form a portion of the address/data bus connected to external memory.

P50 to P57 function not only as I/O port pins but also as multiplexed address/data bus pins (AD8 to AD15) in the control mode (external expansion mode) when an external memory is connected.

Operation mode is specified by the mode specification pin (MODE) and memory expansion mode register (MM).

**(a) Port mode**

P50 to P57 can be set in the input or output port mode in 1-bit units by using port 5 mode register (PM5).

**(b) Control mode (External Expansion Mode)**

P50 to P57 can be specified as AD8 to AD15 by using the MODE pin and MM register.

**(i) AD8 to AD15 (Address/Data 8 to 15) ... 3-state I/O**

These pins constitute a multiplexed address/data bus when the external memory is accessed. They function as the A8 to A15 output pins of a 24-bit address in the address timing (T1 state), and as the higher 8-bit data I/O bus pins of 16-bit data in the data timing (T2, TW, T3).

The output status of these pins changes in synchronization with the rising of the clock in each state of the bus cycle. AD8 to AD15 go into a high-impedance state in the idle state (T1).

**★ (7) P60 to P67 (Port 6) ... 3-state I/O**

These pins constitute an 8-bit I/O port, port 6. They also form a portion of the address/data bus connected to external memory.

P60 to P67 function not only as I/O port pins but also as address bus pins (A16 to A23) in the control mode (external expansion mode) when an external memory is connected. This port can be set in the port or control mode in 2-bit units by using mode specification pin (MODE) and memory expansion mode register (MM).

**(a) Port mode**

P60 to P67 can be set in the input or output port mode in 1-bit units by using port 6 mode register (PM6).

**(b) Control mode (External Expansion Mode)**

P60 to P67 can be specified as A16 to A23 by using the MODE pin and MM register.

**(i) A16 to A23 (Address 16 to 23) ... output**

These pins constitute the higher 8 bits of a 24-bit address bus when the external memory is accessed.

The output status of these pins changes in synchronization with the rising edge of the clock in the T1 state. During the idle state (T1), the address of the bus cycle immediately before entering the idle state is retained.

**(8) P70 to P77 (Port 7), P80 to P87 (Port 8) ... input**

Port 7 and port 8 each are an 8-bit input-only port whose pins are all fixed to input.

P70 to P77 and P80 to P87 function as input ports, as well as A/D converter analog inputs in the control mode. However, the input port and analog input pin cannot be switched.

**(a) Port mode**

P70 to P77 and P80 to P87 are dedicated input ports.

**(b) Control mode**

P70 to P77 also function as ANI0 to ANI7 pins and P80 to P87 also function as ANI8 to ANI15 pins, and cannot be switched.

**(i) ANI0 to ANI15 (Analog Input) ... input**

These pins are analog input pins to A/D converter.

To prevent erroneous operation due to noise, connect a capacitor between these pins and AV<sub>SS</sub>. Make sure that voltage other than the range of AV<sub>SS</sub> and AV<sub>REF</sub> should not be applied to the input pin used for A/D converter input. If there should be the possibility that noise whose voltage is AV<sub>REF</sub> or more or noise whose voltage is AV<sub>SS</sub> or less is generated, clamp the voltage using a diode with small V<sub>F</sub>.

★ **(9) P90 to P96 (Port 9) ... 3-state I/O**

These pins constitute a 7-bit I/O port, port 9, and are also used to output control signals.

P90 to P96 function not only as I/O port pins but also as control signal output pins and bus hold control signal output pins in the control mode (external expansion mode) when an external memory is used.

If port 9 is accessed in 8-bit units, the higher 1-bit is ignored if the access is write, and undefined if the access is read.

Operation mode is specified by the mode specification pin (MODE) and memory expansion mode register (MM).

**(a) Port mode**

P90 to P96 can be set in the input or output port mode in 1-bit units by using port 9 mode register (PM9).

**(b) Control mode (External Expansion Mode)**

P90 to P96 can be used to output control signals when so specified by the MODE pin and MM register when an external memory is used.

**(i)  $\overline{\text{LBEN}}$  (Lower Byte Enable) ... output**

This is the lower byte enable signal of the 16-bit external data bus.

This signal changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. The status of the bus signal remains unchanged in the idle state (TI).

**(ii)  $\overline{\text{UBEN}}$  (Upper Byte Enable) ... output**

This is the upper byte enable signal of the 16-bit external data bus. It becomes active (low) in byte access to an odd address. It becomes inactive (high) in byte access to an even address.

This signal changes in synchronization with the rising of the clock in the T1 state of the bus cycle. The status of the bus signal remains unchanged in the idle state (TI).

Access		$\overline{\text{UBEN}}$	$\overline{\text{LBEN}}$	A0
Word Access		0	0	0
Half-word Access		0	0	0
Byte Access	Even address	1	0	0
	Odd address	0	1	1

**(iii)  $\overline{\text{R/W}}$  (Read/Write Status) ... output**

This is a status signal output pin that indicates whether the bus cycle for external access is a read or write cycle. It goes high in the read cycle and low in the write cycle.

This signal changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. It goes high in the idle state (TI).

**(iv)  $\overline{\text{DSTB}}$  (Data Strobe) ... output**

This is the access strobe signal of the external data bus.

It becomes active (low) in the T2 or TW state of the bus cycle, and becomes inactive (high) in the idle state (TI).

**(v)  $\overline{\text{ASTB}}$  (Address Strobe) ... output**

This is the latch strobe signal of the external address bus.

It becomes active (low) in synchronization with the falling edge of the clock in the T1 state of the bus cycle, and becomes inactive (high) in synchronization with the falling edge of the clock in the T3 state. It goes high in the idle state (TI).

**(vi)  $\overline{\text{HLDAK}}$  (Hold Acknowledge) ... output**

This is an acknowledge signal output pin that indicates that the V854 has set the address bus, data bus, and control bus in the high-impedance state in response to a bus hold request.

As long as this signal is active, the address bus, data bus, and control bus remain in a high impedance state.

**(vii)  $\overline{\text{HLDRQ}}$  (Hold Request) ... input**

This input pin is used by an external device to request that the V854 relinquish control of the address bus, data bus, and control bus. This pin can be input asynchronously with CLKOUT. When this signal becomes active, the V854 sets the address bus, data bus, and control bus in the high-impedance state, after the current bus cycle completes. If there is no current bus activity, the address bus, data bus, and control bus are immediately set to high-impedance.  $\overline{\text{HLDAK}}$  is then made active and the bus and control lines are released.

**(viii)  $\overline{\text{WRL}}$  (Write Strobe Low Byte Data) ... output**

This is a write strobe signal output pin for the lower byte of the external 16-bit data bus.

**(ix)  $\overline{\text{WRH}}$  (Write Strobe High Byte Data) ... output**

This is a write strobe signal output pin for the upper byte of the external 16-bit data bus.

**(x)  $\overline{\text{RD}}$  (Read Strobe)... output**

This is a read strobe signal output pin for the external 16-bit data bus.

**(10) P100 to P103 (Port 10) ... 3-state I/O**

Port 10 is a 4-bit I/O port that can be set in the input or output mode in 1-bit units.

In addition to the function as a port, the pins constituting port 10 are used as output of PWM in the control mode.

If port 10 is accessed in 8-bit units, the higher 4-bit is ignored if the access is write, and undefined if the access is read.

**(a) Port mode**

P100 to P103 can be set in the input or output mode, in 1-bit units, by the port 10 mode register (PM10).

**(b) Control mode**

P100 to P103 function as output pins for PWM control signals when the function is enabled by the port 10 mode control register (PMC10).

**(i) PWM0 to PWM 3 (Pulse Width Modulation 0 to 3) ... output**

These are pulse signals output pins for the PWM.

**(11) P110 to P117 (Port 11) ... 3-state I/O**

Port 11 is an 8-bit I/O port that can be set in the input or output mode in 1-bit units. In addition to the function as a port, the pins constituting port 11 are used as input/output of RPU or external interrupt request input.

**(a) Port mode**

P110 to P117 can be set in the input or output mode, in 1-bit units, by the port 11 mode register (PM11).

**(b) Control mode**

P110 to P117 function as input and output pins for timer 2 when the function is enabled by the port 11 mode control register (PMC11).

**(i) TO21 to TO24 (Timer Output) ... output**

These are pulse signals output pins for timer 2.

**(ii) TI21 to TI24 (Timer Input) ... input**

This pin inputs an external counter clock to timer 2.

**(iii) INTP21 to INTP24 (Interrupt Request from Peripherals) ... input**

These pins are the external interrupt request input pins.

**(12) P120 to P127 (Port 12) ... 3-state I/O**

Port 12 is an 8-bit I/O port that can be set in the input or output mode in 1-bit units. In addition to the function as a port, the pins constituting port 12 are used as serial interface I/O in the control mode.

**(a) Port mode**

P120 to P127 can be set in the input or output mode, in 1-bit units, by the port 12 mode register (PM12).

**(b) Control mode**

P120 to P127 function as input and output of serial interface control signal and output of clock signal by setting of port 12 mode control register (PMC12). However, P126 pin functions only as a port.

**(i) SO2, SO3 (Serial Output 2, 3) ... output**

These are serial transmit data output pins for the CSI.

**(ii) SI2, SI3 (Serial Input 2, 3) ... input**

These are serial receive data input pins for the CSI.

**(iii)  $\overline{\text{SCK2}}$ ,  $\overline{\text{SCK3}}$  (Serial Clock 2, 3) ... 3-state I/O**

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

**(iv) CLO (Clock Output (Divided)) ... output**

This is an output pin for the system clock (with frequency division function).

**(13) P130 to P137 (Port 13) ... 3-state I/O**

Port 13 is an 8-bit I/O port that can be set in the input or output mode in 1-bit units. In addition to the function as a port, it is used as real-time output port in the control mode.

**(a) Port mode**

P130 to P137 can be set in the input or output mode, in 1-bit units, by the port 13 mode register (PM13).

**(b) Control mode**

P130 to P137 function as real-time output port by setting of port 13 mode control register (PMC13).

**(i) RTP0 to RTP7 (Real-time Port 0 to 7) ... output**

These pins are real-time output port.

**(14) P140 to P147 (Port 14) ... 3-state I/O**

Port 14 is an 8-bit I/O port that can be set in the input or output mode in 1-bit units. Port 14 functions only as a port.

**(15) CKSEL (Clock Select) ... input**

This is the input pin that specifies the operation mode of the clock generation circuit. The input value of this pin cannot be changed during normal operation.

CKSEL	Operation Mode
0	PLL mode
1	Direct mode

**(16) PLLSEL (PLL Select) ... input**

This is the input pin that specifies the number of PLL multiplication in the PLL mode (CKSEL = 0). The input value of this pin cannot be changed during normal operation.

This pin has no function in the direct mode (CKSEL = 1) and should be treated as an unused pin.

PLLSEL	PLL Multiplication
0	Multiplication by 1
1	Multiplication by 5

★ **(17) CLKOUT (Clock Output) ... output**

This pin outputs the system clock, even during reset in the ROM-less mode. In the single-chip mode 1, the CLKOUT signal is not output until the PSC register is set (low level output). However, in the single-chip mode 2, the CKOUT signal is output.

★ **(18)  $\overline{\text{WAIT}}$  (Wait) ... input**

This control signal input pin inserts a data wait state to the bus cycle, and can be activated asynchronously to CLKOUT. This pin is sampled at the falling edge of the clock in the T2 and TW states of the bus cycle. If the set/hold time for the sampling timing is not satisfied, the wait state may not be inserted.

★ **(19) MODE0 to MODE2 (Mode 0 to 2) ... input**

These pins specify the operation mode of the V854. Operation modes are roughly classified as normal operation mode and flash memory programming mode. Normal operation modes are further classified as single-chip mode and ROM-less mode. The input value of these pins cannot be changed during normal operation. For details, refer to **3.3 Operation Modes**.

**(a)  $\mu$ PD703006**

MODE2	MODE1	MODE0	Operation Mode	
0	0	0	Normal operation mode	ROM-less mode 1
0	0	1		ROM-less mode 2
Other than above			Setting prohibited	

**(b)  $\mu$ PD703008, 703008Y**

MODE2	MODE1	MODE0	Operation Mode	
0	0	0	Normal operation mode	ROM-less mode 1
0	0	1		ROM-less mode 2
0	1	0		Single-chip mode 1
0	1	1		Single-chip mode 2
Other than above			Setting prohibited	

**(c)  $\mu$ PD70F3008, 70F3008Y**

MODE2	MODE1	MODE0	Operation Mode	
0	0	0	Normal operation mode	ROM-less mode 1
0	0	1		ROM-less mode 2
0	1	0		Single-chip mode 1
0	1	1		Single-chip mode 2
1	1	1	Flash memory programming mode	
Other than above			Setting prohibited	

**(20)  $\overline{\text{RESET}}$  (Reset) ... input**

The  $\overline{\text{RESET}}$  signal is an asynchronous input signal. A valid low-level signal on the  $\overline{\text{RESET}}$  pin initiates a system reset, regardless of the clock operation. In addition to normal system initialization/start functions, the  $\overline{\text{RESET}}$  signal is also used for exiting processor power save modes (HALT, IDLE, or STOP).

**(21) X1, X2 (Crystal) ... input**

An oscillator for internal system clock generation is connected across these pins.

An external clock source can also be referenced by connecting the external clock input to the X1 pin and leaving the X2 pin open.

**(22)  $\text{CV}_{\text{DD}}$  (Power Supply for Clock Generator)**

This pin supplies positive power to the clock generator.

**(23)  $\text{CV}_{\text{SS}}$  (Ground for Clock Generator)**

This is the ground pin of the clock generator.

**(24)  $\text{V}_{\text{DD}}$  (Power Supply)**

This pin supplies positive power. Connect all the  $\text{V}_{\text{DD}}$  pins to a positive power supply.

**(25)  $\text{V}_{\text{SS}}$  (Ground)**

This is a ground pin. Connect all the  $\text{V}_{\text{SS}}$  pins to ground.

**(26)  $\text{AV}_{\text{DD}}$  (Analog  $\text{V}_{\text{DD}}$ )**

Analog power supply pin for A/D converter.

**(27)  $\text{AV}_{\text{SS}}$  (Analog  $\text{V}_{\text{SS}}$ )**

Ground pin for A/D converter.

**(28) AV<sub>REF</sub> (Analog Reference Voltage) ... input**

This is a reference voltage input pin for the A/D converter.

**(29) V<sub>PP</sub> (Programming Power Supply)**

This pin supplies positive power for the PROM mode.

This is for  $\mu$ PD70F3008 or 70F3008Y.

**(30) NC (No Connection)**

This pin is not connected internally.

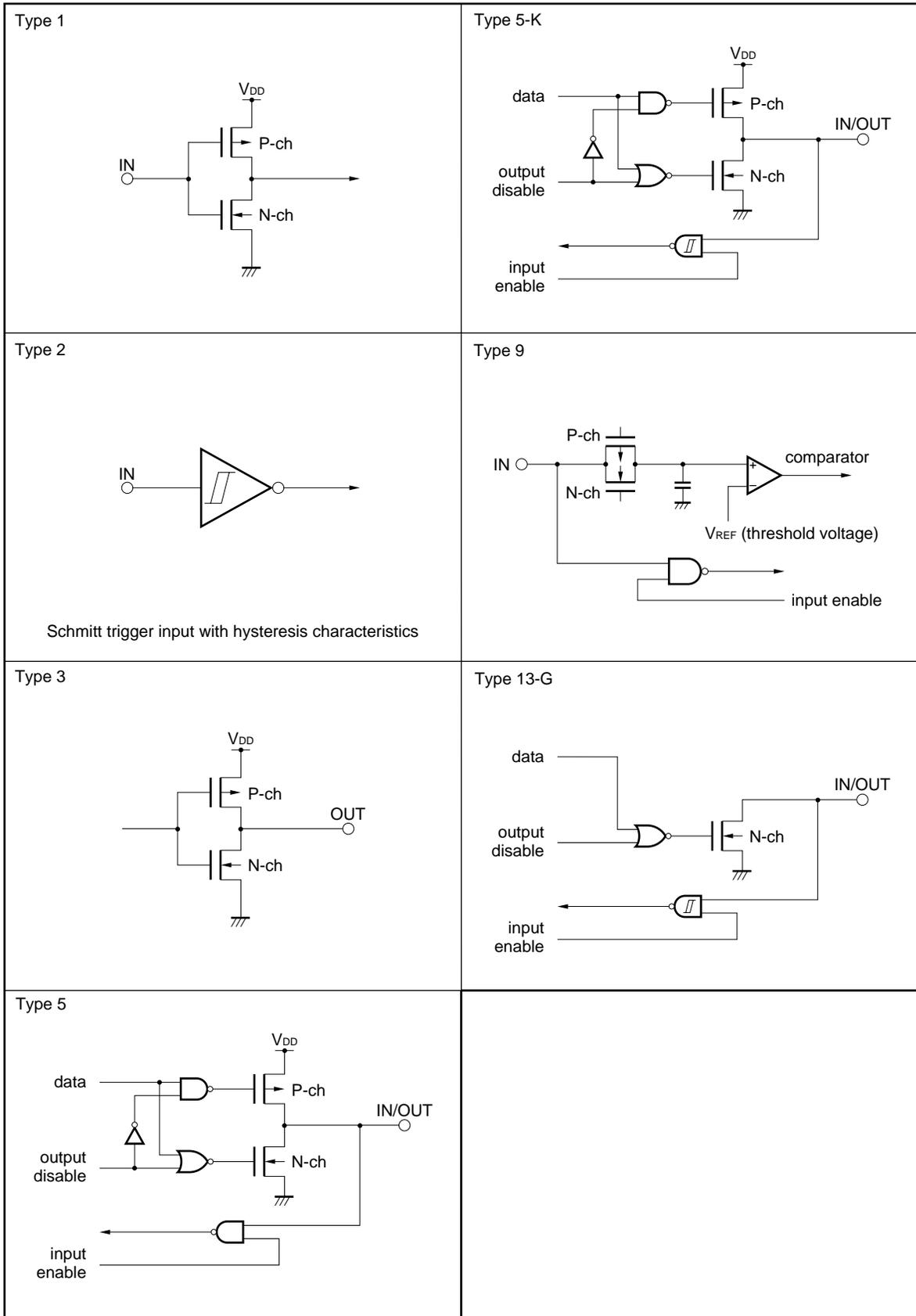
★ This is for the  $\mu$ PD703006, 703008, and 703008Y.

## 2.4 Pin I/O Circuit Type and Connection of Unused Pins

When connecting to  $V_{DD}$  or  $V_{SS}$  via resistor, it is recommended to use 1 to 10-k $\Omega$  resistor.

Pin	I/O Circuit Type	Recommended Connection	
P00/TO00, P01/TO01	5	Independently connect to $V_{DD}$ or $V_{SS}$ via resistor.	
P02/INTP00 to P05/INTP03, P06/TCLR0/INTP04, P07/TI0/INTP05	5-K		
P10/INTP10 to P13/INTP13, P14/TI1/INTP14, P16/TI20/INTP20			
P15/TO20, P17	5		
P20/NMI	2	Connect directly to $V_{SS}$ .	
P21/INTP30, P22/ADTRG, P23/INTP50 to P26/INTP53	5-K	Independently connect to $V_{DD}$ or $V_{SS}$ via resistor.	
P30/TXD/SO0	5		
P31/RXD/SI0, P32/ $\overline{SCK0}$ , P34/SI1	5-K		
P33/SO1/SDA, P35/ $\overline{SCK1}$ /SCL	13-G		
P36	5		
P40/AD0 to P47/AD7	5		Input state: Independently connect to $V_{DD}$ or $V_{SS}$ via resistor.
P50/AD8 to P57/AD15		Output state: Leave open.	
P60/A16 to P67/A23			
P70/ANI0 to P77/ANI7	9	Connect directly to $V_{SS}$ .	
P80/ANI8 to P87/ANI15			
P90/ $\overline{LBEN}$ / $\overline{WRL}$ , P91/ $\overline{UBEN}$ , P92/R/W/WRH, P93/DSTB/RD, P94/ASTB, P95/HLDAK, P96/HLDRQ	5	Input state: Independently connect to $V_{DD}$ or $V_{SS}$ via resistor.  Output state: Leave open.	
P100/PWM0 to P103/PWM3	5	Independently connect to $V_{DD}$ or $V_{SS}$ via resistor.	
P110/TO21, P112/TO22, P114/TO23, P116/TO24			
P111/TI21/INTP21, P113/TI22/INTP22, P115/TI23/INTP23, P117/TI24/INTP24	5-K		
P120/SO2	5		
P121/SI2, P122/ $\overline{SCK2}$	5-K		
P123/SO3	5		
P124/SI3, P125/ $\overline{SCK3}$	5-K		
P126, P127/CLO	5		
P130/RTP0 to P137/RTP7			
P140 to P147			
$\overline{WAIT}$	1		Connect directly to $V_{DD}$ .
CLKOUT	3		Leave open.
MODE0 to MODE2	2		–
$\overline{RESET}$			
$AV_{REF}$ , $AV_{SS}$ , $CV_{SS}$	–	Connect directly to $V_{SS}$ .	
$AV_{DD}$ , $CV_{DD}$	–	Connect directly to $V_{DD}$ .	
PLLSEL	1	Connect directly to $V_{DD}$ or $V_{SS}$ .	
CKSEL			
$V_{PP}/NC$	–	Connect to $V_{SS}$ via resistor ( $R_{VPP}$ ).	

2.5 I/O Circuits of Pins



## CHAPTER 3 CPU FUNCTIONS

The CPU of the V854 is based on the RISC architecture and executes most instructions in one clock cycle by using a 5-stage pipeline.

### 3.1 Features

- Minimum instruction cycle: 30 ns (at internal 33-MHz operation)
- Address space: 16 Mbytes linear
- General registers: Thirty-two 32-bit registers
- Internal 32-bit architecture
- Five-stage pipeline control
- Multiplication/division instructions
- Saturated operation instructions
- Single-clock 32-bit shift instruction
- Long/short instruction format
- Four types of bit manipulation instructions
  - Set
  - Clear
  - Not
  - Test

### 3.2 CPU Register Set

The registers of the V854 can be classified into two categories: a general-purpose program register set and a dedicated system register set. All the registers are 32 bits wide. For details, refer to **V850 Family User's Manual Architecture**.

#### Program register set

31		0
r0	Zero Register	
r1	Reserved for Address Generation	
r2	Interrupt Stack Pointer	
r3	Stack Pointer (SP)	
r4	Global Pointer (GP)	
r5	Text Pointer (TP)	
r6		
r7		
r8		
r9		
r10		
r11		
r12		
r13		
r14		
r15		
r16		
r17		
r18		
r19		
r20		
r21		
r22		
r23		
r24		
r25		
r26		
r27		
r28		
r29		
r30	Element Pointer (EP)	
r31	Link Pointer (LP)	
31		0
PC	Program Counter	

#### System register set

31		0
EIPC	Exception/Interrupt PC	
EIPSW	Exception/Interrupt PSW	
31		0
FEPC	Fatal Error PC	
FEPSW	Fatal Error PSW	
31		0
ECR	Exception Cause Register	
31		0
PSW	Program Status Word	

**3.2.1 Program register set**

The program register set includes general registers and a program counter.

**(1) General registers**

Thirty-two general registers, r0 to r31, are available. Any of these registers can be used as a data variable or address variable.

However, r0 and r30 are implicitly used by instructions, and care must be exercised when using these registers. Also, r1 to r5 and r31 are implicitly used by the assembler and C compiler. Therefore, before using these registers, their contents must be saved so that they are not lost. The contents must be restored to the registers after the registers have been used.

**Table 3-1. Program Registers**

Name	Usage	Operation
r0	Zero register	Always holds 0
r1	Assembler-reserved register	Working register for generating immediate
r2	Interrupt stack pointer	Stack pointer for interrupt handler
r3	Stack pointer	Used to generate stack frame when function is called
r4	Global pointer	Used to access global variable in data area
r5	Text pointer	Register to indicate the start of the text area <sup>Note</sup>
r6 to r29	–	Address/data variable registers
r30	Element pointer	Base pointer register when memory is accessed
r31	Link pointer	Used by compiler when calling function
PC	Program counter	Holds instruction address during program execution

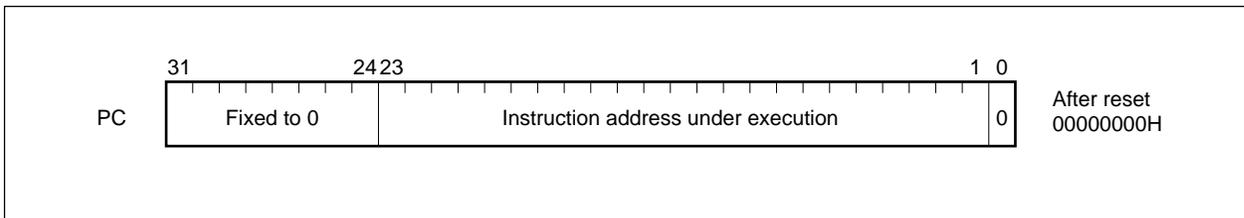
**Note** Area in which program code is mapped.

**(2) Program counter**

This register holds the address of the instruction under execution. The lower 24 bits of this register are valid, and bits 31 to 24 are fixed to 0. If a carry occurs from bit 23 to 24, it is ignored.

Bit 0 is fixed to 0, and branching to an odd address cannot be performed.

**Figure 3-1. Program Counter (PC)**



3.2.2 System register set

System registers control the status of the CPU and hold interrupt information.

Table 3-2. System Register Numbers

No.	System Register Name	Usage	Operation
0	EIPC	Status saving registers during interrupt	These registers save the PC and PSW when an exception or interrupt occurs. Because only one set of these registers is available, their contents must be saved when multiple interrupts are enabled.
1	EIPSW		
2	FEPC	Status saving registers for NMI	These registers save PC and PSW when NMI occurs.
3	FEPSW		
4	ECR	Interrupt source register	If exception, maskable interrupt, or NMI occurs, this register will contain information referencing the interrupt source. The high-order 16 bits of this register are called FECC, to which exception code of NMI is set. The low-order 16 bits are called EICC, to which exception code of exception/interrupt is set (refer to <b>Figure 3-2</b> ).
5	PSW	Program status word	Program status word is collection flags that indicate program status (instruction execution result) and CPU status (refer to <b>Figure 3-3</b> ).
6 to 31	Reserved		

To read/write these system registers, specify a system register number indicated by the system register load/store instruction (LDSR or STSR instruction).

Figure 3-2. Interrupt Source Register (ECR)

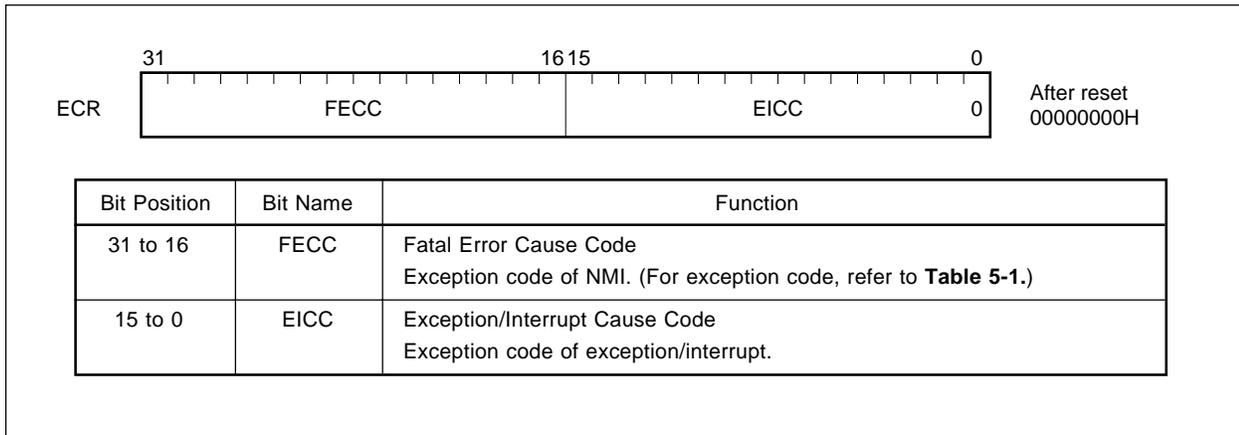
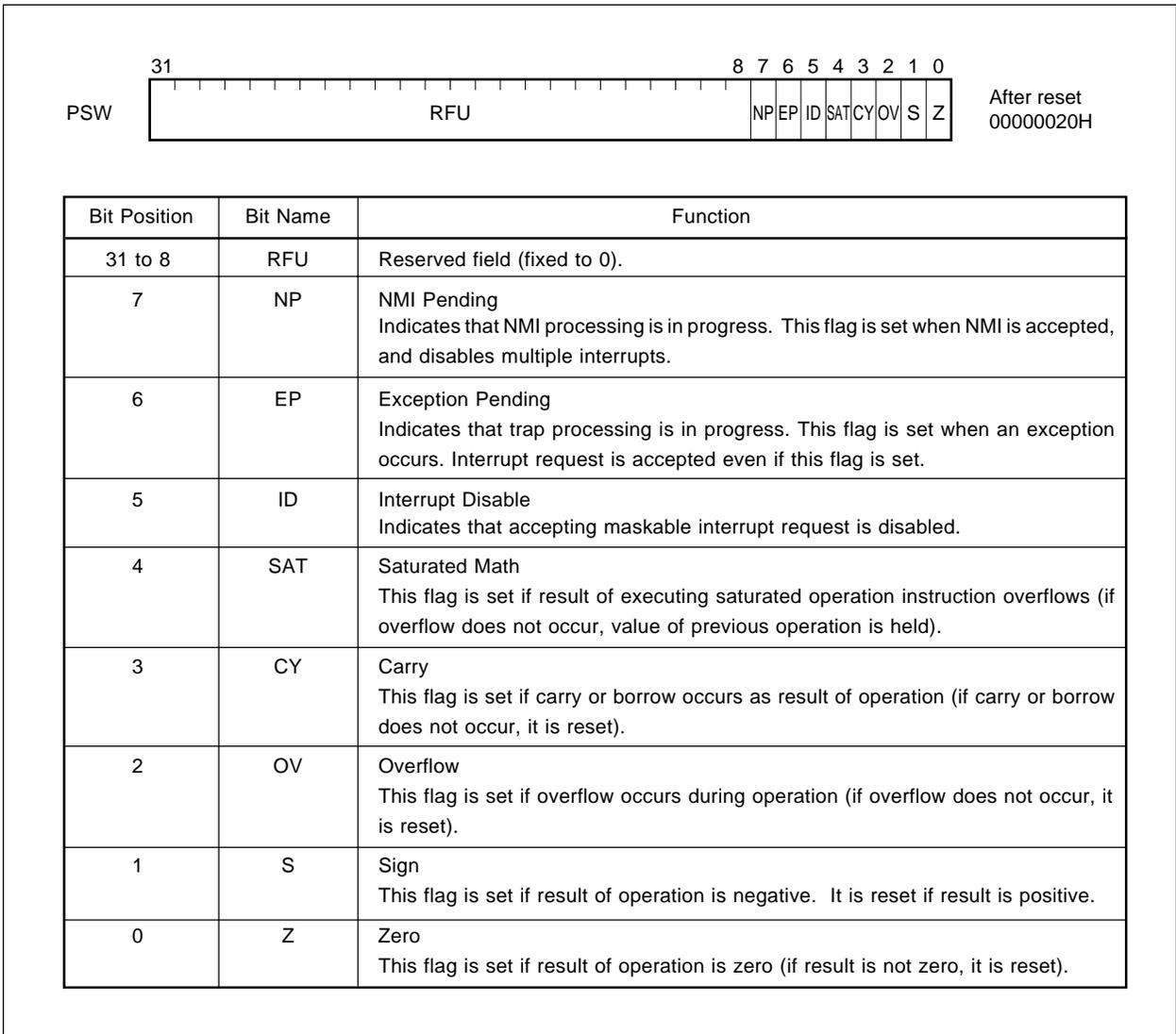


Figure 3-3. Program Status Word (PSW)



★

### 3.3 Operation Modes

#### 3.3.1 Operation modes

The V854 has the following operations modes. These modes are selected by the MODE pin (n = 0 to 2).

##### (1) Normal operation modes

###### (a) Single-chip mode ( $\mu$ PD703008, 70F3008, 703008Y, 70F3008Y only)

After the system has been released from the reset status, the pins related to the bus interface are set for port mode, execution branches to the reset entry address of the internal ROM, and instruction processing is started. However, access to external memory and peripheral devices can be enabled by setting in the external memory expansion mode register by instruction (MM: refer to **3.4.6 (1)**).

CLKOUT signal output is disabled when reset in the single-chip mode 1. However, it is input in the single-chip mode 2.

###### (b) ROM-less mode

After the system has been released from the reset status, the pins related to bus interface are set for control mode, execution branches to the reset address of external memory, and instruction processing is started. Instruction fetch and data access to internal ROM are disabled.

$\overline{UBEN}$ ,  $\overline{LBEN}$ ,  $R/\overline{W}$ , and  $\overline{DSTB}$  signal are output when reset in the ROM-less mode.  $\overline{UBEN}$ ,  $\overline{WRL}$ ,  $\overline{WRH}$ , and  $\overline{RD}$  signal are output in the ROM-less mode 2.

##### (2) Flash memory programming mode ( $\mu$ PD70F3008, 70F3008Y only)

- ★ This mode is provided only to on-chip flash memory model. If flash memory write voltage (7.8 V) is input to  $V_{PP}$  pin, the program operation to internal flash memory by flash writer is possible.

##### ★ 3.3.2 Specifying operation mode

The operation mode of the V854 is specified depending on the status of the MODE pin. Set these pins in the application system (n = 0 to 2).

If the setting is changed during operation, the operation is not guaranteed.

###### (a) $\mu$ PD703006

MODE2	MODE1	MODE0	Operation Mode	
0	0	1	Normal operation mode	ROM-less mode 1
0	0	1		ROM-less mode 2
Other than above			Setting prohibited	

###### (b) $\mu$ PD703008, 703008Y

MODE2	MODE1	MODE0	Operation Mode	
0	0	0	Normal operation mode	ROM-less mode 1
0	0	1		ROM-less mode 2
0	1	0		Single-chip mode 1
0	1	1		Single-chip mode 2
Other than above			Setting prohibited	

(c)  $\mu$ PD70F3008, 70F3008Y

Pin Status				Operation Mode	
V <sub>PP</sub>	MODE2	MODE1	MODE0		
0 V	0	0	0	Normal operation mode	ROM-less mode 1
0 V	0	0	1		ROM-less mode 2
0 V	0	1	0		Single-chip mode 1
0 V	0	1	1		Single-chip mode 2
7.8 V	1	1	1	Flash memory programming mode	
Other than above				Setting prohibited	

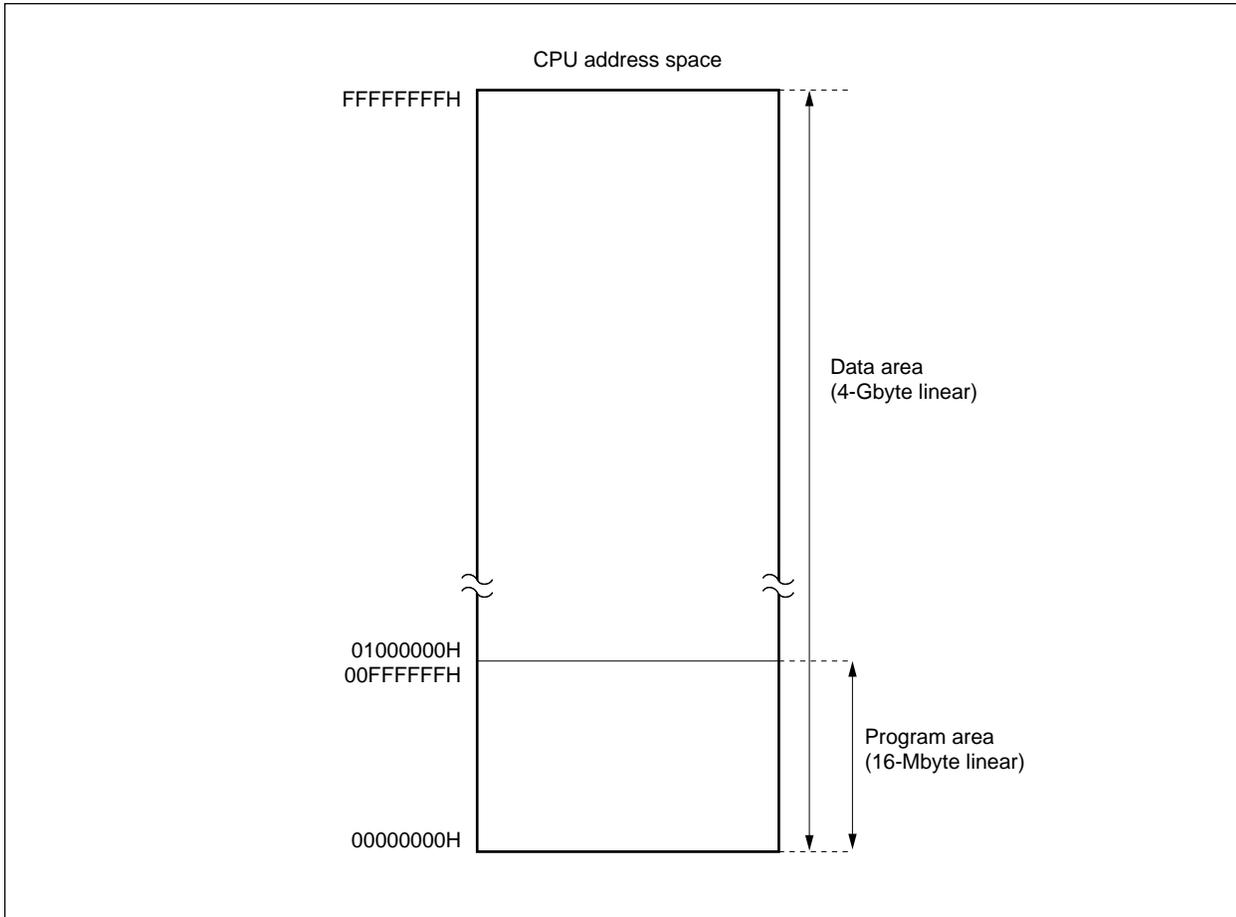
### 3.4 Address Space

#### 3.4.1 CPU address space

The CPU of the V854 is of 32-bit architecture and supports up to 4 Gbytes of linear address space (data space) during operand addressing (data access). When referencing instruction addresses, a linear address space (program space) of up to 16 Mbytes is supported.

Figure 3-4 shows the CPU address space.

Figure 3-4. CPU Address Space

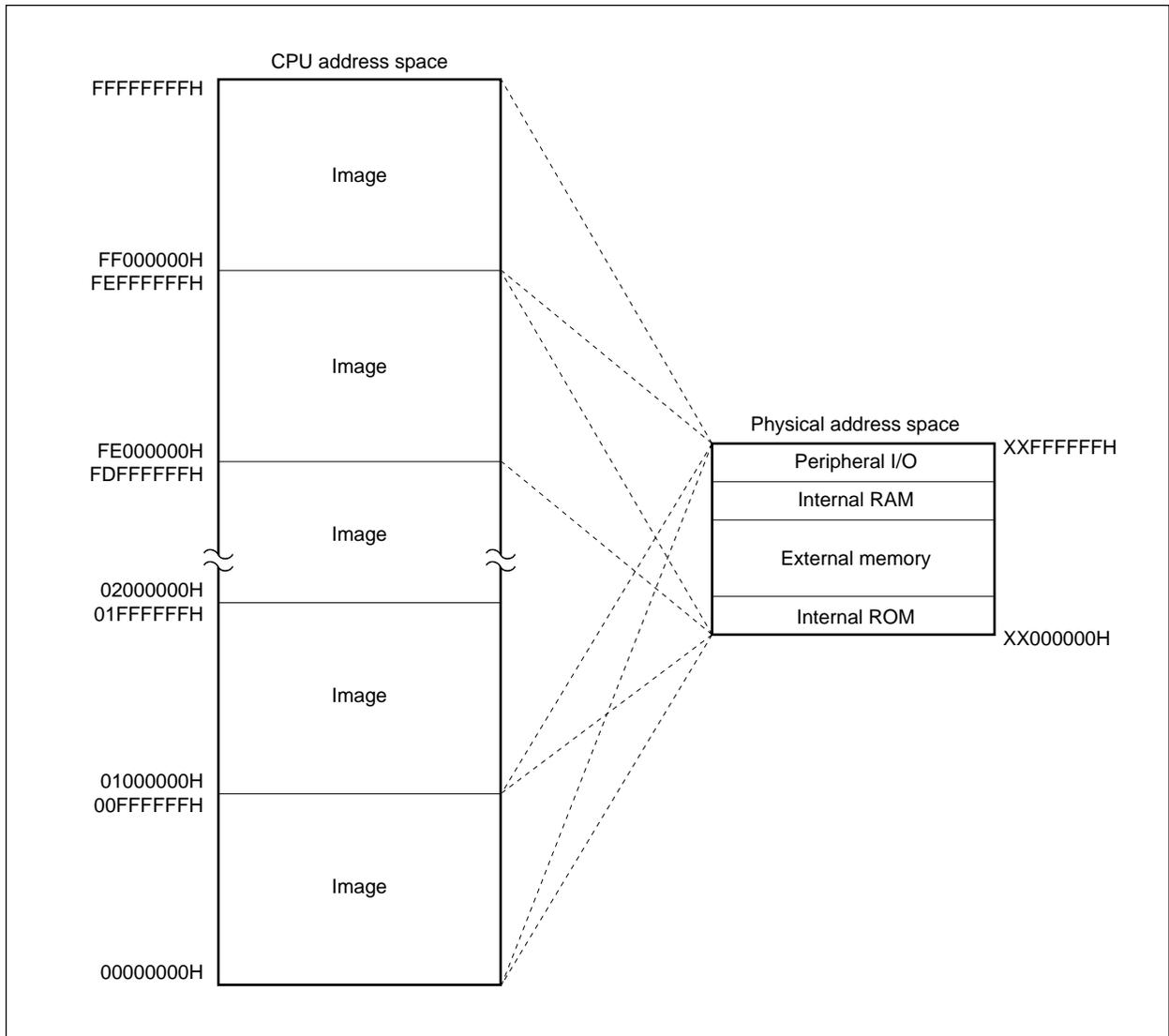


### 3.4.2 Image (Virtual Address Space)

The core CPU supports 4 Gbytes of “virtual” addressing space, or 256 memory blocks, each containing 16-Mbyte memory locations. In actuality, the same 16-Mbyte block is accessed regardless of the values of bits 31 to 24 of the CPU address. Figure 3-5 shows the image of the virtual addressing space.

Because the higher 8 bits of a 32-bit CPU address are ignored and the CPU address is only seen as a 24-bit external physical address, the physical location XX000000H is equally referenced by multiple address values 00000000H, 01000000H, 02000000H... through FF000000H.

Figure 3-5. Image on Address Space



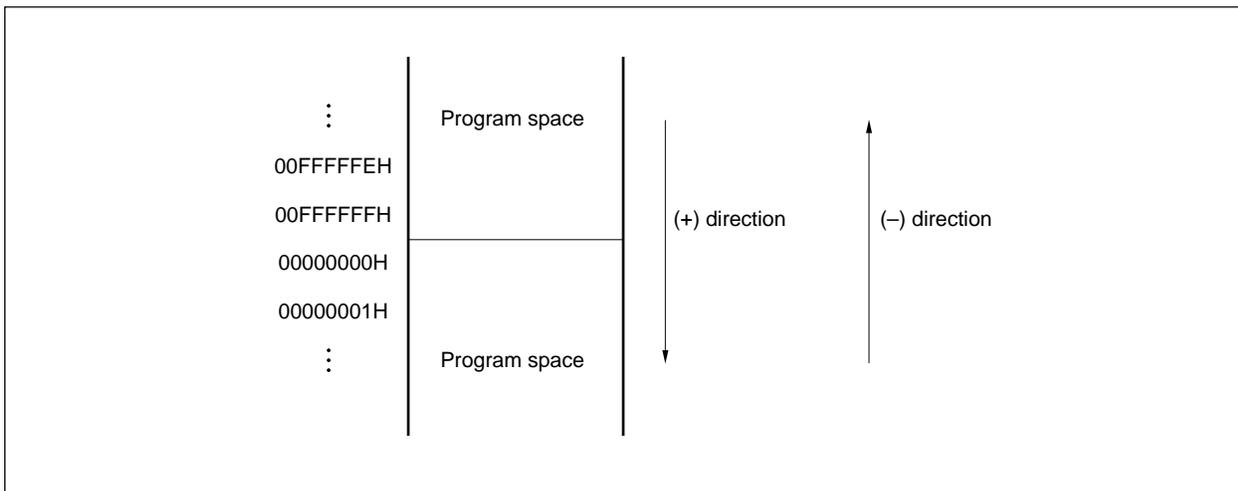
3.4.3 Wrap-around of CPU address space

(1) Program space

Of the 32 bits of the PC (program counter), the higher 8 bits are set to “0”, and only the lower 24 bits are valid. Even if a carry or borrow occurs from bit 23 to bit 24 as a result of branch address calculation, the higher 8 bits ignore the carry or borrow and remain “0”.

Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address 00FFFFFFH are contiguous addresses. The above-mentioned state in which the lower-limit and the upper-limit addresses of the memory space are contiguous addresses is called wraparound.

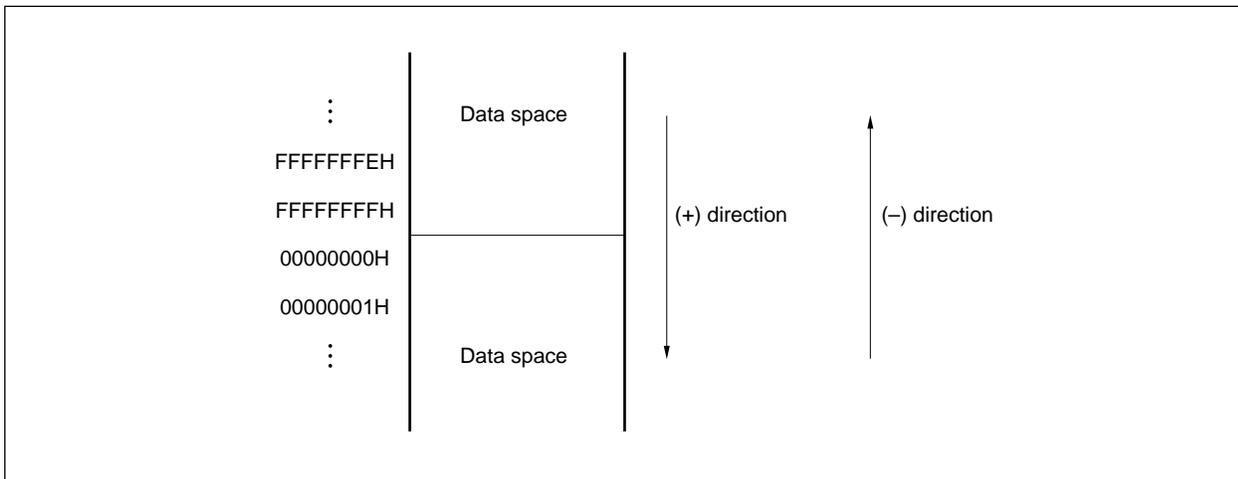
**Caution** No instruction can be fetched from the 4-Kbyte area of 00FFF000H to 00FFFFFFH because this area is defined as peripheral I/O area. Therefore, do not execute any branch operation instructions in which the destination address will reside in any part of this area.



(2) Data space

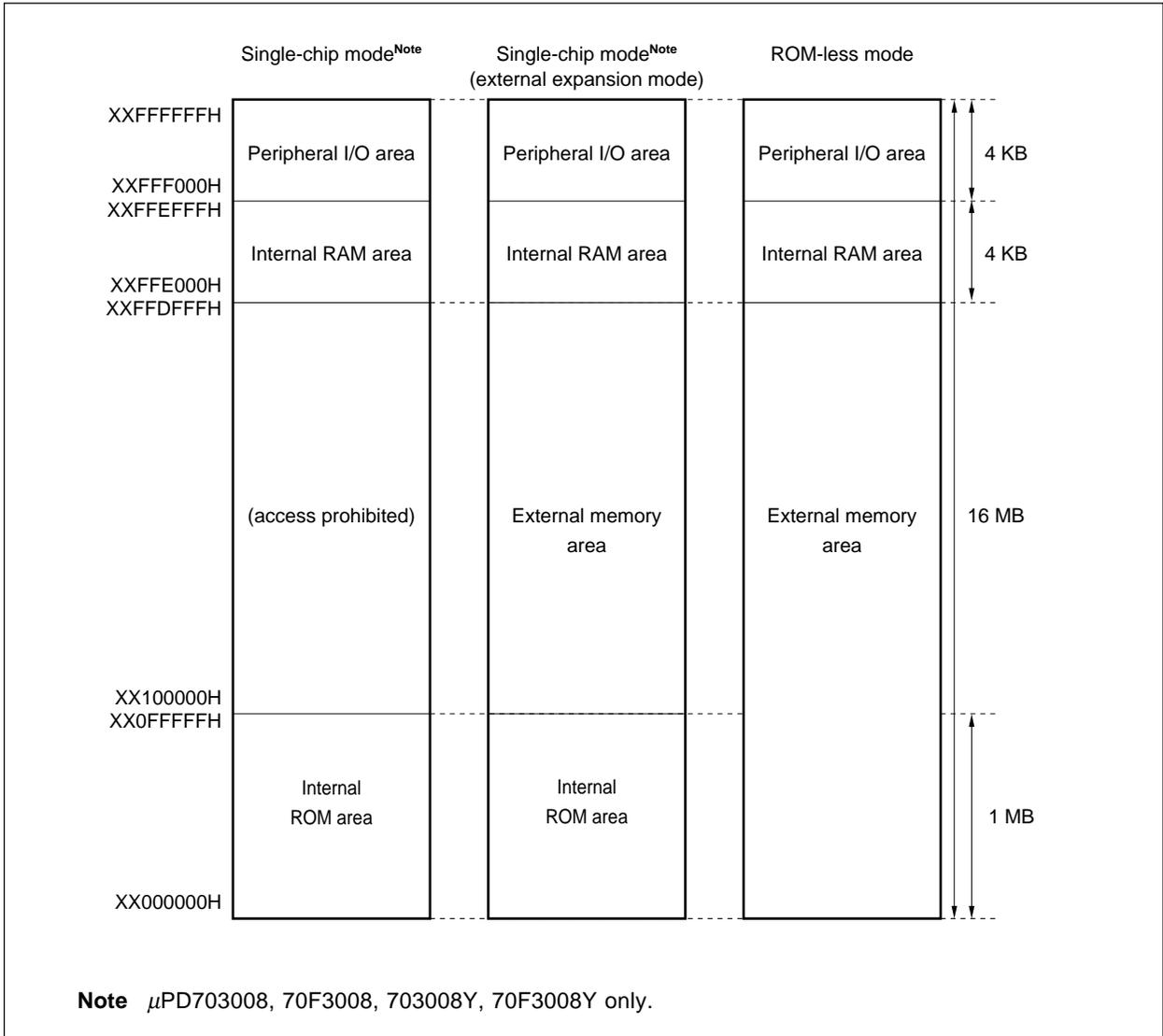
The result of operand address calculation that exceeds 32 bits is ignored.

Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address FFFFFFFFH are contiguous addresses, and the data space is wrapped around at the boundary of these addresses.



3.4.4 Memory map

The V854 reserves areas as shown below. Each mode is specified by using the MODE<sub>n</sub> pin at reset (n = 0 to 2).



3.4.5 Area

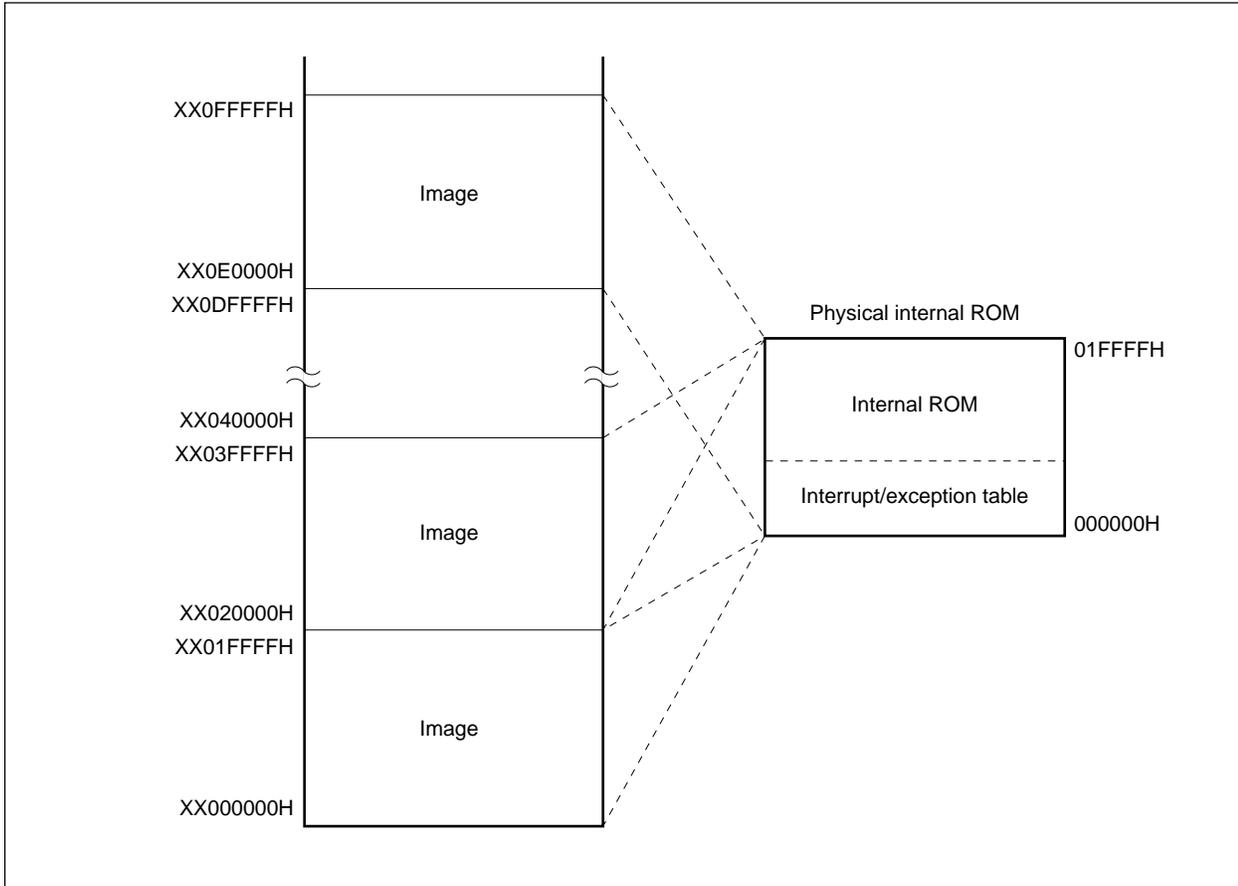
(1) Internal ROM area

A 1-Mbyte area corresponding to addresses 000000H to 0FFFFFFH is reserved for the internal ROM area. The V854 is provided with physical internal ROM as follows:

**Caution** Internal ROM products are  $\mu$ PD703008, 70F3008, 703008Y and 70F3008Y only.

- Physical internal ROM: 000000H to 01FFFFH (128 Kbytes)

The image of 000000H to 01FFFFH is seen in the rest of the area (020000H to 0FFFFFFH).



**Interrupt/exception table**

The V854 increases the interrupt response speed by assigning destination addresses corresponding to interrupts/exceptions.

The collection of these destination addresses is called an interrupt/exception table, which is located in the internal ROM area. When an interrupt/exception request is granted, execution jumps to the corresponding destination address, and the program written at that memory address is executed. Table 3-3 shows the sources of interrupts/exceptions, and the corresponding addresses.

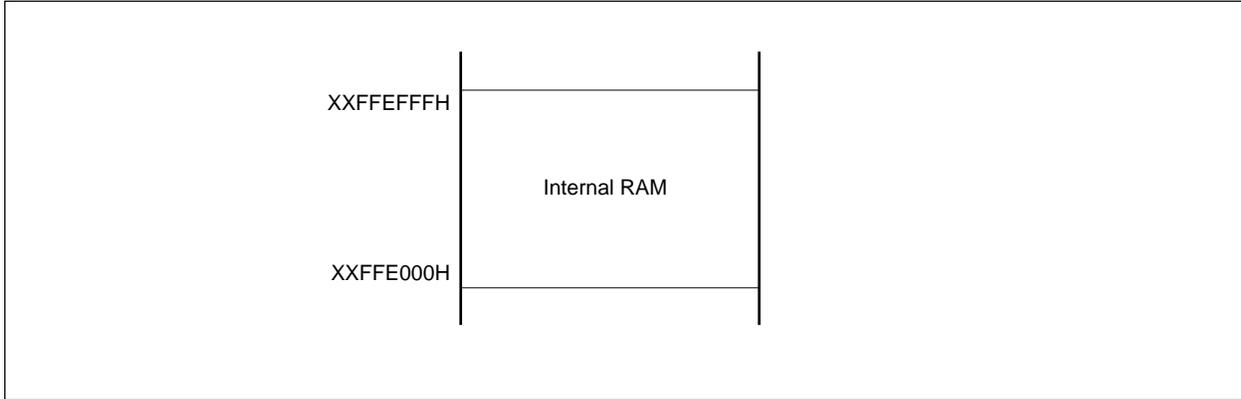
**Table 3-3. Interrupt/Exception Table**

Start Address of Interrupt/Exception Table	Interrupt/Exception Source
00000000H	RESET
00000010H	NMI
00000040H	TRAP0n (n = 0 to FH)
00000050H	TRAP1n (n = 0 to FH)
00000060H	ILGOP
00000080H	INTOV0/INTP04/INTP05
00000090H	INTOV1/INTP14
000000A0H	INTCC00/INTP00
000000B0H	INTCC01/INTP01
000000C0H	INTCC02/INTP02
000000D0H	INTCC03/INTP03
000000E0H	INTC10
000000F0H	INTC11
00000100H	INTCP12
00000110H	INTCP13
00000120H	INTCM10
00000130H	INTCM11
00000140H	INTCM20/INTP20
00000150H	INTCM21/INTP21
00000160H	INTCM22/INTP22
00000170H	INTCM23/INTP23
00000180H	INTCM24/INTP24
00000190H	INTCC3/INTP30
000001A0H	INTCSI0
000001B0H	INTCSI1
000001C0H	INTCSI2
000001D0H	INTCSI3
000001E0H	INIIC
000001F0H	INTSER
00000200H	INTSR
00000210H	INTST
00000220H	INTAD
00000230H	INTP50
00000240H	INTP51
00000250H	INTP52
00000260H	INTP53

**Caution** The internal ROM area becomes the external memory area in ROM-less mode or in the  $\mu$ PD703006. For normal operation after reset, keep the destination address for the reset routine in external memory address 0.

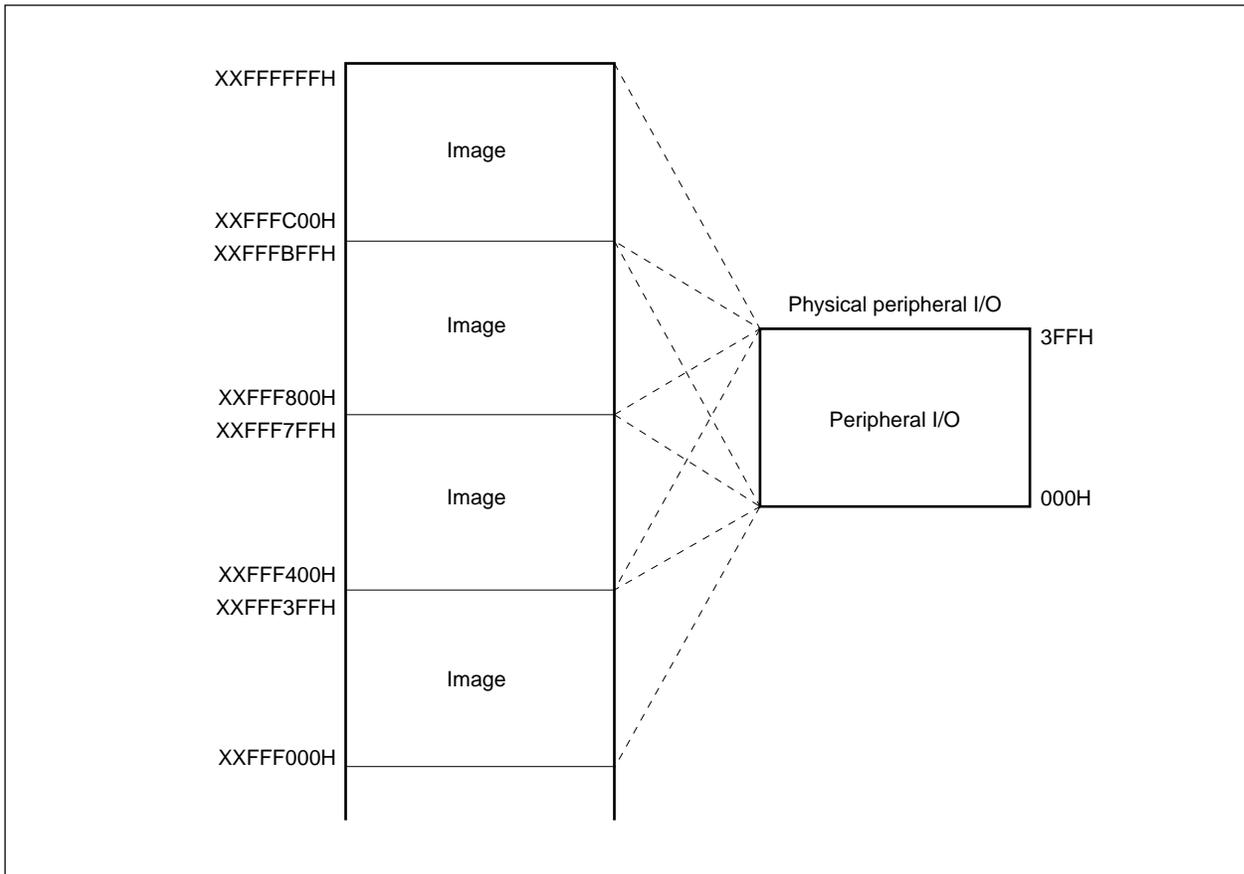
**(2) Internal RAM area**

The V854 is provided with 4 Kbytes of addresses FFE000H to FFEFFFH as a physical internal RAM area.



**(3) Peripheral I/O area**

A 4-Kbyte area of addresses FFF000H to FFFFFFFH is reserved as a peripheral I/O area. The V854 is provided with a 1-Kbyte area of addresses FFF000H to FFF3FFH as a physical peripheral I/O area, and the image of FFF000H to FFF3FFH can be seen on the rest of the area (FFF400H to FFFFFFFH).



Peripheral I/O registers associated with the operation mode specification and the state monitoring for the on-chip peripherals are all memory-mapped to the peripheral I/O area. Program fetches are not allowed in this area.

- Cautions**
1. The least significant bit of an address is not decoded since all registers reside on an even address. If an odd address ( $2n + 1$ ) in the peripheral I/O area is referenced, the register at the next lowest even address ( $2n$ ) will be accessed.
  2. If a register that can be accessed in byte units is accessed in half-word units, the higher 8 bits become undefined, if the access is a read operation. If a write access is made, only the data in the lower 8 bits is written to the register.
  3. If a register with  $n$  address that can be accessed only in halfword units is accessed with a word operation, the operation is replaced with two halfword operations. The first operation (lower 16 bits) accesses to the register with  $n$  address and the second operation (higher 16 bits) accesses to the register with  $n + 2$  address.
  4. If a register with  $n$  address that can be accessed in word units is accessed with a word operation, the operation is replaced with two halfword operations. The first operation (lower 16 bits) accesses to the register with  $n$  address and the second operation (higher 16 bits) accesses to the register with  $n + 2$  address.
  5. Addresses that are not defined as registers are reserved for future expansion. If these addresses are accessed, the operation is undefined and not guaranteed.

**(4) External memory area**

★ The  $\mu$ PD703008, 70F3008, 703008Y, and 70F3008Y can use an area of up to xx100000H to xxFFDFFFH as an external memory area in the single-chip mode.

The  $\mu$ PD703006, 703008, 70F3008, 703008Y, and 70F3008Y can use an area of up to xx000000H to xxFFDFFFH as an external memory area in the ROM-less mode.

In the external memory area, 64 K, 256 K, 1 M, 4 M, or 16 Mbytes of physical external memory can be allocated when the external expansion mode is specified. The same image as that of the physical external memory can be seen continuously on the external memory area, as shown in Figure 3-6, when the memory is not fully expanded (to 16 Mbytes).

The internal RAM area, peripheral I/O area, and internal ROM area in single-chip mode are not subject to external memory access.

**Figure 3-6. External Memory Area (when expanded to 64 K, 256 K, or 1 Mbytes)**

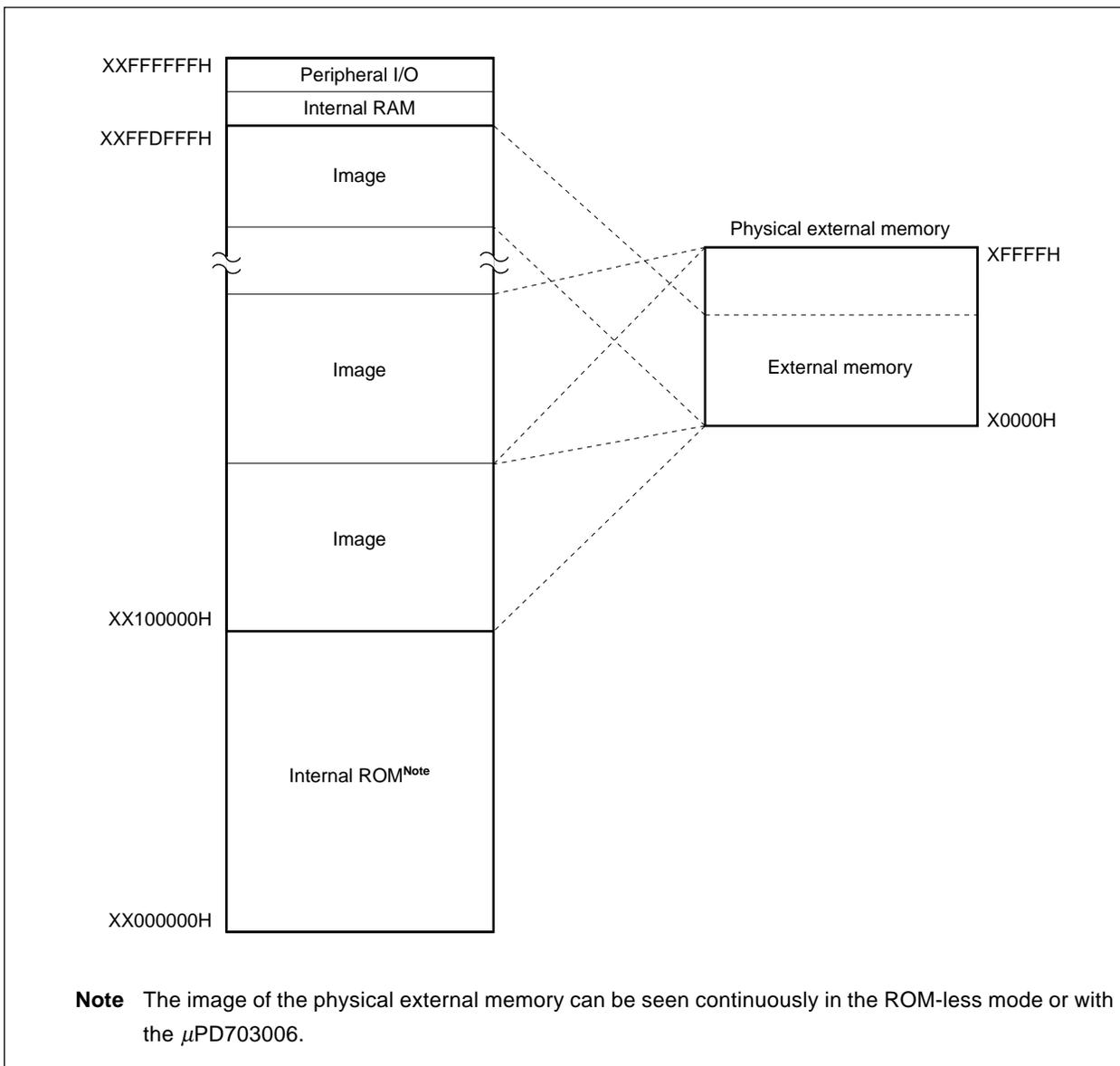
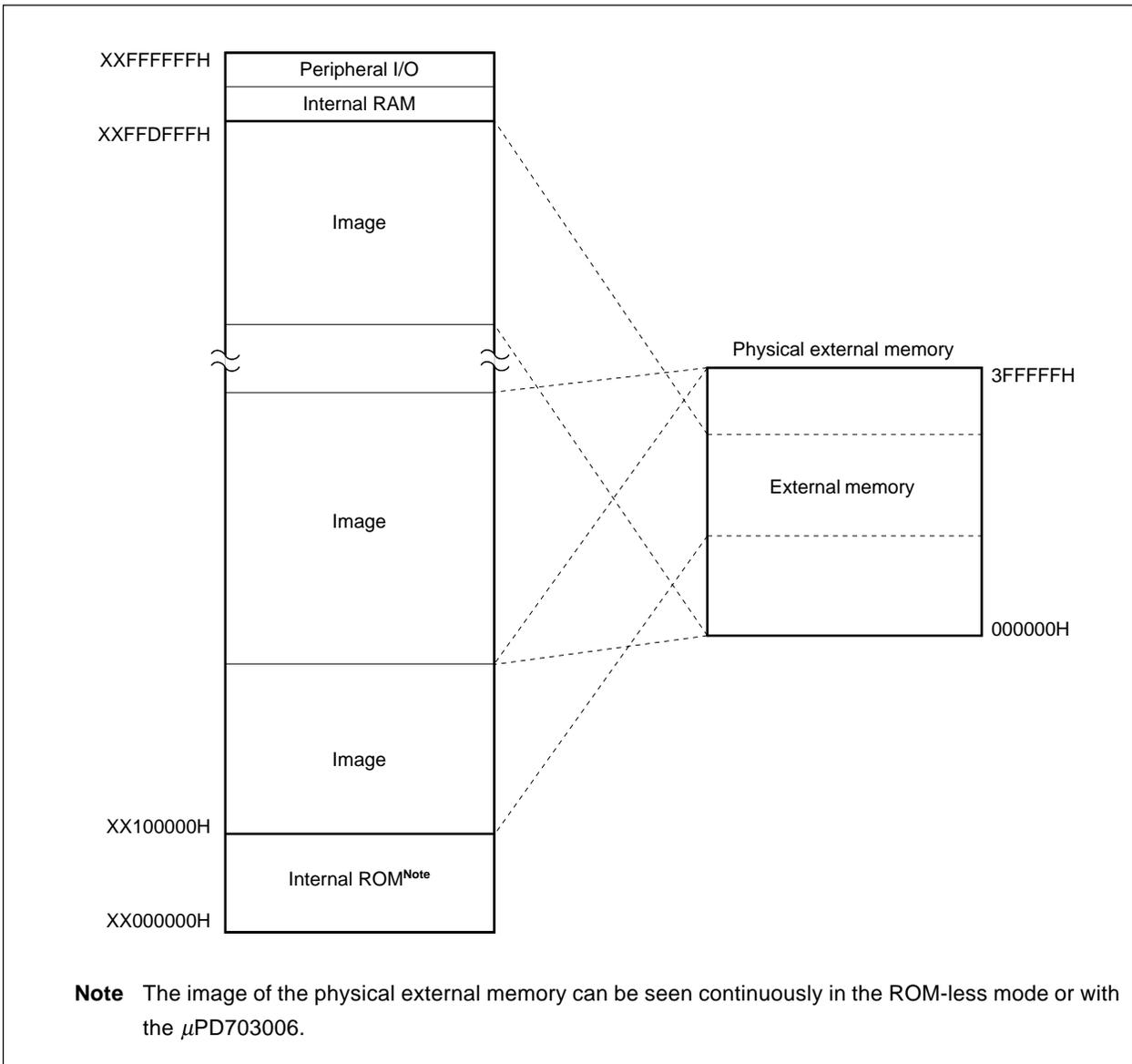
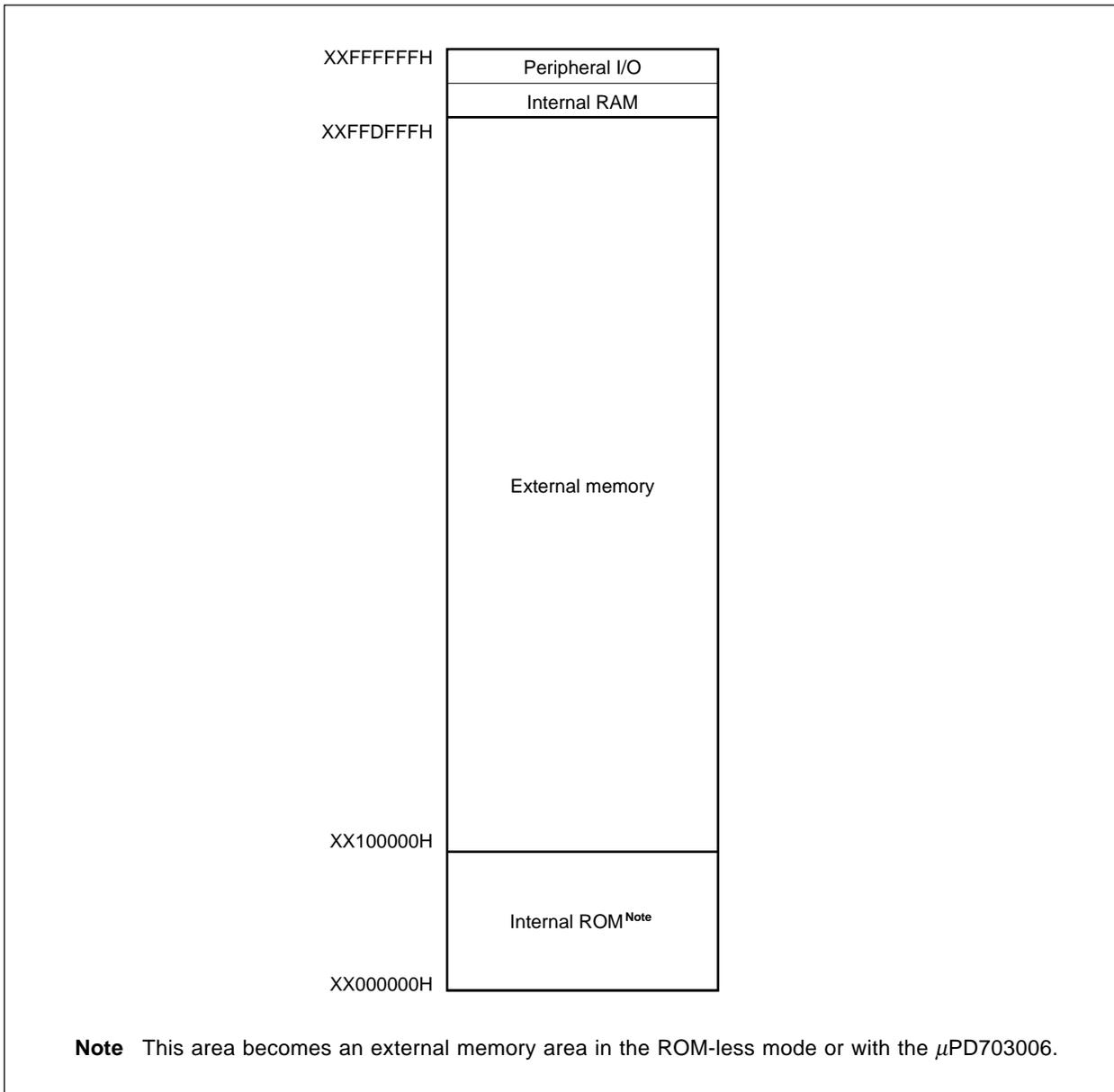


Figure 3-7. External Memory Area (when expanded to 4 Mbytes)



★

Figure 3-8. External Memory Area (when fully expanded)



### 3.4.6 External expansion mode

The V854 allows external devices to be connected to the external memory space by using the pins of ports 4, 5, 6, and 9. To connect an external device, the port pins must be set in the external expansion mode by using the MODEn pins and memory expansion mode register (MM). The MODEn pins specify the operation mode of the V854.

For specifying, refer to **3.3.2 Specifying operation mode**.

In ROM-less mode, the pins of port 4 to port 6 and P90 to P94 become the control mode during reset, thereby the external memory can be used.

In single-chip mode, the port/control mode alternate pins become the port mode, thereby the external memory cannot be used. When the external memory is used (external expansion mode), specify the MM register by the program (the memory area is set by the MM register).

**Remark** n = 0 to 2

**(1) Memory expansion mode register (MM)**

This register sets the mode of each pin of ports 4, 5, 6, and 9. In the external expansion mode, an external device can be connected to the external memory area of up to 16 Mbytes. However, the external device cannot be connected to the internal RAM area, peripheral I/O area, and internal ROM area in the single-chip mode (access is restricted to external locations 100000H through FFE00H).

The MM register can be read/written in 8- or 1-bit units. However, bits 4 to 7 are fixed to 0.



Bit Position	Bit Name	Function																																																
3	MM3	Memory Expansion Mode Specifies operation mode of P95 and P96 of port 9. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>MM3</th> <th>Operation Mode</th> <th>P95</th> <th>P96</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Port mode</td> <td colspan="2" style="text-align: center;">Port</td> </tr> <tr> <td style="text-align: center;">1</td> <td>External expansion mode</td> <td style="text-align: center;"><math>\overline{\text{HLDAK}}</math></td> <td style="text-align: center;"><math>\overline{\text{HLDRQ}}</math></td> </tr> </tbody> </table>	MM3	Operation Mode	P95	P96	0	Port mode	Port		1	External expansion mode	$\overline{\text{HLDAK}}$	$\overline{\text{HLDRQ}}$																																				
MM3	Operation Mode	P95	P96																																															
0	Port mode	Port																																																
1	External expansion mode	$\overline{\text{HLDAK}}$	$\overline{\text{HLDRQ}}$																																															
2 to 0	MM2 to MM0	Memory Expansion Mode Specifies operation mode of ports 4, 5, 6, and 9 (P90 to P94). <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>MM2</th> <th>MM1</th> <th>MM0</th> <th>Address Space</th> <th>Port 4</th> <th>Port 5</th> <th>Port 6</th> <th>Port 9 (P90 to P94)</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">-</td> <td colspan="4" style="text-align: center;">Port mode</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>64-KB expansion</td> <td rowspan="5" style="text-align: center;">AD0 to AD7</td> <td rowspan="5" style="text-align: center;">AD8 to AD15</td> <td rowspan="5" style="text-align: center;">A16 A17 A18 A19 A20 A21 A22 A23</td> <td rowspan="5" style="text-align: center;"><math>\overline{\text{LBEN}}</math>, <math>\overline{\text{UBEN}}</math>, R/<math>\overline{\text{W}}</math>, <math>\overline{\text{DSTB}}</math>, ASTB, WRL, <math>\overline{\text{WRH}}</math>, RD</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>256-KB expansion</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>1-MB expansion</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>4-MB expansion</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>16-MB expansion</td> </tr> <tr> <td colspan="4" style="text-align: center;">Others</td> <td colspan="4" style="text-align: center;">RFU (reserved)</td> </tr> </tbody> </table>	MM2	MM1	MM0	Address Space	Port 4	Port 5	Port 6	Port 9 (P90 to P94)	0	0	0	-	Port mode				0	1	1	64-KB expansion	AD0 to AD7	AD8 to AD15	A16 A17 A18 A19 A20 A21 A22 A23	$\overline{\text{LBEN}}$ , $\overline{\text{UBEN}}$ , R/ $\overline{\text{W}}$ , $\overline{\text{DSTB}}$ , ASTB, WRL, $\overline{\text{WRH}}$ , RD	1	0	0	256-KB expansion	1	0	1	1-MB expansion	1	1	0	4-MB expansion	1	1	1	16-MB expansion	Others				RFU (reserved)			
MM2	MM1	MM0	Address Space	Port 4	Port 5	Port 6	Port 9 (P90 to P94)																																											
0	0	0	-	Port mode																																														
0	1	1	64-KB expansion	AD0 to AD7	AD8 to AD15	A16 A17 A18 A19 A20 A21 A22 A23	$\overline{\text{LBEN}}$ , $\overline{\text{UBEN}}$ , R/ $\overline{\text{W}}$ , $\overline{\text{DSTB}}$ , ASTB, WRL, $\overline{\text{WRH}}$ , RD																																											
1	0	0	256-KB expansion																																															
1	0	1	1-MB expansion																																															
1	1	0	4-MB expansion																																															
1	1	1	16-MB expansion																																															
Others				RFU (reserved)																																														

**Remark** For the details of the operation of each port pin, refer to **2.3 Pin Function**.

**3.4.7 Recommended use of address space**

The architecture of the V854 requires that a register that serves as a pointer be secured for address generation in operand data accessing for data space. The address in this pointer register  $\pm 32$  Kbytes can be accessed directly from instruction. However, general register used as a pointer register is limited. Therefore, by minimizing the deterioration of address calculation performance when changing the pointer value, the number of usable general registers for handling variables is maximized, and the program size can be saved because instructions for calculating pointer addresses are not required.

To enhance the efficiency of using the pointer in connection with the memory map of the V854, the following points are recommended:

**(1) Program space**

Of the 32 bits of the PC (program counter), the higher 8 bits are fixed to "0", and only the lower 24 bits are valid. Therefore, a contiguous 16-Mbyte space, starting from address 00000000H, unconditionally corresponds to the memory map of the program space.

**(2) Data space**

For the efficient use of resources to be performed through the wrap-around feature of the data space, the continuous 8-Mbyte address spaces 00000000H to 007FFFFFFH and FF800000H to FFFFFFFFH of the 4-Gbyte CPU are used as the data space. With the V854, 16-Mbyte physical address space is seen as 256 images in the 4-Gbyte CPU address space. The highest bit (bit 23) of this 24-bit address is assigned as address sign-extended to 32 bits.

**Application of wrap-around**

For example, when R = r0 (zero register) is specified for the LD/ST disp 16 [R] instruction, an addressing range of 00000000H  $\pm$  32 Kbytes can be referenced with the sign-extended, 16-bit displacement value. By mapping the external memory in the 24-Kbyte area in the figure, all resources including on-chip hardware can be accessed with one pointer.

The zero register (r0) is a register set to 0 by the hardware, and eliminates the need for additional registers for the pointer.

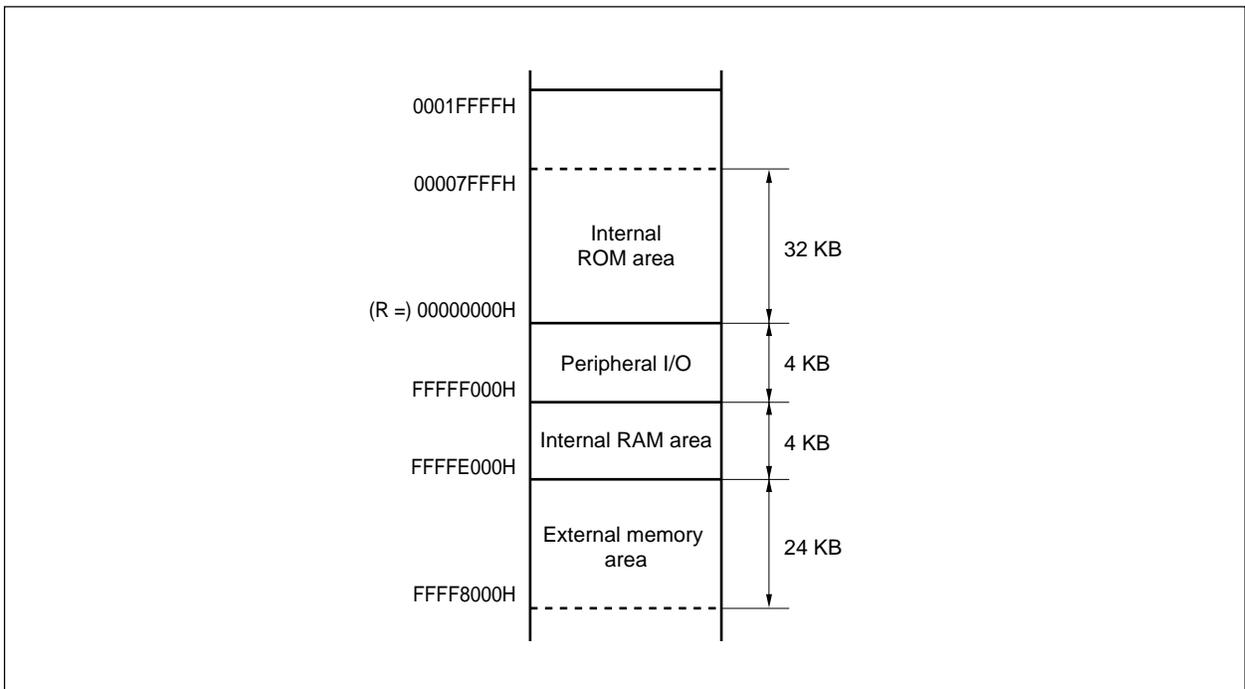
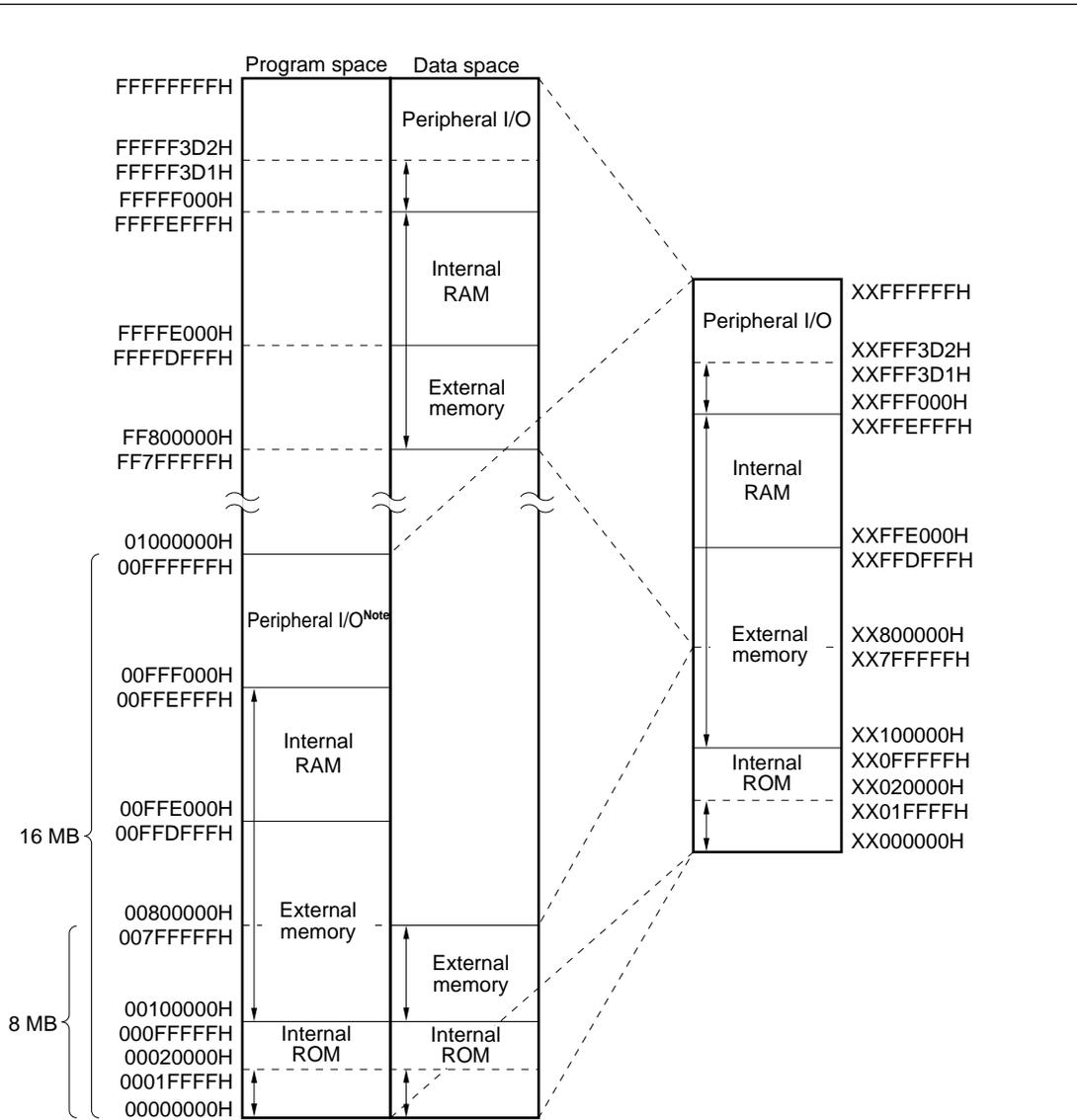


Figure 3-9. Recommended Memory Map



**Note** This area cannot be used as a program area.

- Remarks**
1. The vertical arrows ( $\updownarrow$ ) indicate the recommended area.
  2. In this figure, the  $\mu$ PD703008 is set in the single-chip mode and the recommended memory map when the external expanded memory mode being used is shown.

## 3.4.8 Peripheral I/O registers

(1/6)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 bit	8 bits	16 bits	32bits	
FFFFFF00H	Port 0	P0	R/W	○	○			Undefined
FFFFFF02H	Port 1	P1		○	○			
FFFFFF04H	Port 2	P2		○	○			
FFFFFF06H	Port 3	P3		○	○			
FFFFFF08H	Port 4	P4		○	○			
FFFFFF0AH	Port 5	P5		○	○			
FFFFFF0CH	Port 6	P6		○	○			
FFFFFF0EH	Port 7	P7	R	○	○			
FFFFFF10H	Port 8	P8		○	○			
FFFFFF12H	Port 9	P9	R/W	○	○			
FFFFFF14H	Port 10	P10		○	○			
FFFFFF16H	Port 11	P11		○	○			
FFFFFF18H	Port 12	P12		○	○			
FFFFFF1AH	Port 13	P13		○	○			
FFFFFF1CH	Port 14	P14		○	○			
FFFFFF20H	Port 0 mode register	PM0			○	○		
FFFFFF22H	Port 1 mode register	PM1		○	○			
FFFFFF24H	Port 2 mode register	PM2		○	○			
FFFFFF26H	Port 3 mode register	PM3		○	○			
FFFFFF28H	Port 4 mode register	PM4		○	○			
FFFFFF2AH	Port 5 mode register	PM5		○	○			
FFFFFF2CH	Port 6 mode register	PM6		○	○			
FFFFFF32H	Port 9 mode register	PM9		○	○			
FFFFFF34H	Port 10 mode register	PM10		○	○			
FFFFFF36H	Port 11 mode register	PM11		○	○			
FFFFFF38H	Port 12 mode register	PM12		○	○			
FFFFFF3AH	Port 13 mode register	PM13		○	○			
FFFFFF3CH	Port 14 mode register	PM14		○	○			
FFFFFF40H	Port 0 mode control register	PMC0		○	○			00H
FFFFFF42H	Port 1 mode control register	PMC1		○	○			
FFFFFF44H	Port 2 mode control register	PMC2		○	○			01H
FFFFFF46H	Port 3 mode control register	PMC3		○	○			00H
FFFFFF4CH	Memory expansion mode register	MM		○	○			00H/07H
FFFFFF54H	Port 10 mode control register	PMC10		○	○			00H
FFFFFF56H	Port 11 mode control register	PMC11		○	○			
FFFFFF58H	Port 12 mode control register	PMC12		○	○			

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 bit	8 bits	16 bits	32bits		
FFFFF05AH	Port 13 mode control register	PMC13	R/W	○	○			00H	
FFFFF060H	Data wait control register	DWC				○		FFFFH	
FFFFF062H	Bus cycle control register	BCC				○		AAAAH	
FFFFF064H	System control register	SYC		○	○			00H/1H	
FFFFF070H	Power save control register	PSC		○	○			00H/C0H	
FFFFF072H	Clock control register	CKC		○	○			00H	
FFFFF078H	System status register	SYS		○	○			0000000XB	
FFFFF084H	Baud rate generator compare register 0	BRGC0		○	○			Undefined	
FFFFF086H	Baud rate generator prescaler mode register 0	BPRM0		○	○			00H	
FFFFF088H	Clocked serial interface mode register 0	CSIM0		○	○				
FFFFF08AH	Serial I/O shift register 0	SIO0		○	○			Undefined	
FFFFF094H	Baud rate generator compare register 1	BRGC1		○	○				
FFFFF096H	Baud rate generator prescaler mode register 1	BPRM1		○	○			00H	
FFFFF098H	Clocked serial interface mode register 1	CSIM1		○	○				
FFFFF09AH	Serial I/O shift register 1	SIO1		○	○			Undefined	
FFFFF0A4H	Baud rate generator compare register 2	BRGC2		○	○				
FFFFF0A6H	Baud rate generator prescaler mode register 2	BPRM2		○	○			00H	
FFFFF0A8H	Clocked serial interface mode register 2	CSIM2		○	○				
FFFFF0AAH	Serial I/O shift register 2	SIO2		○	○			Undefined	
FFFFF0B4H	Baud rate generator register 3	BRGC3		○	○				
FFFFF0B6H	Baud rate generator prescaler mode register	BPRM3		○	○			00H	
FFFFF0B8H	Clocked serial interface mode register 3	CSIM3		○	○				
FFFFF0BAH	Serial I/O shift register 3	SIO3		○	○			Undefined	
FFFFF0C0H	Asynchronous serial interface mode register 0	ASIM0		○	○			80H	
FFFFF0C2H	Asynchronous serial interface mode register 1	ASIM1		○	○			00H	
FFFFF0C4H	Asynchronous serial interface status register	ASIS		R	○	○			
FFFFF0C8H	Receive buffer (9 bits)	RXB					○		Undefined
FFFFF0CAH	Receive buffer L (lower 8 bits)	RXBL			○	○			
FFFFF0CCH	Transmit shift register (9 bits)	TXS		W			○		
FFFFF0CEH	Transmit shift register L (lower 8 bits)	TXSL					○		
FFFFF0E0H	IIC control register	IICC		R/W	○	○			00H
FFFFF0E2H	IIC status register	IICS		R	○	○			
FFFFF0E4H	IIC clock selection register	IICCL		R/W	○	○			
FFFFF0E6H	IIC shift register	IIC			○	○			
FFFFF0E8H	Slave address register	SVA	○		○				
FFFFF100H	Interrupt control register	OVIC0	○		○			47H	
FFFFF102H	Interrupt control register	OVIC1	○	○					
FFFFF104H	Interrupt control register	CC0IC0	○	○					

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 bit	8 bits	16 bits	32bits	
FFFFF106H	Interrupt control register	CC0IC1	R/W	○	○			47H
FFFFF108H	Interrupt control register	CC0IC2		○	○			
FFFFF10AH	Interrupt control register	CC0IC3		○	○			
FFFFF10CH	Interrupt control register	P1IC0		○	○			
FFFFF10EH	Interrupt control register	P1IC1		○	○			
FFFFF110H	Interrupt control register	P1IC2		○	○			
FFFFF112H	Interrupt control register	P1IC3		○	○			
FFFFF114H	Interrupt control register	CM1IC0		○	○			
FFFFF116H	Interrupt control register	CM1IC1		○	○			
FFFFF118H	Interrupt control register	CM2IC0		○	○			
FFFFF11AH	Interrupt control register	CM2IC1		○	○			
FFFFF11CH	Interrupt control register	CM2IC2		○	○			
FFFFF11EH	Interrupt control register	CM2IC3		○	○			
FFFFF120H	Interrupt control register	CM2IC4		○	○			
FFFFF122H	Interrupt control register	CM3IC0		○	○			
FFFFF124H	Interrupt control register	CSIC0		○	○			
FFFFF126H	Interrupt control register	CSIC1		○	○			
FFFFF128H	Interrupt control register	CSIC2		○	○			
FFFFF12AH	Interrupt control register	CSIC3		○	○			
FFFFF12CH	Interrupt control register	IIIC0		○	○			
FFFFF12EH	Interrupt control register	SEIC0		○	○			
FFFFF130H	Interrupt control register	SRIC0		○	○			
FFFFF132H	Interrupt control register	STIC0		○	○			
FFFFF134H	Interrupt control register	ADIC0		○	○			
FFFFF136H	Interrupt control register	P51C0		○	○			
FFFFF138H	Interrupt control register	P51C1		○	○			
FFFFF13AH	Interrupt control register	P51C2		○	○			
FFFFF13CH	Interrupt control register	P51C3		○	○			
FFFFF166H	In-service priority register	ISPR	R	○	○			00H
FFFFF170H	Command register	PRCMD	W		○			Undefined
FFFFF180H	External interrupt mode register 0	INTM0	R/W	○	○			00H
FFFFF182H	External interrupt mode register 1	INTM1		○	○			
FFFFF184H	External interrupt mode register 2	INTM2		○	○			
FFFFF18AH	External interrupt mode register 5	INTM5		○	○			
FFFFF18CH	External interrupt mode register 6	INTM6		○	○			
FFFFF18EH	External interrupt mode register 7	INTM7		○	○			
FFFFF1B0H	Event divide control register 0	EDVC0		○	○			
FFFFF1B2H	Event divide control register 1	EDVC1	○	○				

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 bit	8 bits	16 bits	32bits		
FFFFF1B4H	Event divide control register 2	EDVC2	R/W	○	○			01H	
FFFFF1B6H	Event divide counter 0	EDV0	R	○	○			00H	
FFFFF1B8H	Event divide counter 1	EDV1		○	○				
FFFFF1BAH	Event divide counter 2	EDV2		○	○				
FFFFF1C0H	Event selection register	EVS	R/W	○	○			01H	
FFFFF230H	Timer overflow status register	TOVS		○	○				
FFFFF232H	Timer output control register 0	TOC0		○	○				
FFFFF234H	Timer output control register 1	TOC1		○	○				
FFFFF240H	Timer control register 00	TMC00		○	○				01H
FFFFF242H	Timer control register 01	TMC01		○	○				00H
FFFFF244H	Timer control register 02	TMC02		○	○				
FFFFF250H	Timer 0	TM0	R				○	00000000H	
FFFFF254H	Capture/compare register 00	CC00	R/W				○	Undefined	
FFFFF258H	Capture/compare register 01	CC01					○		
FFFFF25CH	Capture/compare register 02	CC02					○		
FFFFF260H	Capture/compare register 03	CC03					○		
FFFFF264H	Timer 0L	TM0L	R			○		0000H	
FFFFF266H	Capture/compare register 00L	CC00L	R/W				○	Undefined	
FFFFF268H	Capture/compare register 01L	CC01L					○		
FFFFF26AH	Capture/compare register 02L	CC02L					○		
FFFFF26CH	Capture/compare register 03L	CC03L					○		
FFFFF270H	Timer control register 1	TMC1		○	○			01H	
FFFFF274H	Timer 1	TM1	R				○	00000000H	
FFFFF278H	Compare register 10	CM10	R/W				○	Undefined	
FFFFF27CH	Compare register 11	CM11					○		
FFFFF280H	Capture register 10	CP10	R				○	0000H	
FFFFF284H	Capture register 11	CP11					○		
FFFFF288H	Capture register 12	CP12					○		
FFFFF28CH	Capture register 13	CP13					○		
FFFFF290H	Timer 1L	TM1L				○		0000H	
FFFFF292H	Compare register 10L	CM10L	R/W				○	Undefined	
FFFFF294H	Compare register 11L	CM11L					○		
FFFFF296H	Capture register 10L	CP10L	R				○	0000H	
FFFFF298H	Capture register 11L	CP11L					○		
FFFFF29AH	Capture register 12L	CP12L					○		
FFFFF29CH	Capture register 13L	CP13L					○		
FFFFF2A0H	Timer control register 20	TMC20	R/W	○	○			01H	
FFFFF2B0H	Timer 20	TM20	R				○	0000H	

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 bit	8 bits	16 bits	32bits		
FFFFF2B2H	Compare register 20	CM20	R/W			○		Undefined	
FFFFF2C0H	Timer control register 21	TMC21		○	○			01H	
FFFFF2D0H	Timer 21	TM21	R			○		0000H	
FFFFF2D2H	Compare register 21	CM21	R/W			○		Undefined	
FFFFF2E0H	Timer control register 22	TMC22		○	○			01H	
FFFFF2F0H	Timer 22	TM22	R			○		0000H	
FFFFF2F2H	Compare register 22	CM22	R/W			○		Undefined	
FFFFF300H	Timer control register 23	TMC23		○	○			01H	
FFFFF310H	Timer 23	TM23	R			○		0000H	
FFFFF312H	Compare register 23	CM23	R/W			○		Undefined	
FFFFF320H	Timer control register 24	TMC24		○	○			01H	
FFFFF330H	Timer 24	TM24	R			○		0000H	
FFFFF332H	Compare register 24	CM24	R/W			○		Undefined	
FFFFF340H	Timer control register 3	TMC3		○	○			01H	
FFFFF350H	Timer 3	TM3	R			○		0000H	
FFFFF352H	Capture/compare register 3	CC3	R/W			○		Undefined	
FFFFF354H	Capture register 3	CP3		R			○		
FFFFF360H	PWM control register 3	PWMC0	R/W	○	○			05H	
FFFFF362H	PWM modulo register 0	PWM0				○		Undefined	
FFFFF364H	PWM prescaler register 0	PWPR0		○	○			00H	
FFFFF368H	PWM control register 1	PWMC1		○	○			05H	
FFFFF36AH	PWM modulo register 1	PWM1				○		Undefined	
FFFFF36CH	PWM prescaler register 1	PWPR1		○	○			00H	
FFFFF370H	PWM control register 2	PWMC2		○	○			05H	
FFFFF372H	PWM modulo register 2	PWM2				○		Undefined	
FFFFF374H	PWM prescaler register 2	PWPR2		○	○			00H	
FFFFF378H	PWM control register 3	PWMC3		○	○			05H	
FFFFF37AH	PWM modulo register 3	PWM3				○		Undefined	
FFFFF37CH	PWM prescaler register 3	PWPR3		○	○			00H	
FFFFF380H	A/D converter mode register 0	ADM0		○	○			07H	
FFFFF382H	A/D converter mode register 1	ADM1		○	○				
FFFFF390H	A/D conversion result register 0	ADCR0		R	○	○			Undefined
FFFFF392H	A/D conversion result register 1	ADCR1			○	○			
FFFFF394H	A/D conversion result register 2	ADCR2			○	○			
FFFFF396H	A/D conversion result register 3	ADCR3			○	○			
FFFFF398H	A/D conversion result register 4	ADCR4			○	○			
FFFFF39AH	A/D conversion result register 5	ADCR5			○	○			
FFFFF39CH	A/D conversion result register 6	ADCR6			○	○			

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 bit	8 bits	16 bits	32bits	
FFFFF39EH	A/D conversion result register 7	ADCR7	R	○	○			Undefined
FFFFF3C0H	Port 13 buffer register	PB	R/W	○	○			
FFFFF3C2H	Output latch	RTP		○	○			
FFFFF3D0H	Clock output mode register	CL0M		○	○			00H

### 3.4.9 Specific registers

Specific registers are registers that are protected from being written with illegal data due to program runaway, etc. The write access of these specific registers is executed in a specific sequence, and if abnormal store operations occur, this is notified by the system status register (SYS). The V854 has two specific registers, the clock control register (CKC) and power save control register (PSC). For details of the CKC register, refer to **6.3.3**, and for details of the PSC register, refer to **6.5.2**.

The following sequence shows the data setting of the specific registers.

- (1) Set the PSW NP bit to 1 (interrupt disabled).
- (2) Write arbitrary 8-bit data in the command register (PRCMD).
- (3) Write the set data in the specific registers (by the following instructions).
  - Store instruction (ST/SST instruction)
  - Bit manipulation instruction (SET1/CLR1/NOT1 instruction)
- (4) Return the PSW NP bit to 0 (interrupt disable canceled).
- (5) To shift to the software STOP mode or IDLE mode, insert the NOP instructions (2 or 5 instructions).

No special sequence is required when reading the specific registers.

**Cautions 1.** If an interrupt request is accepted between the time PRCMD is issued (2) and the specific register write operation (3) that follows immediately after, the write operation to the specific register is not performed and a protection error (PRERR bit of SYS register is “1”) may occur. Therefore, set the NP bit of PSW to 1 (1) to disable the acceptance of INT/NMI. The above also applies when a bit manipulation instruction is used to set a specific register. Moreover, to ensure that the execution routine following release of the software STOP/IDLE mode is performed correctly, insert the NOP instruction as a dummy instruction (5). If the value of the ID bit of PSW does not change as the result of execution of the instruction to return the NP bit to 0 (4), insert two NOP instructions, and if the value of the ID bit of PSW changes, insert five NOP instructions. A description example is given below.

```
[Description example] : In case of PSC register
LDSR rX,5           ; NP bit = 1
ST.B r0,PRCMD [r0] ; Write to PRCMD
ST.B rD,PSC [r0]   ; PSC register setting
LDSR rY,5           ; NP bit = 0
NOP                 ; Dummy instruction (2 or 5 instructions)
                   :
NOP
(next instruction) ; Execution routine following cancellation of software STOP/IDLE mode
                   :
```

rX: Value to be written to PSW  
 rY: Value to be written back to PSW  
 rD: Value to be set to PSC

When saving the value of PSW, the value of PSW prior to setting the NP bit must be transferred to the rY register.

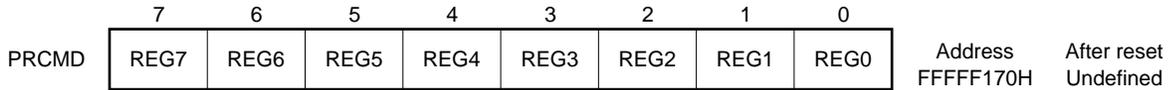
2. The instructions ((4) interrupt disable cancel, (5) NOP instruction) following the store instruction for the specific register for setting the software STOP mode and IDLE mode are executed before a power save mode is entered.

**(1) Command Register (PRCMD)**

The command register (PRCMD) is a register used when write-accessing the special register to prevent incorrect writing to the special registers due to the erroneous program execution.

This register can be read/written in 8-bit units. It becomes undefined values in a read cycle.

Occurrence of illegal store operations can be checked by the PRERR bit of the SYS register.



Bit Position	Bit Name	Function						
7 to 0	REG7 to REG0	Registration Code <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Specific Register</th> <th>Registration Code</th> </tr> </thead> <tbody> <tr> <td>CKC</td> <td>Arbitrary 8-bit data</td> </tr> <tr> <td>PSC</td> <td>Arbitrary 8-bit data</td> </tr> </tbody> </table>	Specific Register	Registration Code	CKC	Arbitrary 8-bit data	PSC	Arbitrary 8-bit data
Specific Register	Registration Code							
CKC	Arbitrary 8-bit data							
PSC	Arbitrary 8-bit data							

**(2) System status register (SYS)**

This register is allocated with status flags showing the operating state of the entire system. This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
SYS	0	0	0	PRERR	0	0	0	UNLOCK	Address FFFFF078H	After reset 0000000XB

Bit Position	Bit Name	Function
4	PRERR	Protection Error Flag Indicates that writing to a special register has not be executed in the correct sequence, and protection error occurred. Accumulate flag 0 : Protection error does not occur. 1 : Protection error occurs.
0	UNLOCK	Unlock Status Flag Read-only flag. Indicates the PLL unlock state. (For details, refer to <b>6.4 PLL Stabilization</b> .) 0 : Locked 1 : Unlocked

**Operation conditions of PRERR Flag**

- Set conditions: (PRERR = "1")
  - (1) If the store instruction most recently executed to peripheral I/O does not write data to the PRCMD register, but to the specific register.
  - (2) If the first store instruction executed after the write operation to the PRCMD register is to a peripheral I/O register other than the specific registers.
  
- Reset conditions: (PRERR = "0")
  - (1) When "0" is written to the PRERR flag of the SYS register.
  - (2) At system reset.

**[MEMO]**

## CHAPTER 4 BUS CONTROL FUNCTION

The V854 is provided with an external bus interface function by which external memories such as ROM and RAM, and I/O can be connected.

### 4.1 Features

- 16-bit data bus
- Can be connected to external devices with pins having alternate function as port
- Wait function
  - Programmable wait function of up to 3 states per 2 blocks
  - External wait function through  $\overline{\text{WAIT}}$  pin
- Idle state insertion function
- Bus mastership arbitration function
- Bus hold function

## 4.2 Bus Control Pins and Control Register

### 4.2.1 Bus control pins

The following pins are used for interfacing to external devices:

External Bus Interface Function	Corresponding Port (pins)
Address/data bus (AD0 to AD7)	Port 4 (P40 to P47)
Address/data bus (AD8 to AD15)	Port 5 (P50 to P57)
Address bus (A16 to A23)	Port 6 (P60 to P67)
Read/write control ( $\overline{\text{LBEN}}$ , $\overline{\text{UBEN}}$ , $\overline{\text{R/W}}$ , $\overline{\text{DSTB}}$ , $\overline{\text{WRL}}$ , $\overline{\text{WRH}}$ , $\overline{\text{RD}}$ )	Port 9 (P90 to P93)
Address strobe (ASTB)	Port 9 (P94)
Bus hold control ( $\overline{\text{HLDRQ}}$ , $\overline{\text{HLDK}}$ )	Port 9 (P95, P96)
External wait control ( $\overline{\text{WAIT}}$ )	$\overline{\text{WAIT}}$

The bus interface function of each pin is enabled by the memory expansion mode register (MM). In ROM-less mode, the bus interface function of each pin is unconditionally enabled by the MODE input (n = 0 to 2). For the details of specifying an operation mode of the external bus interface, refer to **3.4.6 (1) Memory expansion mode register (MM)**.

### 4.2.2 Control register

#### (1) System control register (SYC)

This register switches control signals for bus interface.

The system control register can be read/written in 8- or 1-bit units.

SYC	7	6	5	4	3	2	1	0	Address FFFFFF064H	After reset 00H (in single-chip mode 1, 2 and ROM-less mode 1) 01H (in ROM-less mode 2)
	0	0	0	0	0	0	0	BIC		

Bit Position	Bit Name	Function
1	BIC	Bus Interface Control Changes bus interface control signal 0 : $\overline{\text{DSTB}}$ , $\overline{\text{R/W}}$ , $\overline{\text{UBEN}}$ , $\overline{\text{LBEN}}$ signal outputs 1 : $\overline{\text{RD}}$ , $\overline{\text{WRL}}$ , $\overline{\text{WRH}}$ , $\overline{\text{UBEN}}$ signal outputs

$\overline{\text{RD}}$ ,  $\overline{\text{WRL}}$ ,  $\overline{\text{WRH}}$ , and  $\overline{\text{UBEN}}$  signals are output immediately after reset in ROM-less mode 2.

### 4.3 Bus Access

#### 4.3.1 Number of access clocks

The number of basic clocks necessary for accessing each resource is as follows:

Bus Cycle Type	Resource (bus width)			
	Internal ROM (32 bits)	Internal RAM (32 bits)	Internal Peripheral I/O (16 bits)	External Memory (16 bits)
Instruction fetch	1	3	Disabled	3 + n
Operand data access	3	1	3 + n	3 + n

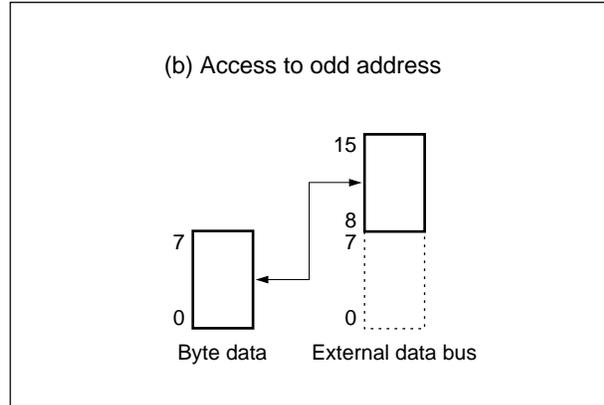
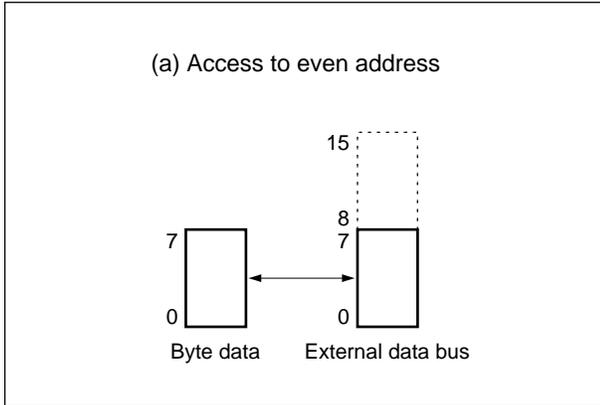
- Remarks**
1. Unit : clock/access
  2. n : number of wait insertions

**4.3.2 Bus width**

The V854 carries out peripheral I/O access and external memory access in 8-, 16-, or 32-bit. The following shows the operation for each access.

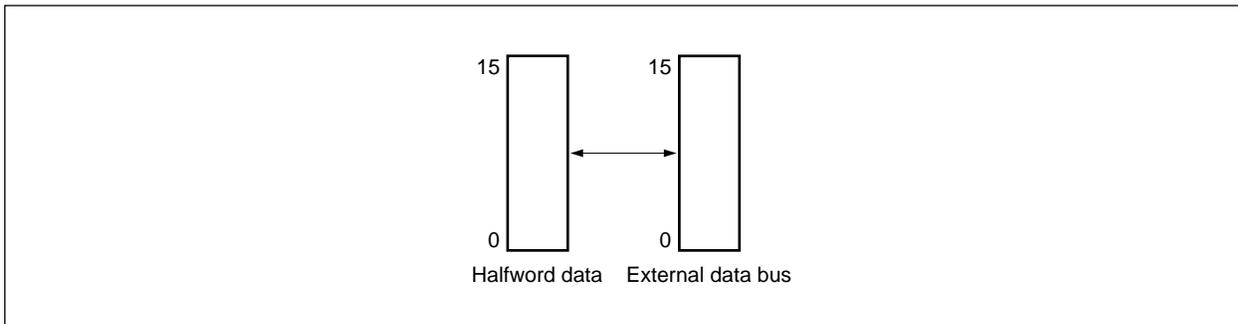
**(1) Byte access (8 bits)**

Byte access is divided into two types, the access to even address and the access to odd address.



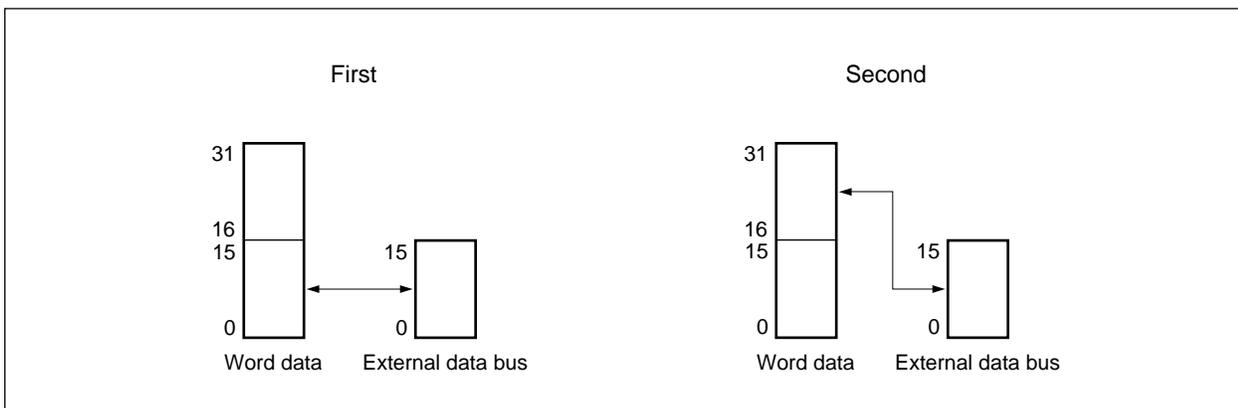
**(2) Halfword access (16 bits)**

In halfword access to external memory, data is dealt with as it is because the data bus is fixed to 16 bits.



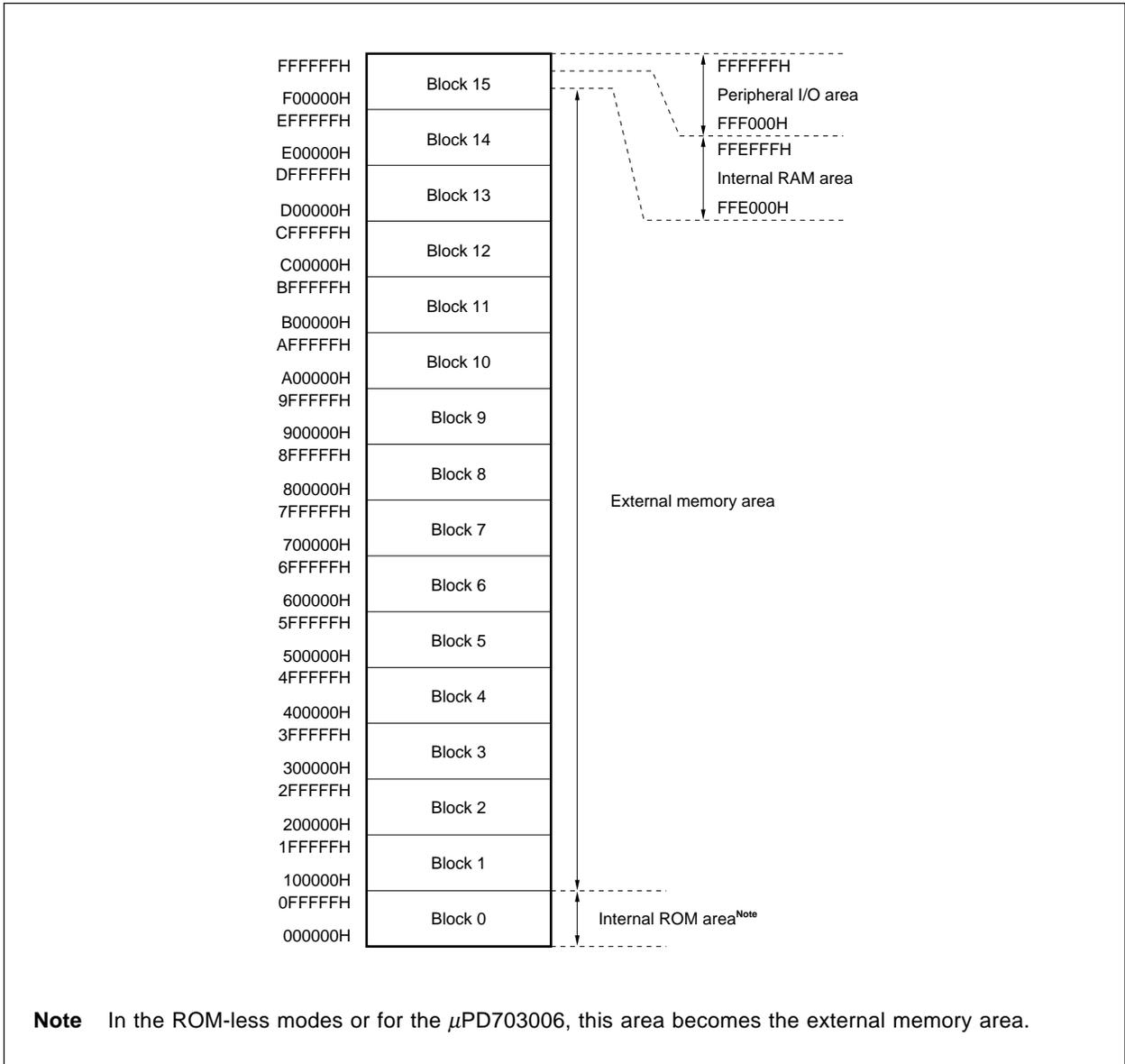
**(3) Word access (32 bits)**

In word access to external memory, lower halfword is accessed first and then the upper halfword is accessed.



### 4.4 Memory Block Function

The 16-Mbyte memory space is divided into memory blocks of 1-Mbyte units. The programmable wait function and bus cycle operation mode can be independently controlled for every two memory blocks.



## 4.5 Wait Function

### 4.5.1 Programmable wait function

To facilitate interfacing with low-speed memories and I/O devices, up to 3 data wait states can be inserted in a bus cycle for two memory blocks. The number of wait states can be programmed by using data wait control register (DWC). Immediately after the system has been reset, three data wait states are automatically programmed for all memory blocks.

#### (1) Data wait control register (DWC)

This register can be read/written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DWC	DW71	DW70	DW61	DW60	DW51	DW50	DW41	DW40	DW31	DW30	DW21	DW20	DW11	DW10	DW01	DW00	Address	After reset
																	FFFFFF060H	FFFFH

Bit Position	Bit Name	Function																																	
15 to 0	DWn1 DWn0 (n = 0 to 7)	Data Wait Specifies number of wait states to be inserted																																	
		<table border="1" style="width: 100%; border-collapse: collapse; margin-bottom: 10px;"> <thead> <tr> <th style="width: 15%;">DWn1</th> <th style="width: 15%;">DWn0</th> <th style="width: 70%;">Number of wait states to be inserted</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">2</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">3</td> </tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">n</th> <th style="width: 85%;">Blocks into which wait states are inserted</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Blocks 0/1</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Blocks 2/3</td> </tr> <tr> <td style="text-align: center;">2</td> <td>Blocks 4/5</td> </tr> <tr> <td style="text-align: center;">3</td> <td>Blocks 6/7</td> </tr> <tr> <td style="text-align: center;">4</td> <td>Blocks 8/9</td> </tr> <tr> <td style="text-align: center;">5</td> <td>Blocks 10/11</td> </tr> <tr> <td style="text-align: center;">6</td> <td>Blocks 12/13</td> </tr> <tr> <td style="text-align: center;">7</td> <td>Blocks 14/15</td> </tr> </tbody> </table>	DWn1	DWn0	Number of wait states to be inserted	0	0	0	0	1	1	1	0	2	1	1	3	n	Blocks into which wait states are inserted	0	Blocks 0/1	1	Blocks 2/3	2	Blocks 4/5	3	Blocks 6/7	4	Blocks 8/9	5	Blocks 10/11	6	Blocks 12/13	7	Blocks 14/15
DWn1	DWn0	Number of wait states to be inserted																																	
0	0	0																																	
0	1	1																																	
1	0	2																																	
1	1	3																																	
n	Blocks into which wait states are inserted																																		
0	Blocks 0/1																																		
1	Blocks 2/3																																		
2	Blocks 4/5																																		
3	Blocks 6/7																																		
4	Blocks 8/9																																		
5	Blocks 10/11																																		
6	Blocks 12/13																																		
7	Blocks 14/15																																		

- Cautions**
1. **Block 0 is reserved for the internal ROM area in the single-chip mode. It is not subject to programmable wait control, regardless of the setting of DWC, and is always accessed without wait states.**
  2. **The internal RAM area of block 15 is not subject to programmable wait control and is always accessed without wait states. The peripheral I/O area of this block is not subject to programmable wait control, either. The only wait control is dependent upon the execution of each peripheral function.**

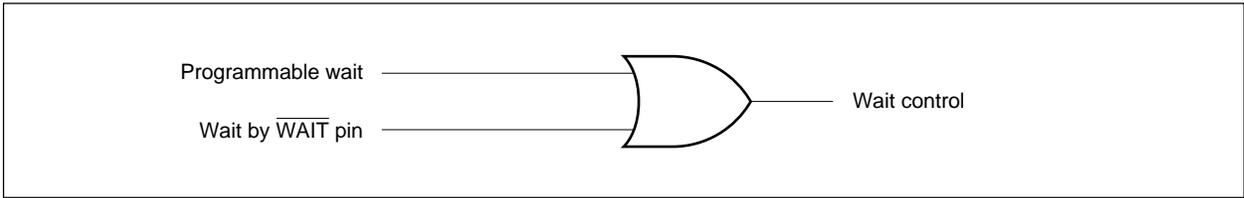
**4.5.2 External wait function**

When an extremely slow device, I/O, or asynchronous system is connected, any number of wait states can be inserted in a bus cycle by sampling the external wait pin ( $\overline{\text{WAIT}}$ ) to synchronize with the external device.

The external  $\overline{\text{WAIT}}$  signal does not affect the access times of the internal ROM, internal RAM, and peripheral I/O areas. Input of the external  $\overline{\text{WAIT}}$  signal can be done asynchronously to CLKOUT and is sampled at the falling edge of the clock in the T2 and TW states of a bus cycle. If the set up and hold time of the  $\overline{\text{WAIT}}$  input are not satisfied, the wait state may or may not be inserted in the next state.

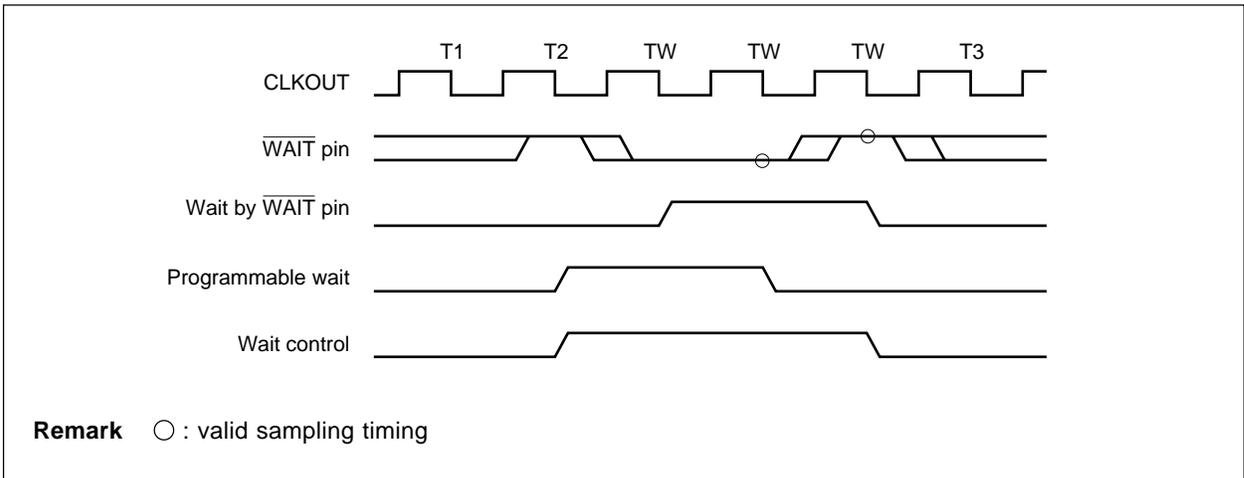
**4.5.3 Relations between programmable wait and external wait**

A wait cycle is inserted as a result of an OR operation between the wait cycle specified by the set value of programmable wait and the wait cycle controlled by the  $\overline{\text{WAIT}}$  pin. In other words, the number of wait cycles is determined by the programmable wait value or the length of evaluation at the  $\overline{\text{WAIT}}$  input pin.



For example, if the number of programmable wait states is 2 and the timing of the  $\overline{\text{WAIT}}$  pin input signal is as illustrated below, three wait states will be inserted in the bus cycle.

**Figure 4-1. Example of Inserting Wait States**



### 4.6 Idle State Insertion Function

To facilitate interfacing with low-speed memory devices and meeting the data output float delay time ( $t_{DF}$ ) on memory read accesses, one idle state (TI) can be inserted into the current bus cycle after the T3 state. The bus cycle following continuous bus cycles starts after one idle state.

Specifying insertion of the idle state is programmable by using the bus cycle control register (BCC).

Immediately after the system has been reset, idle state insertion is automatically programmed for all memory blocks.

#### (1) Bus cycle control register (BCC)

This register can be read/written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
BCC	BC71	0	BC61	0	BC51	0	BC41	0	BC31	0	BC21	0	BC11	0	BC01	0	Address FFFF062H	After reset AAAAH

Bit Position	Bit Name	Function																		
15, 13, 11, 9, 7, 5, 3, 1	BCn1 (n = 0 to 7)	Bus Cycle Specifies insertion of idle state.  0: Not inserted 1: Inserted  <table border="1" style="width: 100%; border-collapse: collapse; margin-left: 20px;"> <thead> <tr> <th style="width: 10%;">n</th> <th style="width: 90%;">Blocks into Which Idle State Is Inserted</th> </tr> </thead> <tbody> <tr><td style="text-align: center;">0</td><td>Blocks 0/1</td></tr> <tr><td style="text-align: center;">1</td><td>Blocks 2/3</td></tr> <tr><td style="text-align: center;">2</td><td>Blocks 4/5</td></tr> <tr><td style="text-align: center;">3</td><td>Blocks 6/7</td></tr> <tr><td style="text-align: center;">4</td><td>Blocks 8/9</td></tr> <tr><td style="text-align: center;">5</td><td>Blocks 10/11</td></tr> <tr><td style="text-align: center;">6</td><td>Blocks 12/13</td></tr> <tr><td style="text-align: center;">7</td><td>Blocks 14/15</td></tr> </tbody> </table>	n	Blocks into Which Idle State Is Inserted	0	Blocks 0/1	1	Blocks 2/3	2	Blocks 4/5	3	Blocks 6/7	4	Blocks 8/9	5	Blocks 10/11	6	Blocks 12/13	7	Blocks 14/15
n	Blocks into Which Idle State Is Inserted																			
0	Blocks 0/1																			
1	Blocks 2/3																			
2	Blocks 4/5																			
3	Blocks 6/7																			
4	Blocks 8/9																			
5	Blocks 10/11																			
6	Blocks 12/13																			
7	Blocks 14/15																			

**Cautions**

1. Block 0 is reserved for the internal ROM area in the single-chip mode; therefore, no insertion of the idle state can be specified for block 0 regardless of the BCC settings.
2. The internal RAM area and peripheral I/O area of block 15 are not subject to insertion of the idle state.
3. Be sure to set bits 0, 2, 4, 6, 8, 10, 12, and 14 to 0. If these bits are set to 1, the operation is not guaranteed.

## 4.7 Bus Hold Function

### 4.7.1 Outline of function

When P95 and P96 of port 9 are programmed to be in the control mode, the functions of the  $\overline{\text{HLDRQ}}$  and  $\overline{\text{HLDAK}}$  pins become valid.

When the  $\overline{\text{HLDRQ}}$  pin becomes active (low) indicating that another bus master is requesting acquisition of the bus, the external address/data bus and strobe pins go into a high-impedance state, and the bus is released (bus hold status). When the  $\overline{\text{HLDRQ}}$  pin becomes inactive (high) indicating that the request for the bus is cleared, these pins are driven again.

During bus hold period, the V854 continues internal operation until the next external memory access.

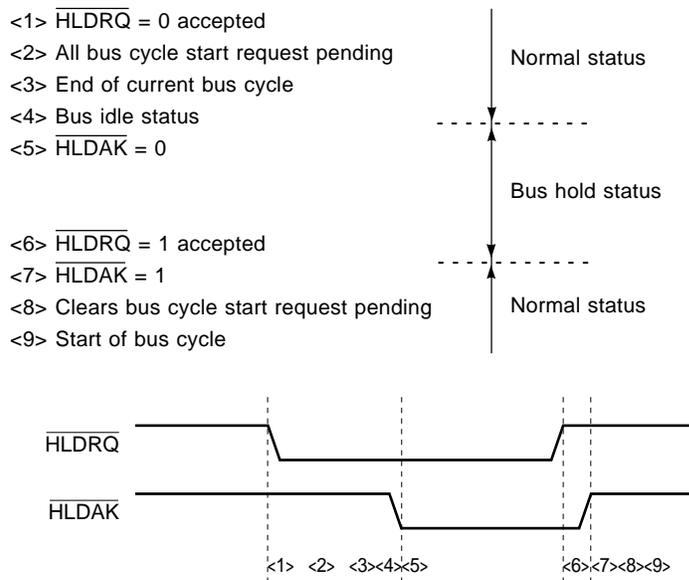
In the bus hold status, the  $\overline{\text{HLDAK}}$  pin becomes active (low).

This feature can be used to design a system where two or more bus masters exist, such as when multi-processor configuration is used and when a DMA controller is connected.

Bus hold request is not acknowledged between the first and the second word access. Bus hold request is also acknowledged between read access and write access in read modify write access of bit manipulation instruction.

### 4.7.2 Bus hold procedure

The procedure of the bus hold function is illustrated below.



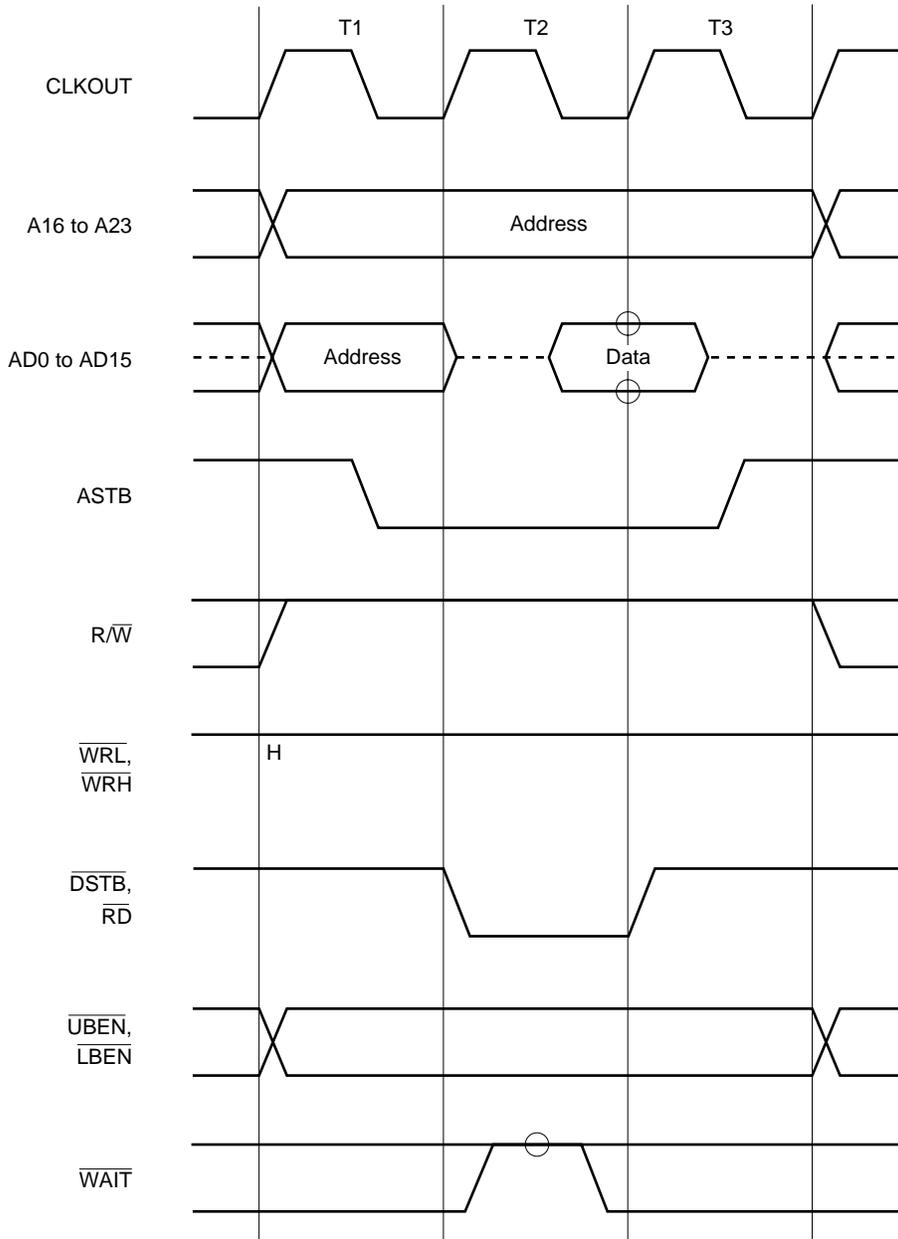
### 4.7.3 Operation in power save mode

In the STOP or IDLE mode, the system clock is stopped. Consequently, the bus hold status is not set even if the  $\overline{\text{HLDRQ}}$  pin becomes active.

In the HALT mode, the  $\overline{\text{HLDAK}}$  pin immediately becomes active when the  $\overline{\text{HLDRQ}}$  pin becomes active, and the bus hold status is set. When the  $\overline{\text{HLDRQ}}$  pin becomes inactive, the  $\overline{\text{HLDAK}}$  pin becomes inactive. As a result, the bus hold status is cleared, and the HALT mode is set again.

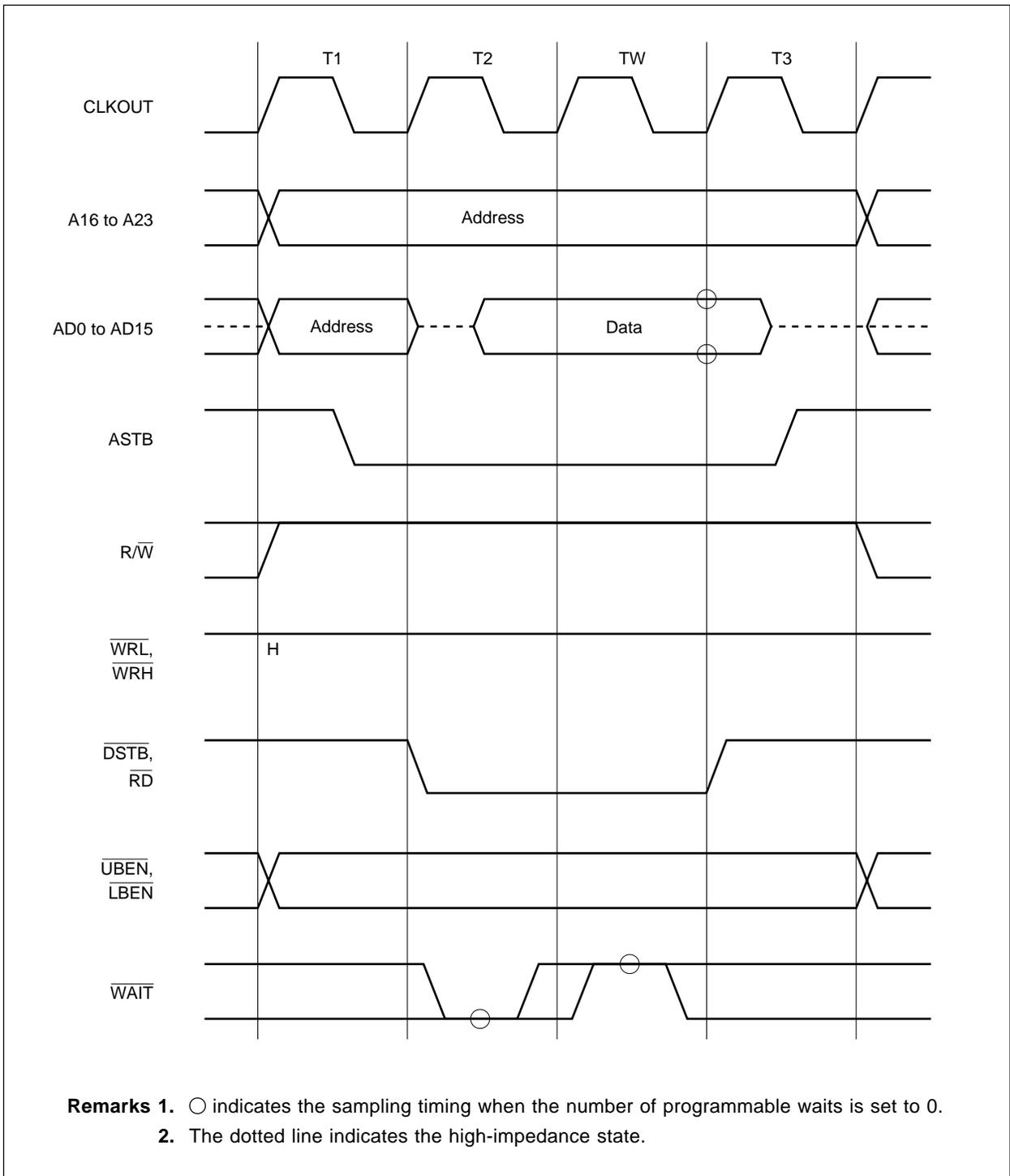
4.8 Bus Timing

(1) Memory read (0 wait)

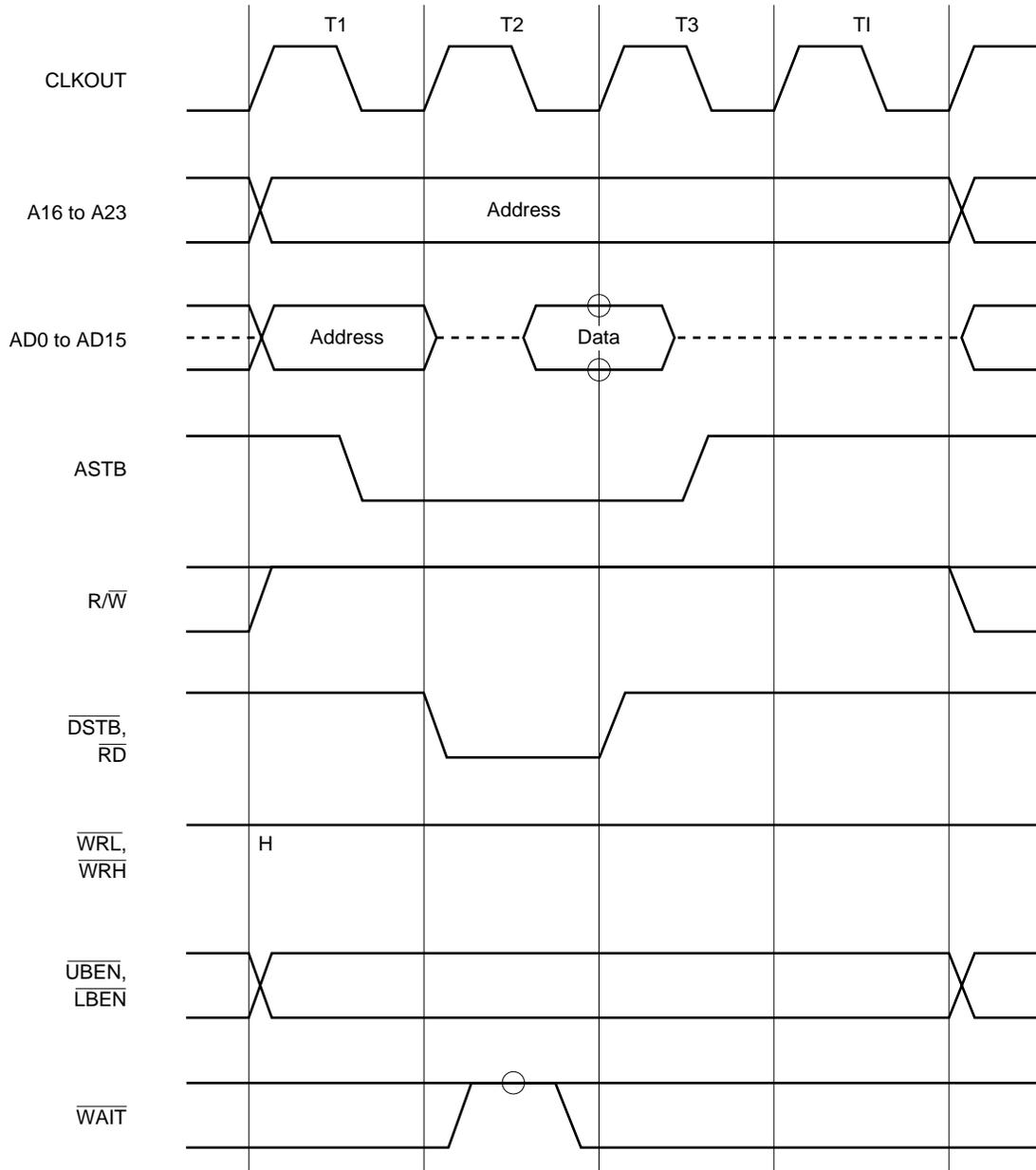


- Remarks**
- indicates the sampling timing when the number of programmable waits is set to 0.
  - The dotted line indicates the high-impedance state.

(2) Memory read (1 wait)

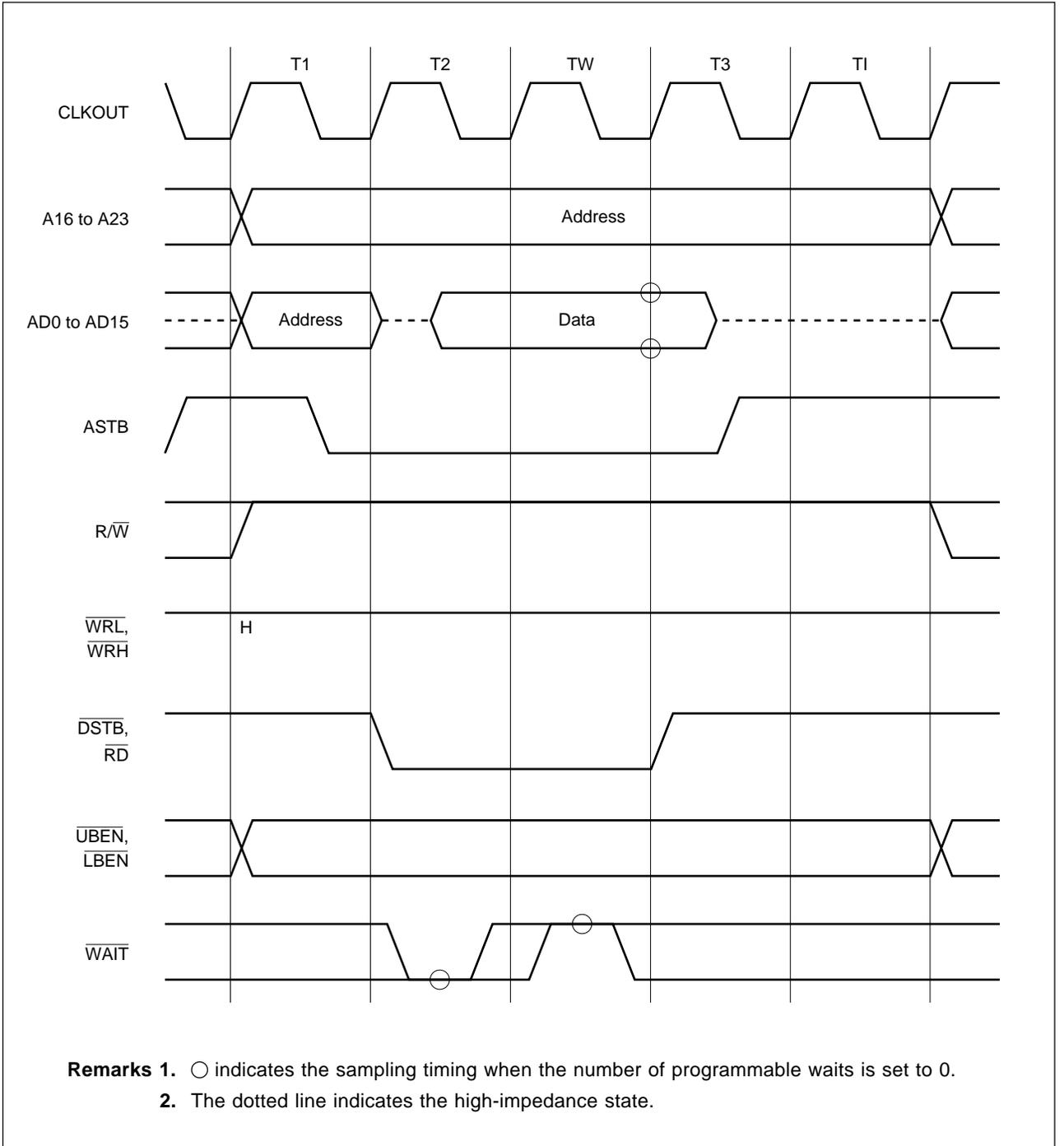


(3) Memory read (0 wait, idle state)

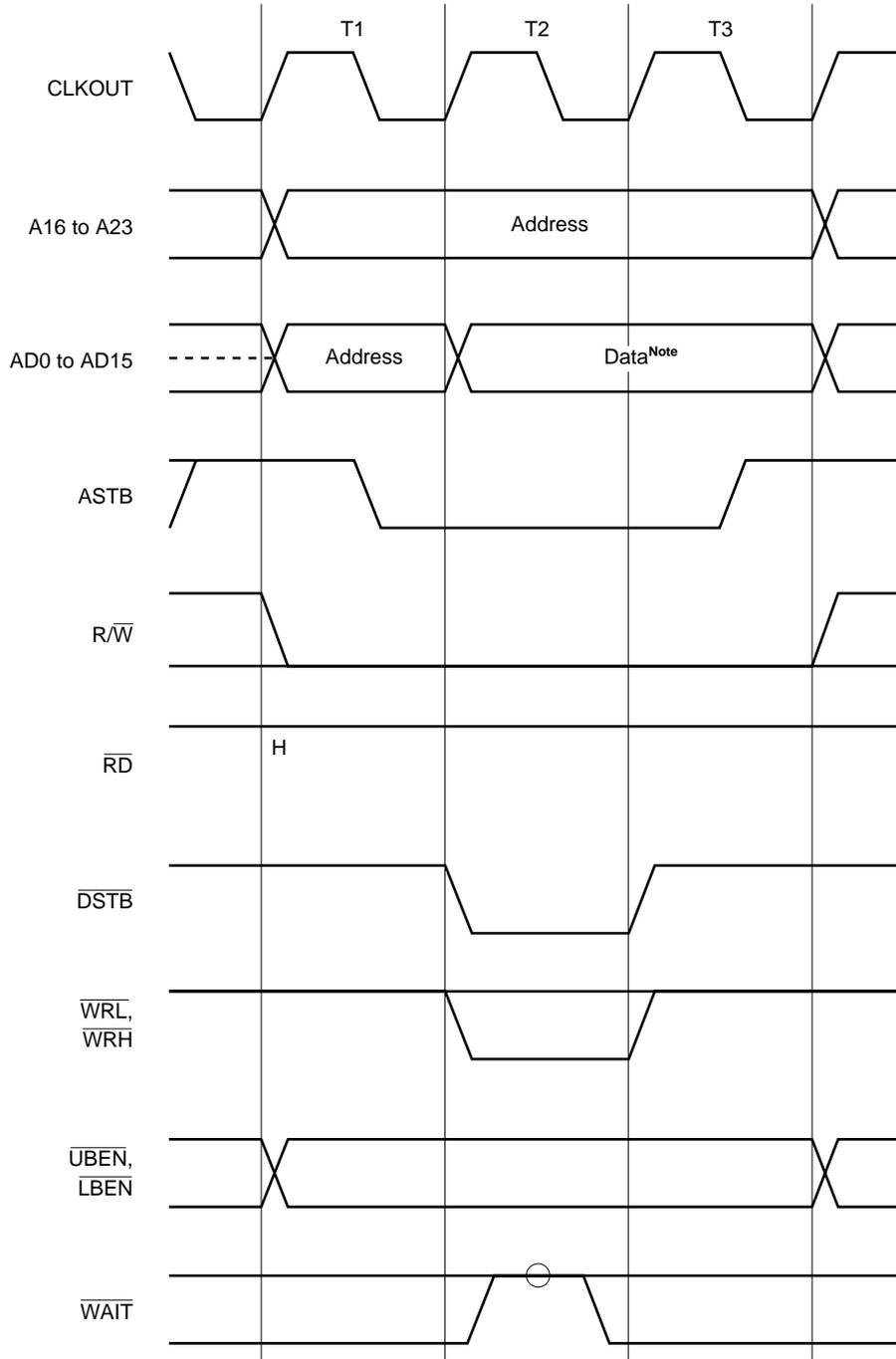


- Remarks**
- indicates the sampling timing when the number of programmable waits is set to 0.
  - The dotted line indicates the high-impedance state.

(4) Memory read (1 wait, idle state)



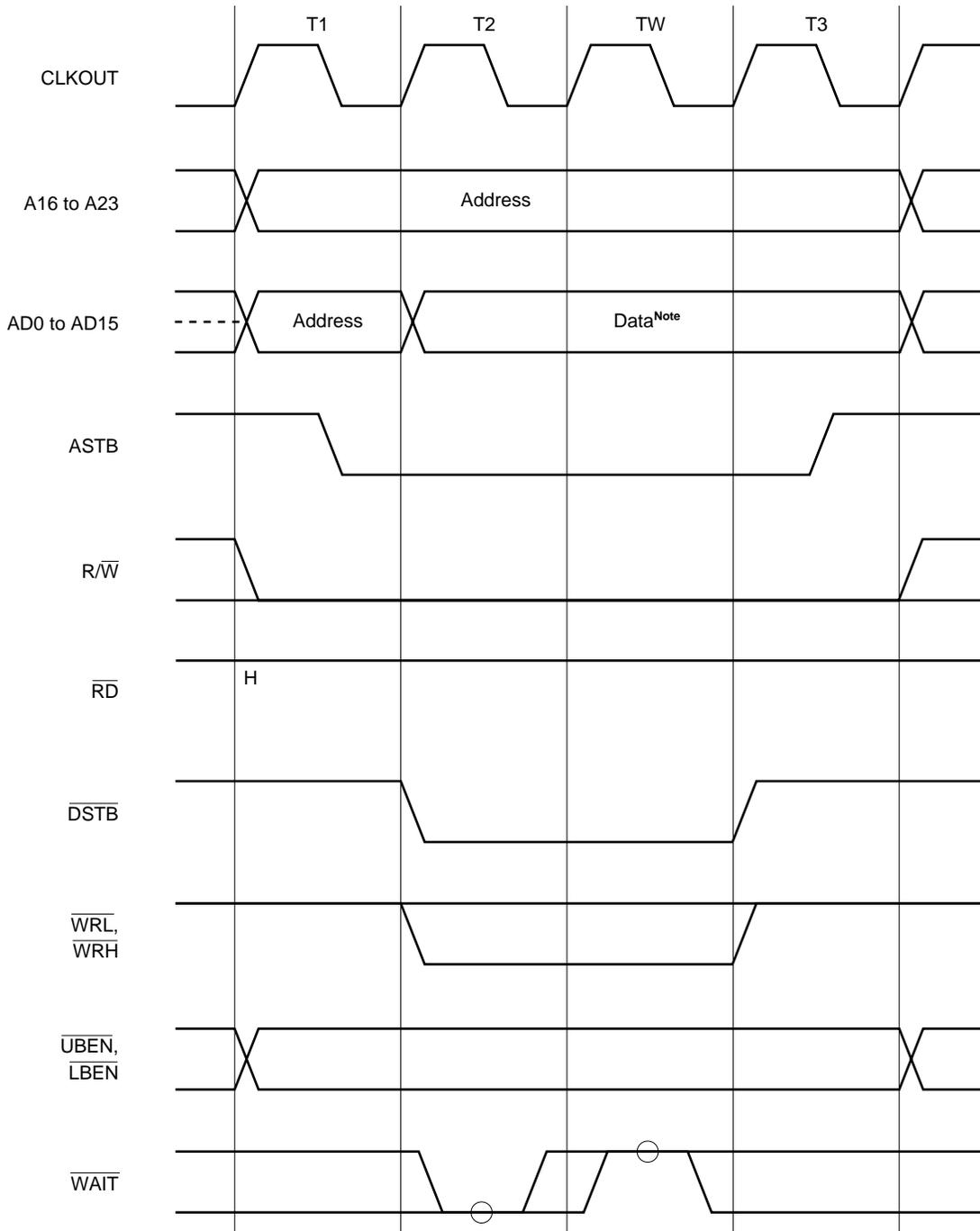
(5) Memory write (0 wait)



**Note** AD0 to AD7 output invalid data when odd address byte data is accessed.  
AD8 to AD15 output invalid data when even address byte data is accessed.

**Remarks** 1. ○ indicates the sampling timing when the number of programmable waits is set to 0.  
2. The dotted line indicates the high-impedance state.

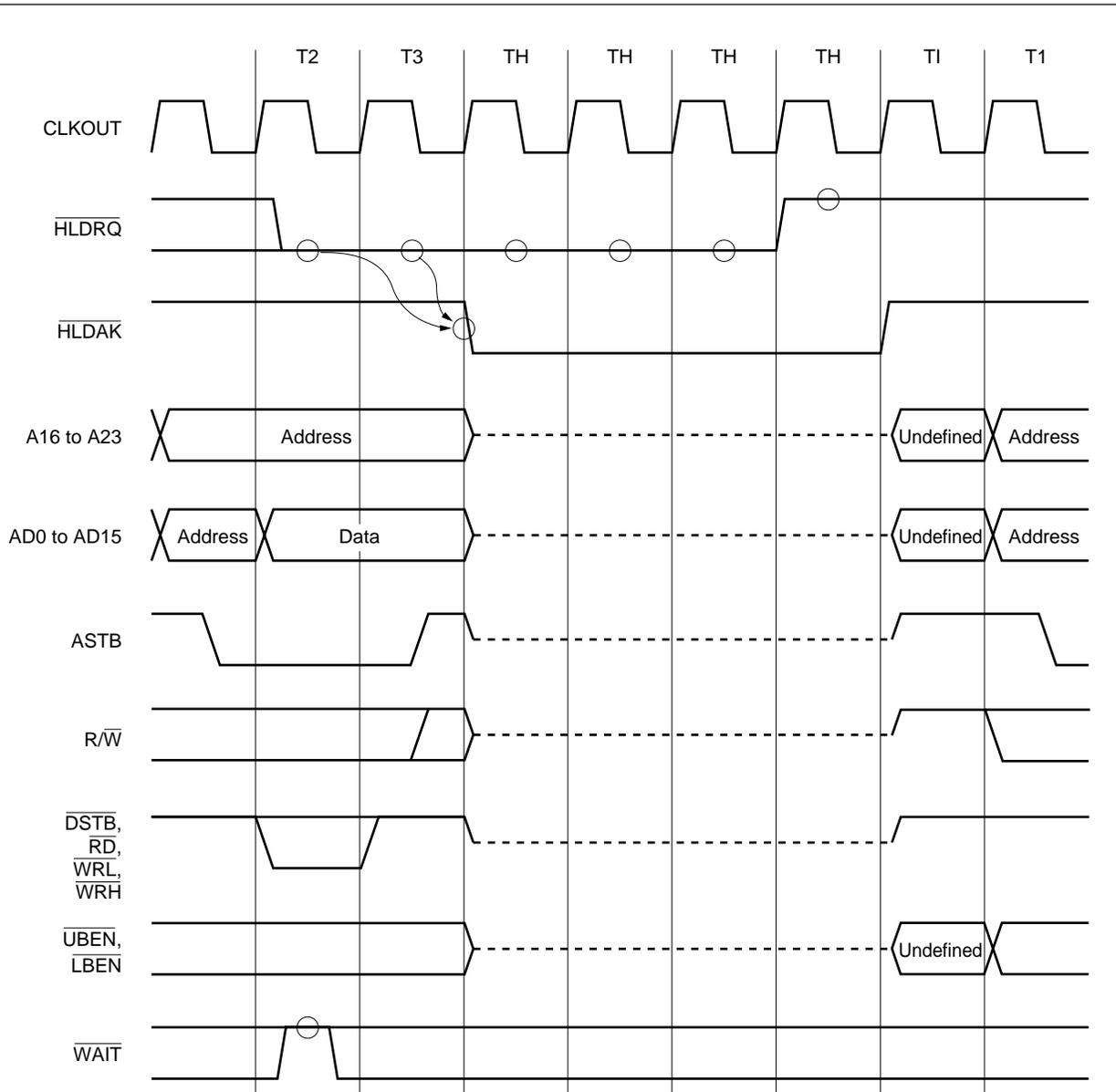
(6) Memory write (1 wait)



**Note** AD0 to AD7 output invalid data when odd address byte data is accessed.  
 AD8 to AD15 output invalid data when even address byte data is accessed.

**Remarks 1.** ○ indicates the sampling timing when the number of programmable waits is set to 0.  
**2.** The dotted line indicates the high-impedance state.

(7) Bus hold timing



- Remarks**
1. ○ indicates the sampling timing.
  2. The dotted line indicates the high-impedance state.

**Caution** When the bus hold state is entered after the write cycle, a high-level signal may be briefly output from the  $R/\bar{W}$  pin immediately before the  $\overline{HLDAK}$  signal changes from the high level to the low level.

## 4.9 Bus Priority

There are four external bus cycles: bus hold, operand data access, instruction fetch (branch), and instruction fetch (continuous). The bus hold cycle is given the highest priority, followed by operand data access, instruction fetch (branch), and instruction fetch (continuous) in that order.

The instruction fetch cycle may be inserted in between the read access and write access of read-modify-write access.

No instruction fetch cycle and bus hold are inserted between the lower half-word access and higher half-word access of word operations.

**Table 4-1. Bus Priority**

External Bus Cycle	Priority
Bus hold	1
Operand data access	2
Instruction fetch (branch)	3
Instruction fetch (continuous)	4

## 4.10 Memory Boundary Operation Condition

### 4.10.1 Program space

- (1) Do not execute branch to the peripheral I/O area or continuous fetch from the internal RAM area to peripheral I/O area. Of course, it is impossible to fetch from external memory. If branch or instruction fetch is executed nevertheless, the NOP instruction code is continuously fetched.
- (2) A prefetch operation straddling over the peripheral I/O area (invalid fetch) does not take place if a branch instruction exists at the upper-limit address of the internal RAM area.

### 4.10.2 Data space

Only the address aligned at the half-word (when the least significant bit of the address is "0")/word (when the lowest 2 bits of the address are "0") boundary is accessed for data half-word (16 bits)/word (32 bits) long.

Therefore, access that straddles over the memory or memory block boundary does not take place.

For the details, refer to **V850 Family Architecture User's Manual** .

### 4.11 Internal Peripheral I/O Interface

Access to the internal peripheral I/O area is not output to the external bus. Therefore, the internal peripheral I/O area can be accessed in parallel with instruction fetch access.

Accesses to the internal peripheral I/O area takes, in most cases, three clock cycles. However, when accessing to certain timer/counter registers, wait may take place from 3 to 4 cycles.

Peripheral I/O Register	Access	Number of Waits	Number of Cycles
CC00 to CC03	Read	2	8
	Write	0/2	6/8
CC00L to CC03L, CC3	Read	1	4
	Write	0/1	3/4
CP10 to CP13, CM10 CM11, TM0, TM1	Read	2	8
	Write	0	6
CP10L, CP13L, CM10L, CM11L, TM0L, TM1L, CP3, TM3	Read	1	4
	Write	0	3
CM20 to CM24	Read	0/1	6/8
	Write	0/1	6/8
TM20 to TM24	Read	0/1	6/8
	Write	0	6
Others	Read	0	3
	Write	0	3

**Remark** In 32-bit access, two 16-bit accesses are executed. Therefore, the numbers of waits and cycles are twice those in 16-bit access.

## CHAPTER 5 INTERRUPT/EXCEPTION PROCESSING FUNCTION

The V854 is provided with a dedicated interrupt controller (INTC) for interrupt processing and can process a total of 32 interrupt requests.

An interrupt is an event that occurs independently of program execution, and an exception is an event that occurs dependently on program execution. Generally, an exception takes precedence over an interrupt.

The V854 can process interrupt requests from the internal peripheral hardware and external sources. Moreover, exception processing can be started by the TRAP instruction (software exception) or by generation of an exception event (fetching of an illegal op code).

### 5.1 Features

- Interrupt
  - Non-maskable interrupt: 1 source
  - Maskable interrupt: 31 sources
  - 8 levels programmable priorities control
  - Multiple interrupt control according to priority
  - Mask specification for each maskable interrupt request
  - Noise elimination, edge detection, and valid edge of external interrupt request signal can be specified.
- Exception
  - Software exception: 32 sources
  - Exception trap: 1 source (illegal op code exception)

Interrupt/exception sources are listed in Table 5-1.

Table 5-1. Interrupt List (1/2)

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Vector Address	Restored PC
		Name	Control Register	Generating Source	Generating Unit				
Reset	Interrupt	RESET	–	Reset input	–	–	0000H	00000000H	Undefined
Non-maskable	Interrupt	NMI	–	NMI input	–	–	0010H	00000010H	nextPC
Software exception	Exception	TRAP0n <sup>Note</sup>	–	TRAP instruction	–	–	004n <sup>Note</sup> H	00000040H	nextPC
	Exception	TRAP1n <sup>Note</sup>	–	TRAP instruction	–	–	005n <sup>Note</sup> H	00000050H	nextPC
Exception trap	Exception	ILGOP	–	Illegal op code	–	–	0060H	00000060H	nextPC
Maskable	Interrupt	INTOV0/ INTP04/ INTP05	OVIC0	Timer 0 overflow/ INTP04, INTP05 input	Pin/RPU	0	0080H	00000080H	nextPC
	Interrupt	INTOV1/ INTP14	OVIC1	Timer 1 overflow/ INTP14 input	Pin/RPU	1	0090H	00000090H	nextPC
	Interrupt	INTP00/ INTCC00	CC0IC0	INTP00/CC00 coincidence	Pin/RPU	2	00A0H	000000A0H	nextPC
	Interrupt	INTP01/ INTCC01	CC0IC1	INTP01/CC01 coincidence	Pin/RPU	3	00B0H	000000B0H	nextPC
	Interrupt	INTP02/ INTCC02	CC0IC2	INTP02/CC02 coincidence	Pin/RPU	4	00C0H	000000C0H	nextPC
	Interrupt	INTP03/ INTCC03	CC0IC3	INTP03/CC03 coincidence	Pin/RPU	5	00D0H	000000D0H	nextPC
	Interrupt	INTCP10	P1IC0	INTP10 input	Pin	6	00E0H	000000E0H	nextPC
	Interrupt	INTCP11	P1IC1	INTP11 input	Pin	7	00F0H	000000F0H	nextPC
	Interrupt	INTCP12	P1IC2	INTP12 input	Pin	8	0100H	00000100H	nextPC
	Interrupt	INTCP13	P1IC3	INTP12/INTP13 input	Pin	9	0110H	00000110H	nextPC
	Interrupt	INTCM10	CM1IC0	CM10 coincidence	RPU	10	0120H	00000120H	nextPC
	Interrupt	INTCM11	CM1IC1	CM11 coincidence	RPU	11	0130H	00000130H	nextPC
	Interrupt	INTP20/ INTCM20	CM2IC0	INTP20/CM20 coincidence	Pin/RPU	12	0140H	00000140H	nextPC
	Interrupt	INTP21/ INTCM21	CM2IC1	INTP21/CM21 coincidence	Pin/RPU	13	0150H	00000150H	nextPC
	Interrupt	INTP22/ INTCM22	CM2IC2	INTP22/CM22 coincidence	Pin/RPU	14	0160H	00000160H	nextPC
	Interrupt	INTP23/ INTCM23	CM2IC3	INTP23/CM23 coincidence	Pin/RPU	15	0170H	00000170H	nextPC
Interrupt	INTP24/ INTCM24	CM2IC4	INTP24/CM24 coincidence	Pin/RPU	16	0180H	00000180H	nextPC	

**Note** n: value of 0 to FH

**Remarks 1.** Default Priority : Priority that takes precedence when two or more maskable interrupt requests occur at the same time. The highest priority is 0.

Restored PC : The value of the PC saved to EIPC or FEPC when interrupt/exception processing is started. However, the value of the PC saved when an interrupt is granted during the DIVH (division) instruction execution is the value of the PC of the current instruction (DIVH).

2. The execution address of the illegal instruction when an illegal op code exception occurs is calculated with (Restored PC – 4).

Table 5-1. Interrupt List (2/2)

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Vector Address	Restored PC
		Name	Control Register	Generating Source	Generating Unit				
Maskable	Interrupt	INTP30/ INTCC3	CC3IC0	INTP30/CC3 coincidence	Pin/RPU	17	0190H	00000190H	nextPC
	Interrupt	INTCSI0	CSIC0	CSI0 transmission/ reception completion	CSI	18	01A0H	000001A0H	nextPC
	Interrupt	INTCSI1	CSIC1	CSI1 transmission/ reception completion	CSI	19	01B0H	000001B0H	nextPC
	Interrupt	INTCSI2	CSIC2	CSI2 transmission/ reception completion	CSI	20	01C0H	000001C0H	nextPC
	Interrupt	INTCSI3	CSIC3	CSI3 transmission/ reception completion	CSI	21	01D0H	000001D0H	nextPC
	Interrupt	INTIIC	IIC0	I <sup>2</sup> C interrupt	I <sup>2</sup> C	22	01E0H	000001E0H	nextPC
	Interrupt	INTSER	SEIC0	UART reception error	UART	23	01F0H	000001F0H	nextPC
	Interrupt	INTSR	SRIC0	UART reception completion	UART	24	0200H	00000200H	nextPC
	Interrupt	INTST	STIC0	UART transmission completion	UART	25	0210H	00000210H	nextPC
	Interrupt	INTAD	ADIC0	A/D conversion end	ADC	26	0220H	00000220H	nextPC
	Interrupt	INTP50	P5IC0	INTP50 input	Pin	27	0230H	00000230H	nextPC
	Interrupt	INTP51	P5IC1	INTP51 input	Pin	28	0240H	00000240H	nextPC
	Interrupt	INTP52	P5IC2	INTP52 input	Pin	29	0250H	00000250H	nextPC
Interrupt	INTP53	P5IC3	INTP53 input	Pin	30	0260H	00000260H	nextPC	

- Remarks 1.** Default Priority : Priority that takes precedence when two or more maskable interrupt requests occur at the same time. The highest priority is 0.
- Restored PC : The value of the PC saved to EIPC or FEPC when interrupt/exception processing is started. However, the value of the PC saved when an interrupt is granted during the DIVH (division) instruction execution is the value of the PC of the current instruction (DIVH).
- 2.** The execution address of the illegal instruction when an illegal op code exception occurs is calculated with (Restored PC – 4).

## 5.2 Non-Maskable Interrupt

The non-maskable interrupt is accepted unconditionally, even when interrupts are disabled (DI states) in the interrupt disabled (DI) status. The NMI is not subject to priority control and takes precedence over all the other interrupts.

The non-maskable interrupt request is input from the NMI pin. When the valid edge specified by bit 0 (ESN0) of the external interrupt mode register 0 (INTM0) is detected on the NMI pin, the interrupt occurs.

While the service routine of the non-maskable interrupt is being executed (PSW.NP = 1), the acceptance of another non-maskable interrupt request is kept pending. The pending NMI is accepted after the original service routine of the non-maskable interrupt under execution has been terminated (by the RETI instruction), or when PSW.NP is cleared to 0 by the LDSR instruction. Note that if two or more NMI requests are input during the execution of the service routine for an NMI, the number of NMIs that will be acknowledged after PSW.NP goes to "0", is only one.

The operation at the execution of pending non-maskable interrupt request differs depending on the V854 operation mode.

### (1) In single-chip mode

The operation is returned to the main routine once and at least one instruction in the main routine is executed between the end of the first non-maskable interrupt processing and the start of the pending non-maskable interrupt processing.

### (2) In ROM-less mode

The pending non-maskable interrupt processing starts following the end of the first non-maskable interrupt processing. The operation is not returned to the main routine.

**5.2.1 Operation**

If the non-maskable interrupt is generated by NMI input, the CPU performs the following processing, and transfers control to the handler routine:

- (1) Saves the restored PC to FEPC.
- (2) Saves the current PSW to FEPSW.
- (3) Writes exception code 0010H to the higher half-word (FECC) of ECR.
- (4) Sets the NP and ID bits of PSW and clears the EP bit.
- (5) Loads the handler address (00000010H) of the non-maskable interrupt routine to the PC, and transfers control.

Figure 5-1 illustrates how the non-maskable interrupt is processed.

**Figure 5-1. Non-Maskable Interrupt Processing**

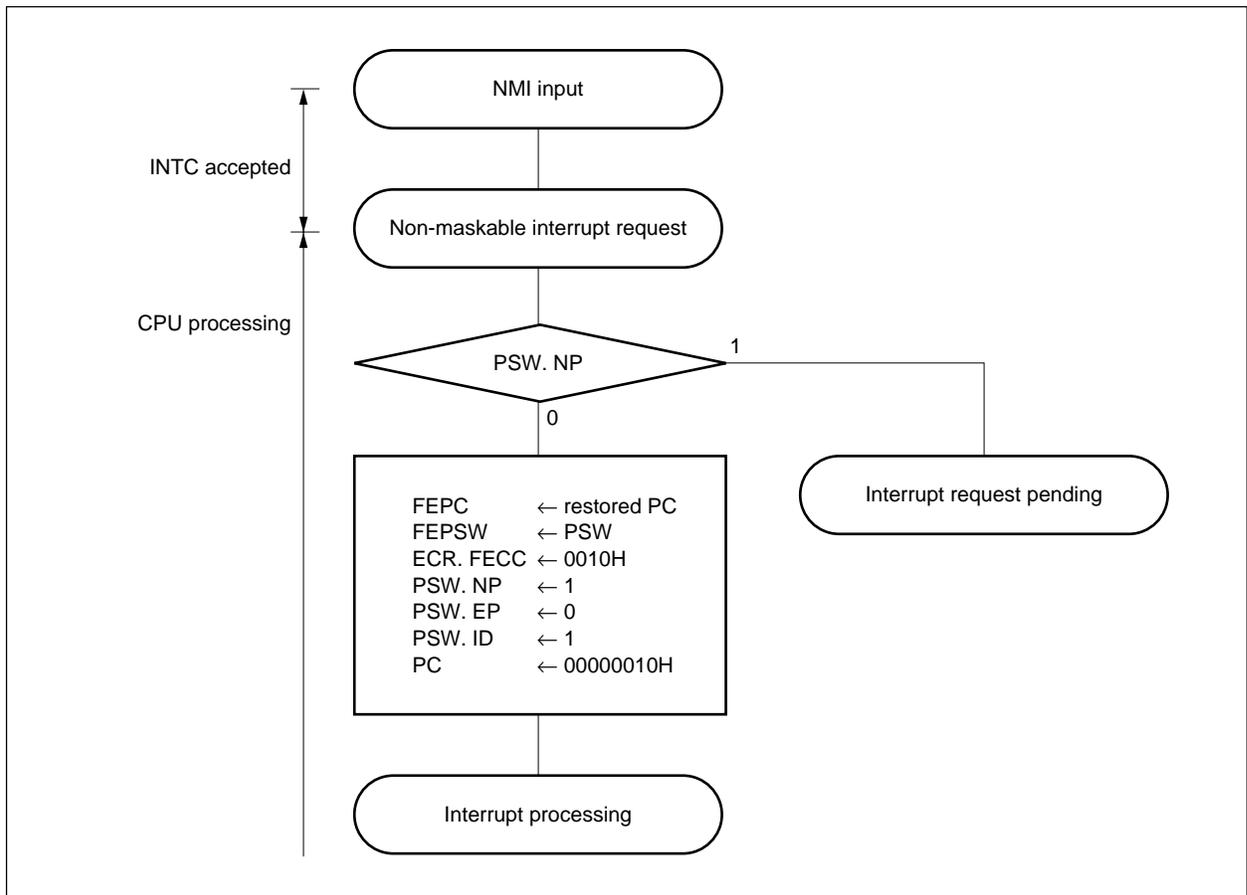
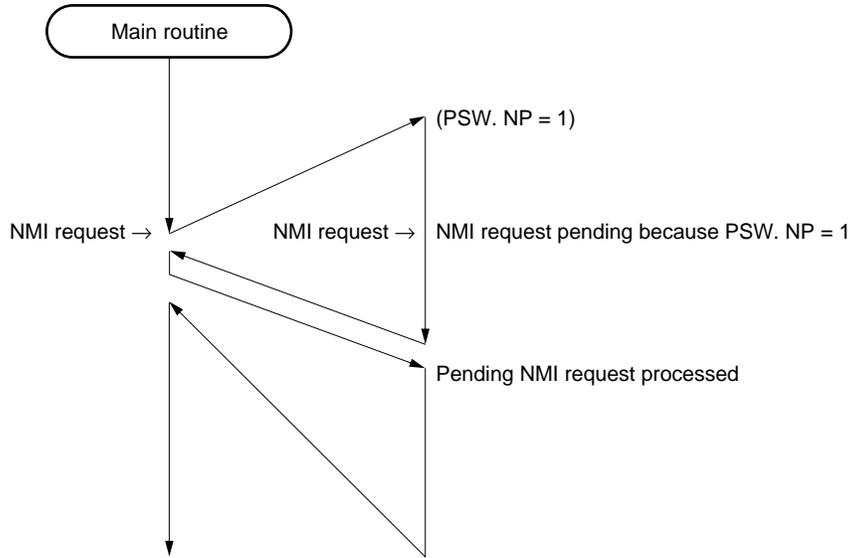
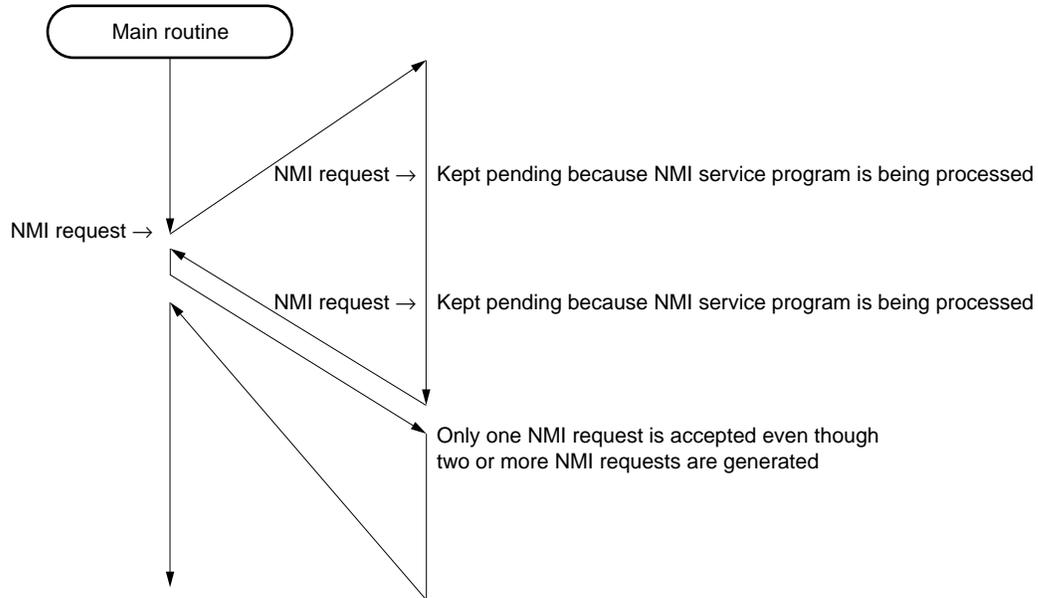


Figure 5-2. Accepting Non-Maskable Interrupt Request

(a) If a new NMI request is generated while an NMI service routine is executing:



(b) If a new NMI request is generated twice while an NMI service routine is executing:



### 5.2.2 Restore

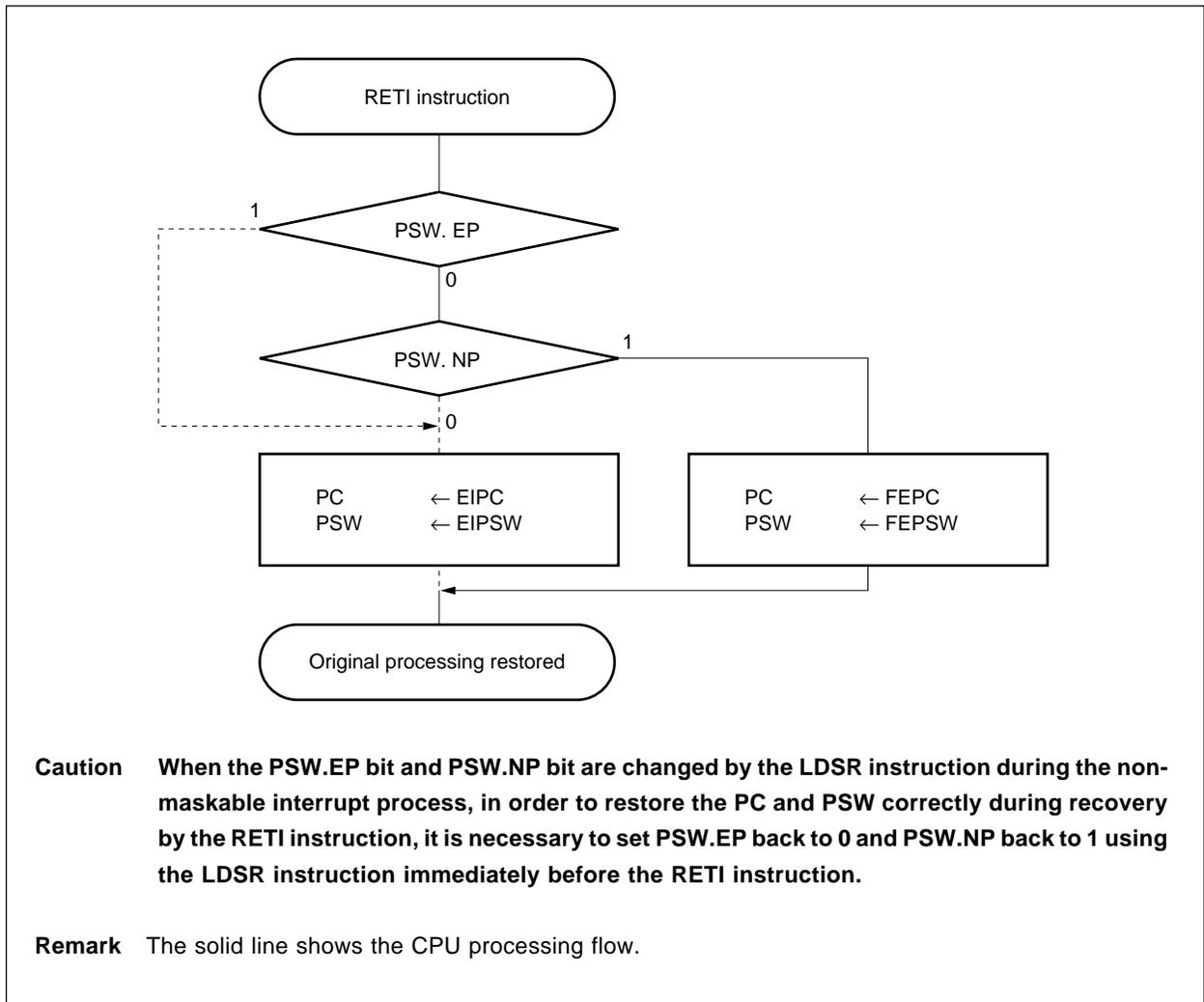
Execution is restored from the non-maskable interrupt processing by the RETI instruction.

When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

- (1) Restores the values of PC and PSW from FEPC and FEPSW, respectively, because the EP bit of PSW is 0 and the NP bit of PSW is 1.
- (2) Transfers control back to the restored PC address and PSW status.

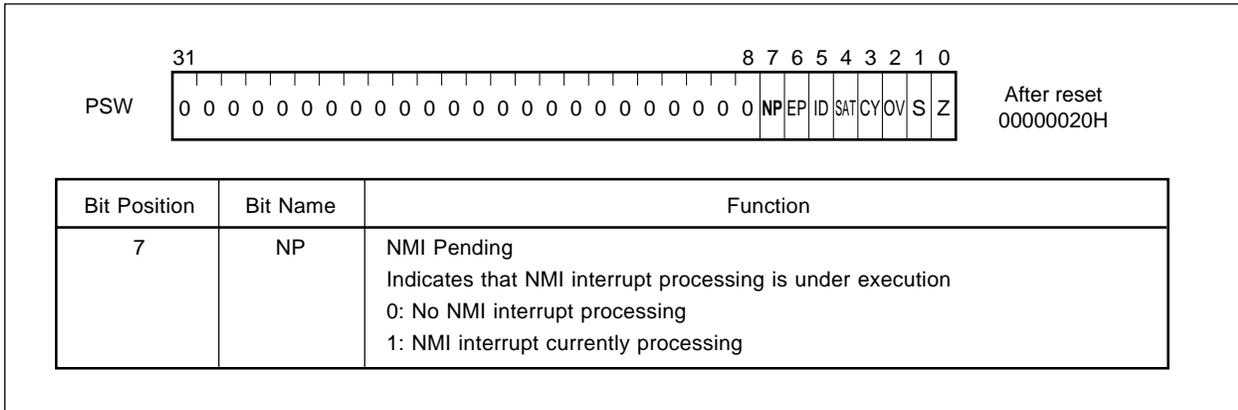
Figure 5-3 illustrates how the RETI instruction is processed.

**Figure 5-3. RETI Instruction Processing**



**5.2.3 Non-maskable interrupt status flag (NP)**

The NP flag is a status flag that indicates that non-maskable interrupt (NMI) processing is under execution. This flag is set when all the interrupts and requests have been accepted, and masks all interrupt requests and exceptions to prohibit multiple interrupts from being acknowledged.



**5.2.4 Noise elimination circuit of NMI pin**

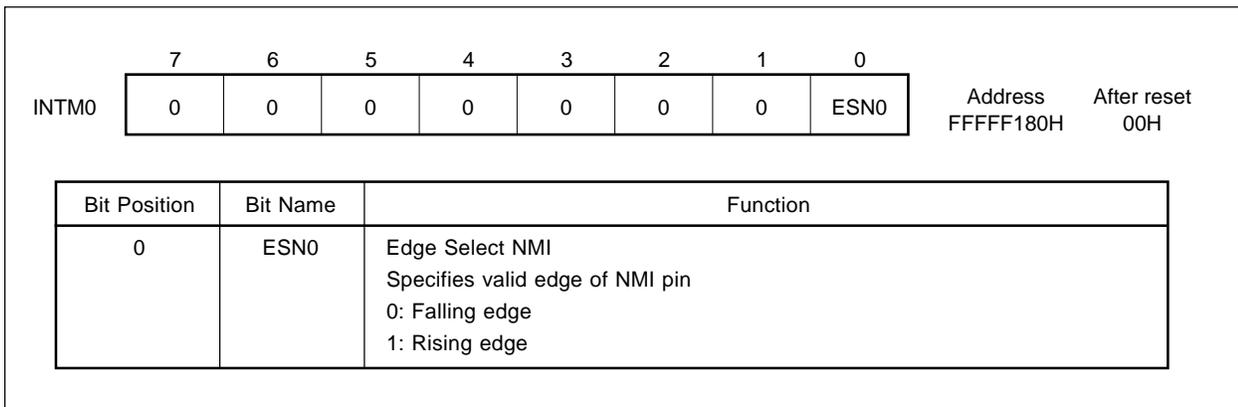
NMI pin noise is eliminated with analog delay. The delay time is 60 to 220 ns. The signal input that changes in less than this time period is not internally acknowledged.

NMI pin is used for canceling the software stop mode. In the software stop mode, noise elimination does not use system clock for noise elimination because the internal system clock is stopped.

**5.2.5 Edge detection function of NMI pin**

INTM0 is a register that specifies the valid edge of the non-maskable interrupt (NMI). The valid edge of NMI can be specified as the rising or falling edge by the ESN0 bit of this register.

This register can be read or written in 8- or 1-bit units.



### 5.3 Maskable Interrupts

Maskable interrupt requests can be masked by interrupt control registers. The V854 has 31 maskable interrupt sources.

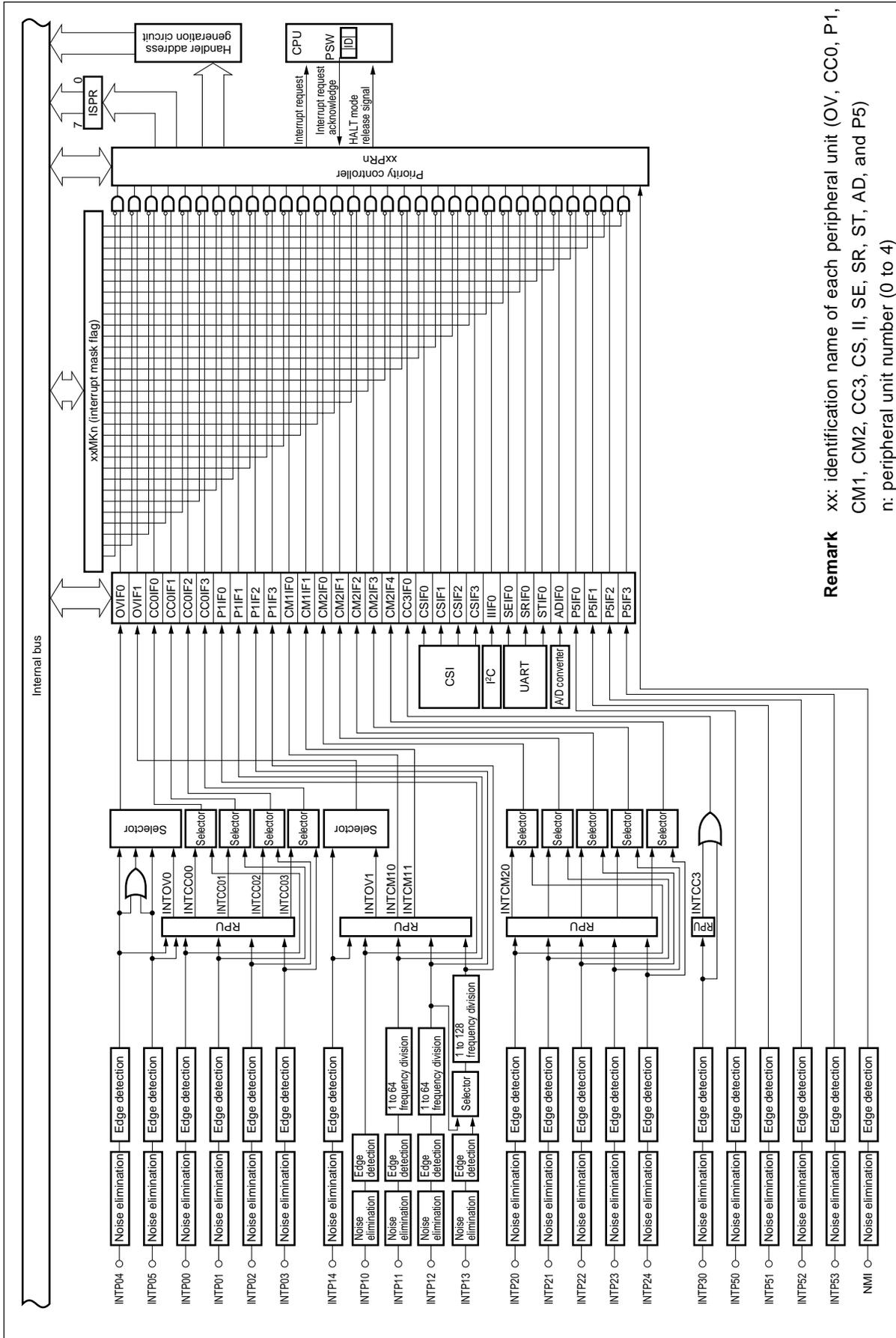
If two or more maskable interrupt requests are generated at the same time, they are accepted according to the default priority. In addition to the default priority, eight levels of priorities can be specified by using the interrupt control registers, allowing programmable priority control.

When an interrupt request has been acknowledged, the acceptance of other maskable interrupts is disabled and the interrupt disabled (DI) status is set.

When the EI instruction is executed in an interrupt processing routine, the interrupt enabled (EI) status is set which enables interrupts having a higher priority to immediately interrupt the current service routine in progress. Note that only interrupts with a higher priority will have this capability; interrupts with the same priority level cannot be nested.

To use multiple interrupts, it is necessary to save EIPC and EIPSW to memory or a register before executing the EI instruction, and restore EIPC and EIPSW to the original values by executing the DI instruction before the RETI instruction.

Figure 5-4. Block Diagram of Maskable Interrupt



**Remark** xx: identification name of each peripheral unit (OV, CC0, P1, CM1, CM2, CC3, CS, II, SE, SR, ST, AD, and P5)  
 n: peripheral unit number (0 to 4)

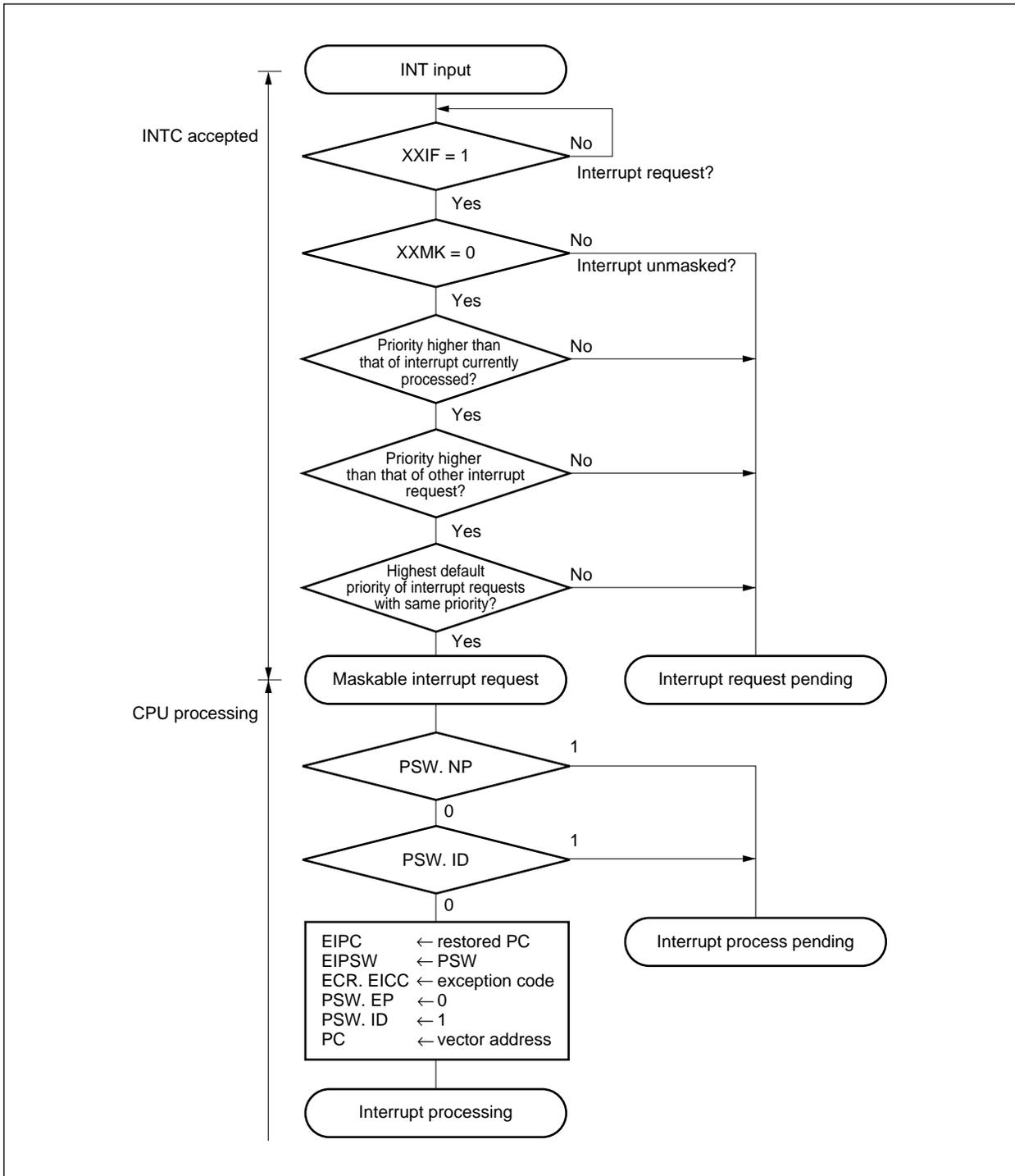
### 5.3.1 Operation

If a maskable interrupt occurs, the CPU performs the following processing, and transfers control to a handler routine:

- (1) Saves the restored PC to EIPC.
- (2) Saves the current PSW to EIPSW.
- (3) Writes an exception code to the lower half-word of ECR (EICC).
- (4) Sets the ID bit of PSW and clears the EP bit.
- (5) Loads the corresponding handler address to the PC, and transfers control.

Figure 5-5 illustrates how the maskable interrupts are processed.

Figure 5-5. Maskable Interrupt Processing



The INT input masked by the interrupt controllers and the INT input that occurs while the other interrupt is being processed (when PSW.NP = 1 or PSW.ID = 1) are internally pending by the interrupt controller. When the interrupts are unmasked, or when PSW.NP = 0 and PSW.ID = 0 by using the RETI and LDSR instructions, the pending INT input starts the new maskable interrupt processing.

### 5.3.2 Restore

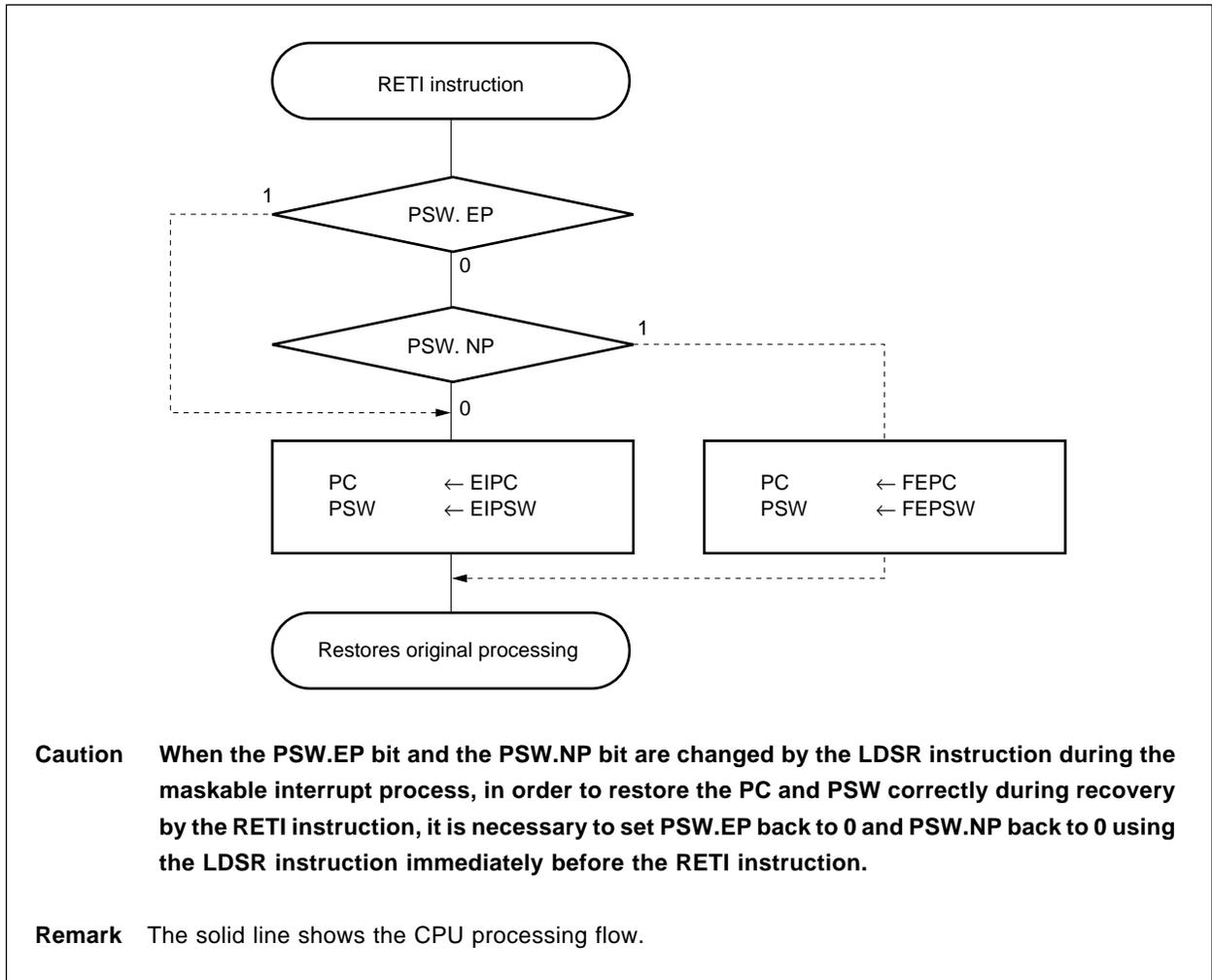
To restore execution from the maskable interrupt processing, the RETI instruction is used.

When the RETI instruction is executed, the CPU performs the following steps, and transfers control to the address of the restored PC.

- (1) Restores the values of PC and PSW from EIPC and EIPSW because the EP bit of PSW is 0 and the NP bit of PSW is 0.
- (2) Transfers control to the restored PC address and PSW status.

Figure 5-6 illustrates the processing of the RETI instruction.

**Figure 5-6. RETI Instruction Processing**



### 5.3.3 Priorities of maskable interrupts

The V854 provides multiple interrupt service that accepts an interrupt while servicing another interrupt. Multiple interrupts can be controlled by priority levels.

There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels that are specified by the interrupt priority level specification bits (xxPRn0 to xxPRn2) of the interrupt control register (xxICn). When two or more interrupts having the same priority level specified by the xxPRn0 to xxPRn2 are generated at the same time, the control based on default priority levels services them in the order of the priority level allocated to each interrupt request type (default priority level) beforehand. For more information refer to **Table 5-1**. The programmable priority control customizes interrupt requests into eight levels by setting the priority level specification flag.

The relation between the programmable priority levels and default priority levels is as follows.

Programmable priority levels > Default priority levels

Note that when an interrupt is acknowledged, the ID flag of PSW is automatically set to "1". Therefore, when multiple interrupts are to be used, clear the ID flag to "0" beforehand (for example, by placing the EI instruction into the interrupt service program) to set the interrupt enable mode.

**Remarks** xx : Each peripheral unit identifier (OV, CC0, P1, CM1, CM2, CC3, CS, II, SE, SR, ST, AD, P5)  
n : Peripheral unit number (0 to 4)

Figure 5-7. Example of Interrupt Nesting Process (1/2)

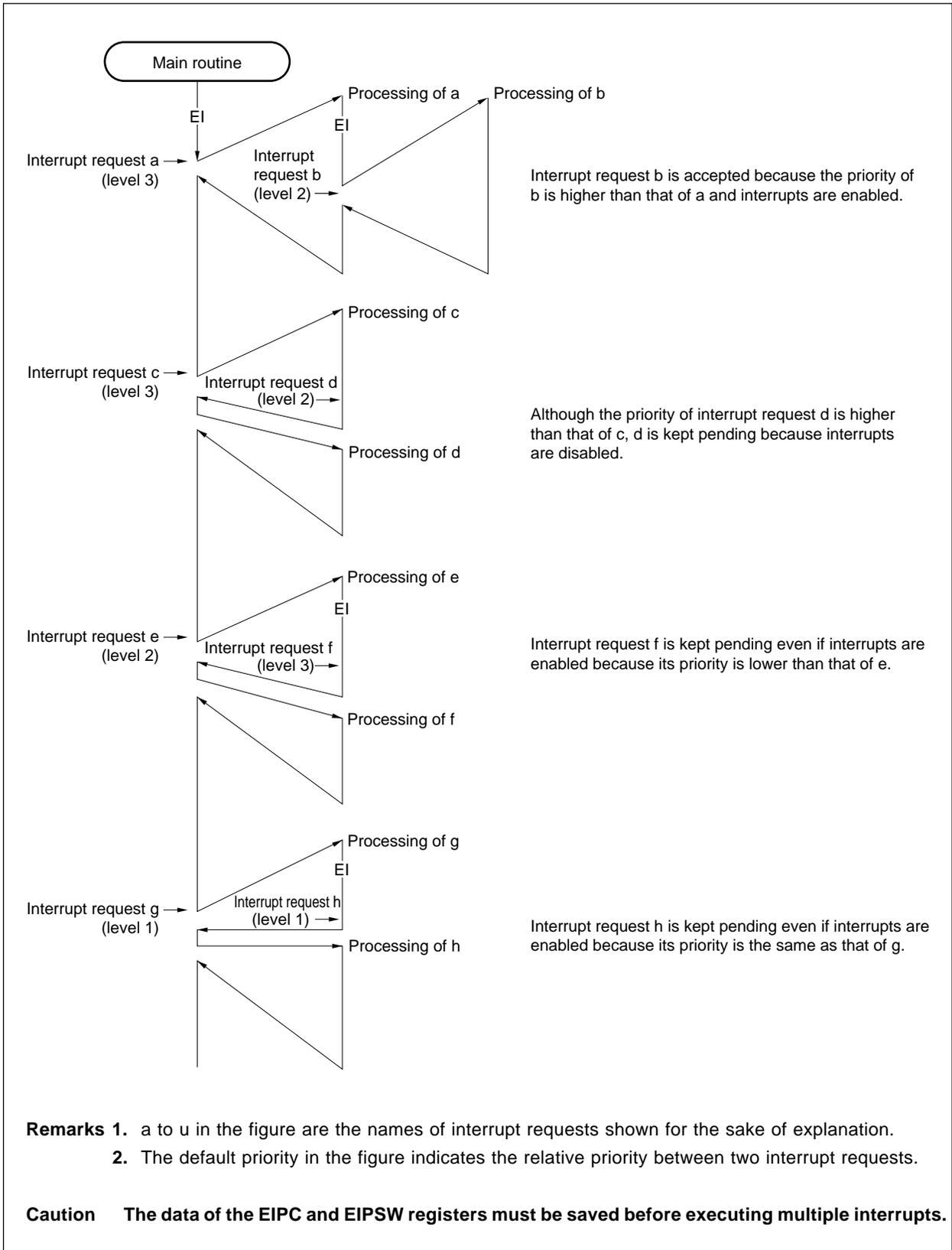


Figure 5-7. Example of Interrupt Nesting Process (2/2)

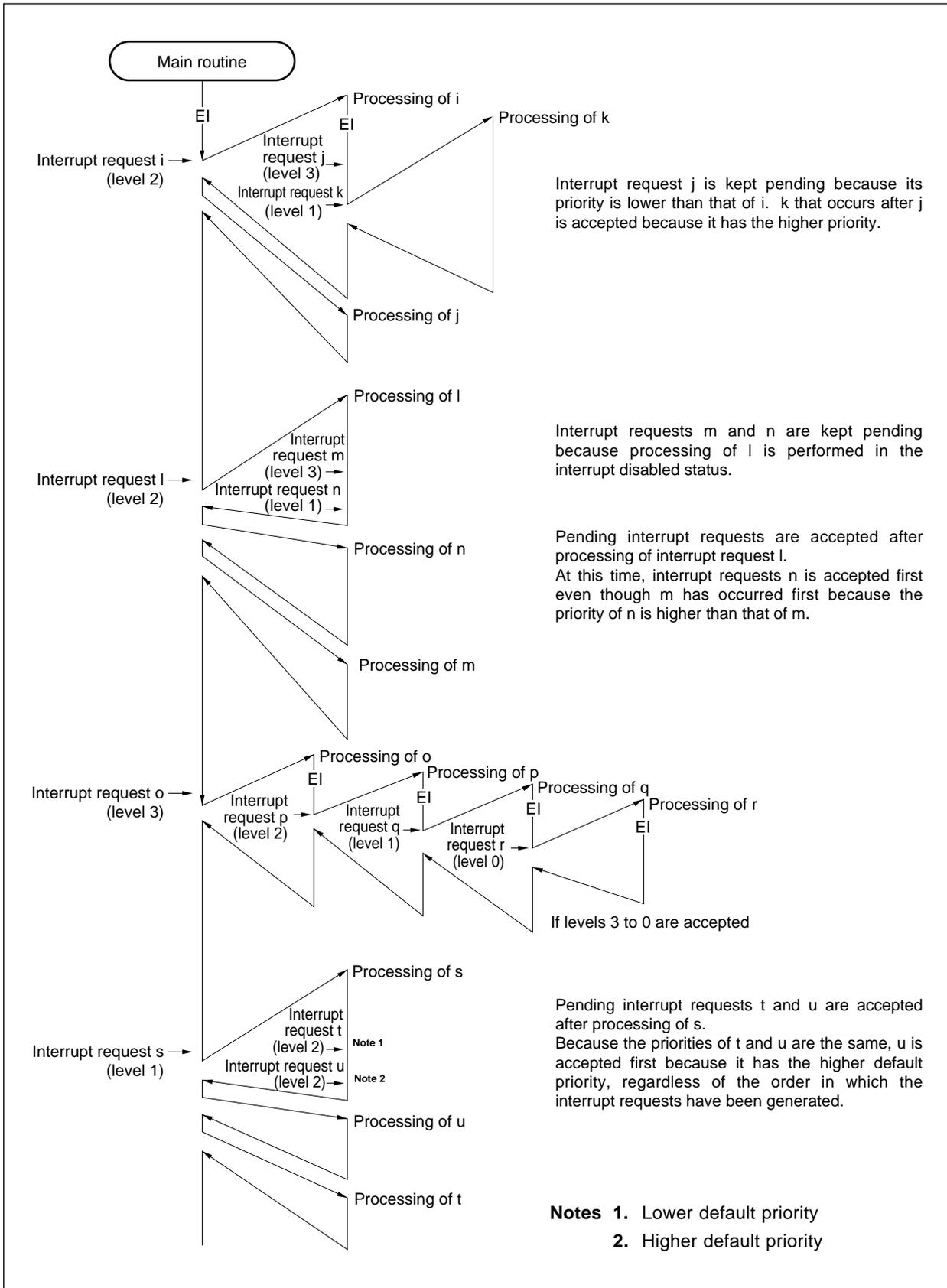
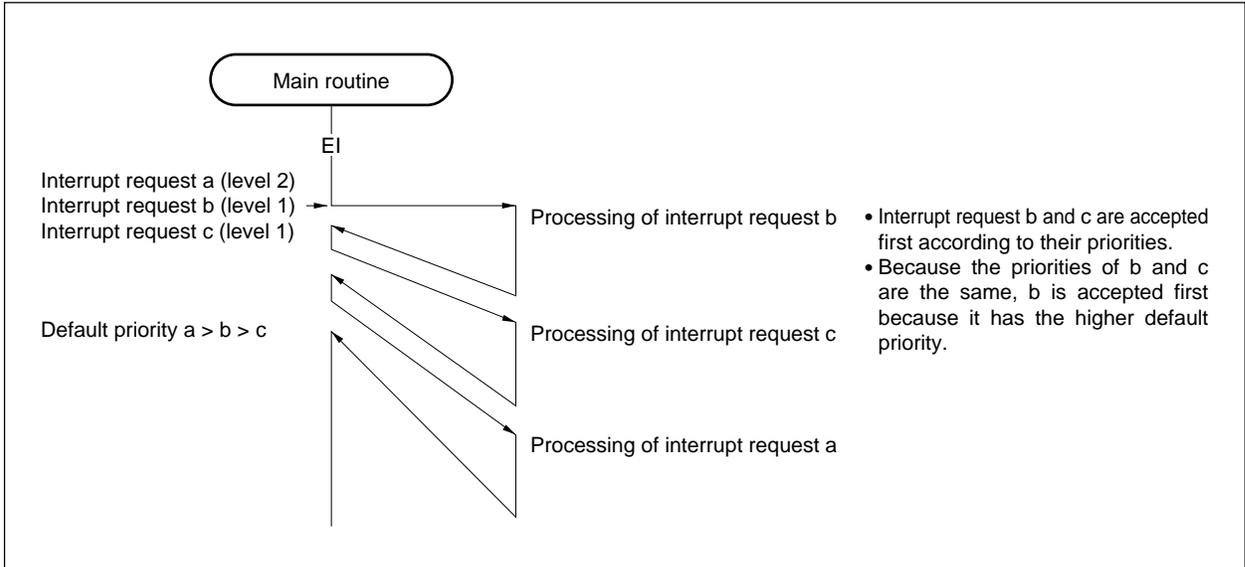


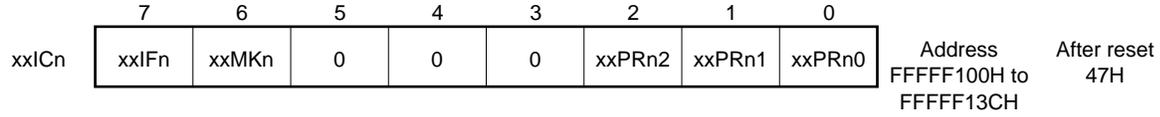
Figure 5-8. Example of Processing Interrupt Requests Simultaneously Generated



**5.3.4 Interrupt control register (xxICn)**

An interrupt control register is assigned to each maskable interrupt and sets the control conditions for each maskable interrupt request.

The interrupt control register can be read/written in 8- or 1-bit units.



Bit Position	Bit Name	Function																																				
7	xxIFn	Interrupt Request Flag Interrupt request flag 0: Interrupt request not issued 1: Interrupt request issued xxIFn flag is automatically reset by hardware when interrupt request is accepted.																																				
6	xxMKn	Mask Flag Interrupt mask flag 0: Enables interrupt processing 1: Disables interrupt processing (pending)																																				
2 to 0	xxPRn2 to xxPRn0	Priority Specifies eight levels of priorities for each interrupt <table border="1" style="border-collapse: collapse; margin: 10px auto; width: 80%;"> <thead> <tr> <th style="text-align: center;">xxPRn2</th> <th style="text-align: center;">xxPRn1</th> <th style="text-align: center;">xxPRn0</th> <th style="text-align: center;">Interrupt priority specification bit</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">Specifies level 0 (highest)</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Specifies level 1</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">Specifies level 2</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Specifies level 3</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">Specifies level 4</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Specifies level 5</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">Specifies level 6</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Specifies level 7 (lowest)</td> </tr> </tbody> </table>	xxPRn2	xxPRn1	xxPRn0	Interrupt priority specification bit	0	0	0	Specifies level 0 (highest)	0	0	1	Specifies level 1	0	1	0	Specifies level 2	0	1	1	Specifies level 3	1	0	0	Specifies level 4	1	0	1	Specifies level 5	1	1	0	Specifies level 6	1	1	1	Specifies level 7 (lowest)
xxPRn2	xxPRn1	xxPRn0	Interrupt priority specification bit																																			
0	0	0	Specifies level 0 (highest)																																			
0	0	1	Specifies level 1																																			
0	1	0	Specifies level 2																																			
0	1	1	Specifies level 3																																			
1	0	0	Specifies level 4																																			
1	0	1	Specifies level 5																																			
1	1	0	Specifies level 6																																			
1	1	1	Specifies level 7 (lowest)																																			

**Remark** xx : identification name of each peripheral unit (OV, CC0, P1, CM1, CM2, CC3, CS, II, SE, SR, ST, AD, P5)  
 n : peripheral unit number (0 to 4)

Address and bit of each interrupt control register is as follows:

Address	Register	Bit							
		7	6	5	4	3	2	1	0
FFFFF100H	OVIC0	OVIF0	OVMK0	0	0	0	OVPR02	OVPR01	OVPR00
FFFFF102H	OVIC1	OVIF1	OVMK1	0	0	0	OVPR12	OVPR11	OVPR10
FFFFF104H	CC0IC0	CC0IF0	CC0MK0	0	0	0	CC0PR02	CC0PR01	CC0PR00
FFFFF106H	CC0IC1	CC0IF1	CC0MK1	0	0	0	CC0PR12	CC0PR11	CC0PR10
FFFFF108H	CC0IC2	CC0IF2	CC0MK2	0	0	0	CC0PR22	CC0PR21	CC0PR20
FFFFF10AH	CC0IC3	CC0IF3	CC0MK3	0	0	0	CC0PR32	CC0PR31	CC0PR30
FFFFF10CH	P1IC0	P1IF0	P1MK0	0	0	0	P1PR02	P1PR01	P1PR00
FFFFF10EH	P1IC1	P1IF1	P1MK1	0	0	0	P1PR12	P1PR11	P1PR10
FFFFF110H	P1IC2	P1IF2	P1MK2	0	0	0	P1PR22	P1PR21	P1PR20
FFFFF112H	P1IC3	P1IF3	P1MK3	0	0	0	P1PR32	P1PR31	P1PR30
FFFFF114H	CM1IC0	CM1IF0	CM1MK0	0	0	0	CM1PR02	CM1PR01	CM1PR00
FFFFF116H	CM1IC1	CM1IF1	CM1MK1	0	0	0	CM1PR12	CM1PR11	CM1PR10
FFFFF118H	CM2IC0	CM2IF0	CM2MK0	0	0	0	CM2PR02	CM2PR01	CM2PR00
FFFFF11AH	CM2IC1	CM2IF1	CM2MK1	0	0	0	CM2PR12	CM2PR11	CM2PR10
FFFFF11CH	CM2IC2	CM2IF2	CM2MK2	0	0	0	CM2PR22	CM2PR21	CM2PR20
FFFFF11EH	CM2IC3	CM2IF3	CM2MK3	0	0	0	CM2PR32	CM2PR31	CM2PR30
FFFFF120H	CM2IC4	CM2IF4	CM2MK4	0	0	0	CM2PR42	CM2PR41	CM2PR40
FFFFF122H	CC3IC0	CC3IF0	CC3MK0	0	0	0	CC3PR02	CC3PR01	CC3PR00
FFFFF124H	CSIC0	CSIF0	CSMK0	0	0	0	CSPR02	CSPR01	CSPR00
FFFFF126H	CSIC1	CSIF1	CSMK1	0	0	0	CSPR12	CSPR11	CSPR10
FFFFF128H	CSIC2	CSIF2	CSMK2	0	0	0	CSPR22	CSPR21	CSPR20
FFFFF12AH	CSIC3	CSIF3	CSMK3	0	0	0	CSPR32	CSPR31	CSPR30
FFFFF12CH	IIC0	IIF0	IIMK0	0	0	0	IIPR02	IIPR01	IIPR00
FFFFF12EH	SEIC0	SEIF0	SEMK0	0	0	0	SEPR02	SEPR01	SEPR00
FFFFF130H	SRIC0	SRIF0	SRMK0	0	0	0	SRPR02	SRPR01	SRPR00
FFFFF132H	STIC0	STIF0	STMK0	0	0	0	STPR02	STPR01	STPR00
FFFFF134H	ADIC0	ADIF0	ADMK0	0	0	0	ADPR02	ADPR01	ADPR00
FFFFF136H	P5IC0	P5IF0	P5MK0	0	0	0	P5PR02	P5PR01	P5PR00
FFFFF138H	P5IC1	P5IF1	P5MK1	0	0	0	P5PR12	P5PR11	P5PR10
FFFFF13AH	P5IC2	P5IF2	P5MK2	0	0	0	P5PR22	P5PR21	P5PR20
FFFFF13CH	P5IC3	P5IF3	P5MK3	0	0	0	P5PR32	P5PR31	P5PR30

### 5.3.5 In-service priority register (ISPR)

This register holds the priority level of the maskable interrupt currently accepted. When an interrupt request is accepted, the bit of this register corresponding to the priority level of that interrupt is set to 1 and remains set while the interrupt is serviced.

When the RETI instruction is executed, the bit corresponding to the interrupt request having the highest priority is automatically reset to 0 by hardware. However, it is not reset when execution is returned from non-maskable processing or exception processing.

This register can be only read in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
ISPR	ISPR7	ISPR6	ISPR5	ISPR4	ISPR3	ISPR2	ISPR1	ISPR0	Address	After reset
									FFFFF166H	00H

Bit Position	Bit Name	Function
7 to 0	ISPR7 to ISPR0	In-Service Priority Flag Indicates priority of interrupt currently accepted 0: Interrupt request with priority n not accepted 1: Interrupt request with priority n accepted

**Remark** n: 0 to 7 (priority level)

### 5.3.6 Maskable interrupt status flag (ID)

The interrupt disable status flag (ID) of the PSW controls the enabling and disabling of maskable interrupt requests.

	31	8	7	6	5	4	3	2	1	0			
PSW	0	0	0	0	0	0	0	0	0	0	0	NP EP ID SAT CY OV S Z	After reset
													00000020H

Bit Position	Bit Name	Function
5	ID	Interrupt Disable Indicates enabling/disabling of maskable interrupt processing. 0: Maskable interrupt accepting enabled 1: Maskable interrupt accepting disabled (pending) It is set to 1 by the DI instruction and reset to 0 by the EI instruction. Its value is also modified by the RETI instruction or LDSR instruction when referencing the PSW. Non-maskable interrupt and exceptions are acknowledged regardless of this flag. When a maskable interrupt is accepted, ID flag is automatically set to 1 by hardware. The interrupt request generated during the accepting disabled period (ID:1) is accepted when the xxIFn bit of xxICn is set to 1, and ID flag is reset to 0.

**5.3.7 Noise elimination**

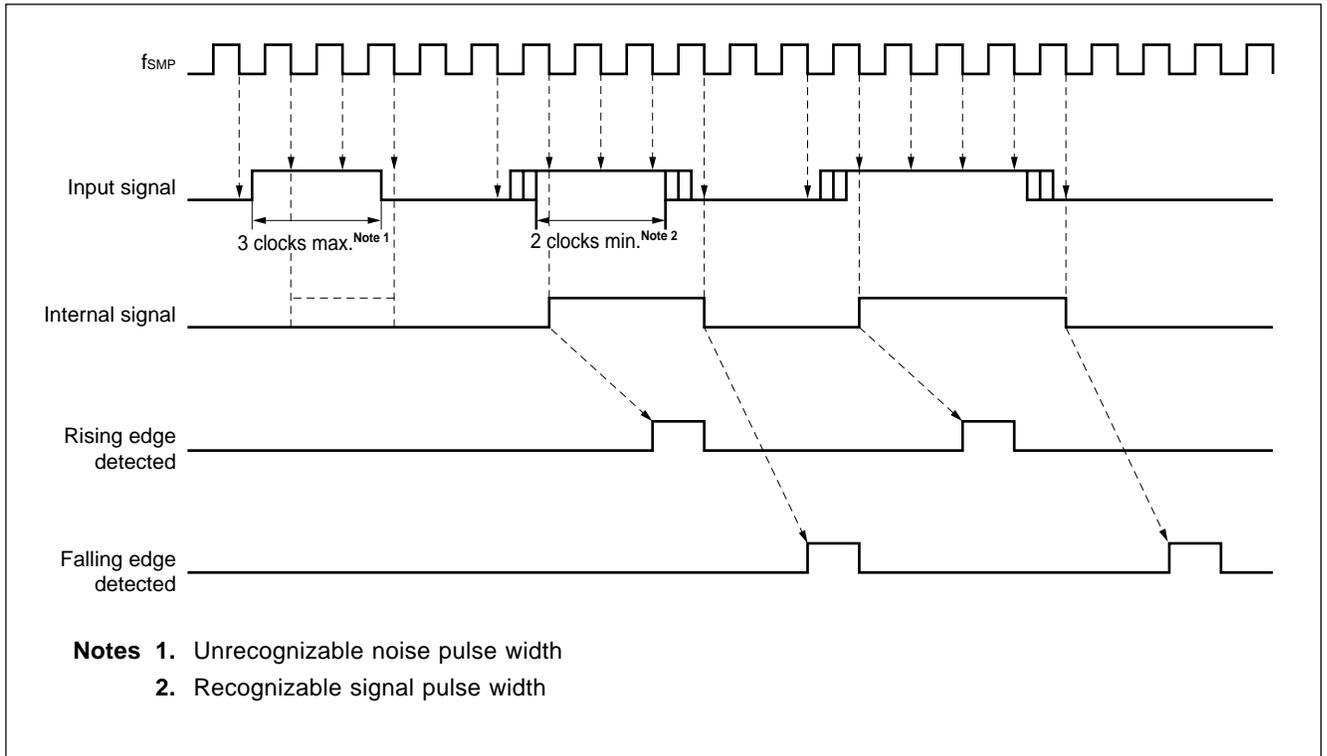
INTP, TI, TCLR, and ADTRG pins are attached with respective digital noise elimination circuit. Thereby, the input levels of these pins are sampled at each sampling clock ( $f_{SMP}$ ). As a result, if the same level cannot be detected three times consecutively, the input pulse is eliminated as a noise.

The noise elimination time for each pin is shown below. The sampling clock of INTP30 pin can be selected from  $\phi$ ,  $\phi/64$ ,  $\phi/128$ , or  $\phi/256$ . For the settings, write values to INTM7 register (refer to 5.3.8 (2) (a) External interrupt request register 7 (INTM7)).

Pin	$f_{SMP}$	Noise Elimination Time
INTP00 to INTP03	$\phi$	2 to 3 system clocks
TCLR0/INTP04	$\phi$	
TI0/INTP05	$\phi$	
INTP10 to INTP13	$\phi$	
TI1/INTP14	$\phi$	
TI20/INTP20 to TI24/INTP24	$\phi$	
ADTRG	$\phi$	
INTP30	$\phi$	2 to 3 system clocks
	$\phi/64$	128 to 192 system clocks
	$\phi/128$	256 to 384 system clocks
	$\phi/256$	512 to 768 system clocks

**Remark**  $f_{SMP}$ : Sampling clock  
 $\phi$  : Internal system clock

**Figure 5-9. Example of Noise Elimination Timing**



- Cautions**
1. In the case that the input pulse width is two to three sampling clocks, it is indefinite whether the input pulse is detected as a valid edge or eliminated as a noise.
  2. To securely detect the level as a pulse, input the same level at least three sampling clocks.
  3. When noise is generated in synchronization with sampling, its may recognized as noise. In such cases, attach a filter to the input pin to eliminate the noise.

### 5.3.8 Edge detection function

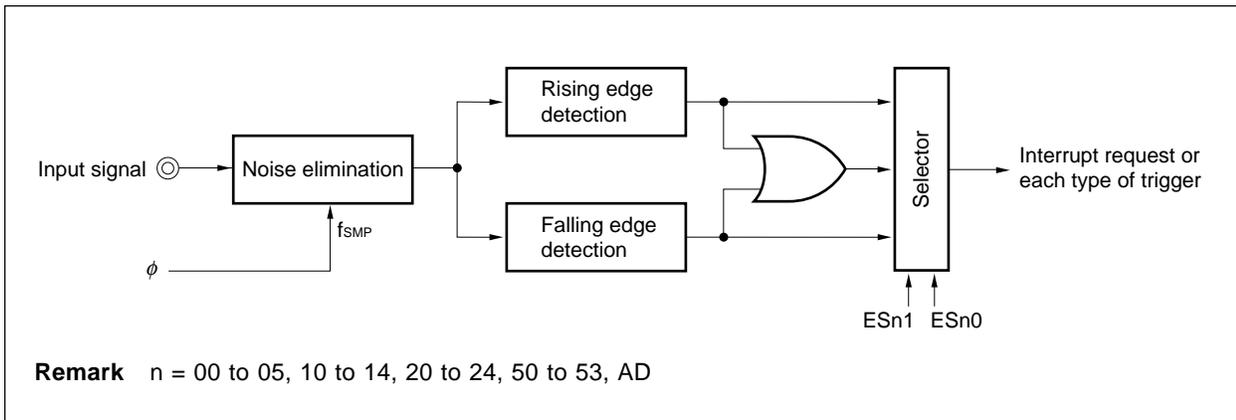
#### (1) Edge detection of INTP pin (except INTP30 pin), TI pin, TCLR pin, ADTRG pin

These pins can be programmably selected. The valid edge can be selected from the followings:

- Rising edge
- Falling edge
- Both rising and falling edges

The detected INTP signal becomes an interrupt source or capture trigger.

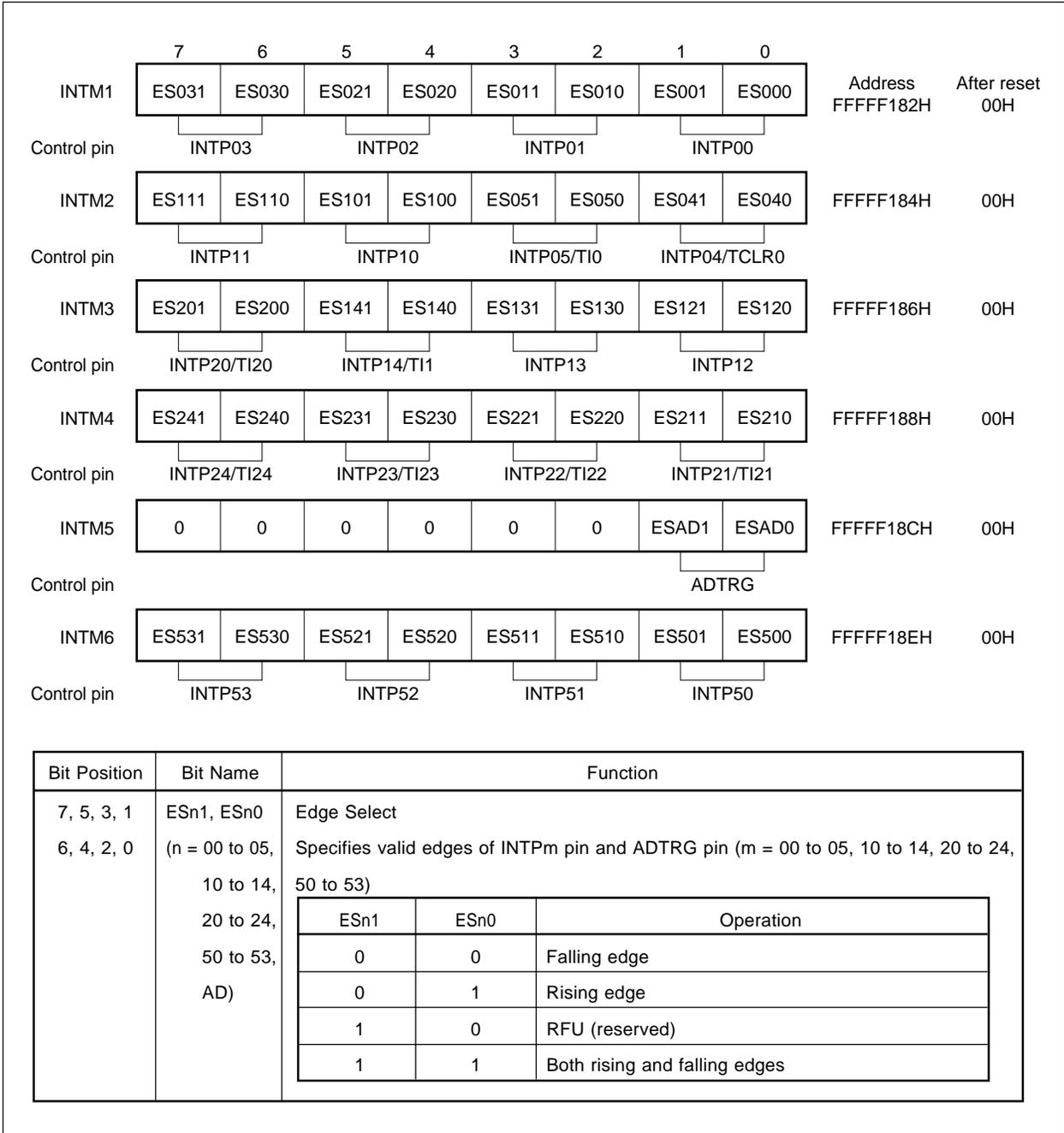
The block diagram of the edge detection of these pins is shown below.



**(2) External interrupt mode registers 1 to 6 (INTM1 to INTM6)**

These registers specify the valid edges of external interrupt requests or each type of trigger that are input from external pins.

The valid edge of each pin can be specified to be the rising, falling, and both rising and falling edges. Both the registers can be read/written in 8- or 1-bit units.



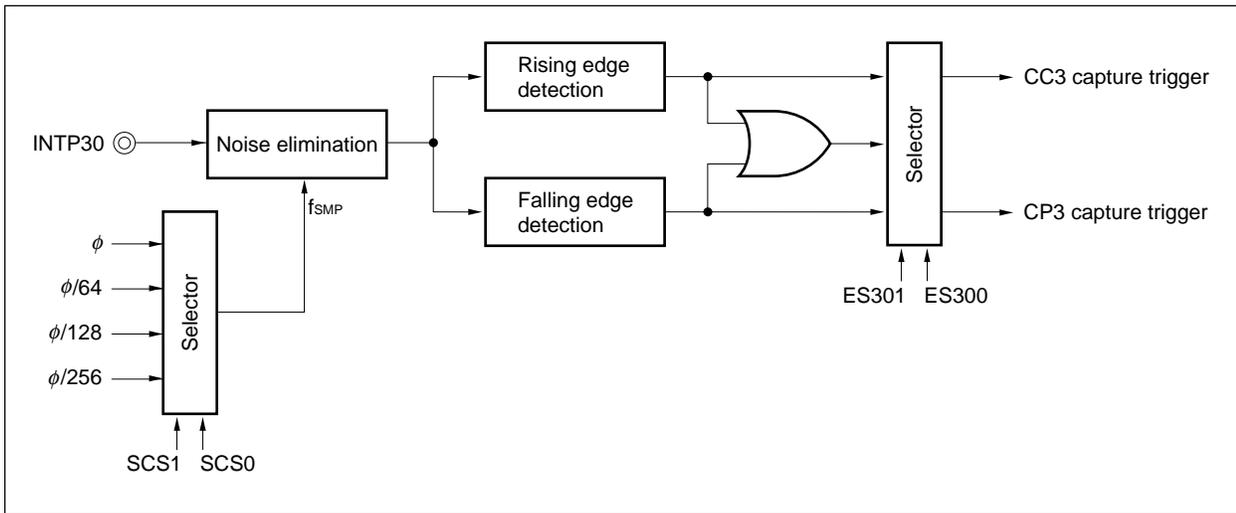
**(3) Edge detection of INTP30 pin**

To set the valid edge of INTP30 pin, write values to INTM7 register. The valid edge can be selected from the followings.

- Rising edge
- Falling edge
- Both rising and falling edges

The edge detected INTP30 signal becomes the capture trigger of CC3 register and CP3 register of timer function. The triggers of CC3 register and CP3 register have reverse edges. However, when both edges are specified, either one trigger is valid.

The block diagram of the edge detection of INTP30 pin is shown below.



**(a) External interrupt mode register 7 (INTM7)**

This register specifies the sampling clock ( $f_{SMP}$ ) and the valid edge of digital noise elimination by INTP30 pins.

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
INTM7	ES301	ES300	0	0	0	0	SCS1	SCS0	Address FFFFFF18EH	After reset 00H

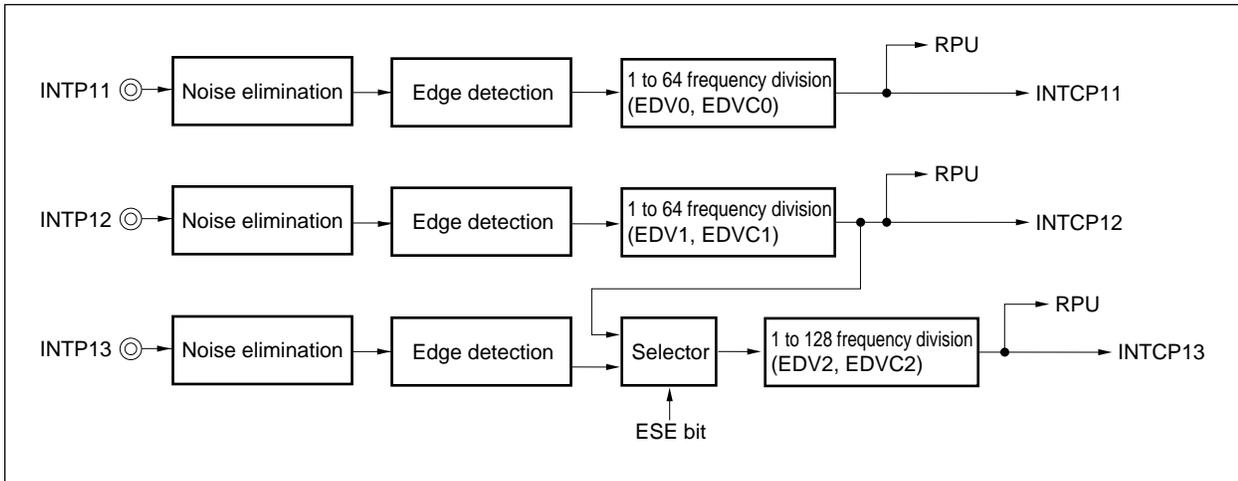
Bit Position	Bit Name	Function																									
7, 6	ES301, ES300	Edge Select Specifies valid edge of INTP30 pin <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">ES301</th> <th style="width: 10%;">ES300</th> <th style="width: 35%;">INTP30, CC3 Capture Trigger</th> <th style="width: 45%;">CP3 Capture Trigger</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Falling edge</td> <td>Rising edge</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Rising edge</td> <td>Falling edge</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Without selection edge</td> <td>Both rising and falling edges</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Both rising and falling edges</td> <td>Not captured</td> </tr> </tbody> </table> <p><b>Remark</b>     <math>\phi</math>: internal system clock</p>	ES301	ES300	INTP30, CC3 Capture Trigger	CP3 Capture Trigger	0	0	Falling edge	Rising edge	0	1	Rising edge	Falling edge	1	0	Without selection edge	Both rising and falling edges	1	1	Both rising and falling edges	Not captured					
ES301	ES300	INTP30, CC3 Capture Trigger	CP3 Capture Trigger																								
0	0	Falling edge	Rising edge																								
0	1	Rising edge	Falling edge																								
1	0	Without selection edge	Both rising and falling edges																								
1	1	Both rising and falling edges	Not captured																								
1, 0	SCS1, SCS0	Sampling Clock Select Specifies sampling clock <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">SCS1</th> <th style="width: 10%;">SCS0</th> <th style="width: 15%;">Sampling Clock (<math>f_{SMP}</math>)</th> <th style="width: 20%;">Pulse Width Eliminated as Noise</th> <th style="width: 45%;">Minimum Pulse Width Recognized as Signal</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td><math>\phi</math></td> <td><math>2/\phi</math></td> <td><math>3/\phi</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td><math>\phi/64</math></td> <td><math>128/\phi</math></td> <td><math>192/\phi</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td><math>\phi/128</math></td> <td><math>256/\phi</math></td> <td><math>384/\phi</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td><math>\phi/256</math></td> <td><math>512/\phi</math></td> <td><math>768/\phi</math></td> </tr> </tbody> </table> <p><b>Remark</b>     <math>\phi</math>: internal system clock</p>	SCS1	SCS0	Sampling Clock ( $f_{SMP}$ )	Pulse Width Eliminated as Noise	Minimum Pulse Width Recognized as Signal	0	0	$\phi$	$2/\phi$	$3/\phi$	0	1	$\phi/64$	$128/\phi$	$192/\phi$	1	0	$\phi/128$	$256/\phi$	$384/\phi$	1	1	$\phi/256$	$512/\phi$	$768/\phi$
SCS1	SCS0	Sampling Clock ( $f_{SMP}$ )	Pulse Width Eliminated as Noise	Minimum Pulse Width Recognized as Signal																							
0	0	$\phi$	$2/\phi$	$3/\phi$																							
0	1	$\phi/64$	$128/\phi$	$192/\phi$																							
1	0	$\phi/128$	$256/\phi$	$384/\phi$																							
1	1	$\phi/256$	$512/\phi$	$768/\phi$																							

### 5.3.9 Frequency divider

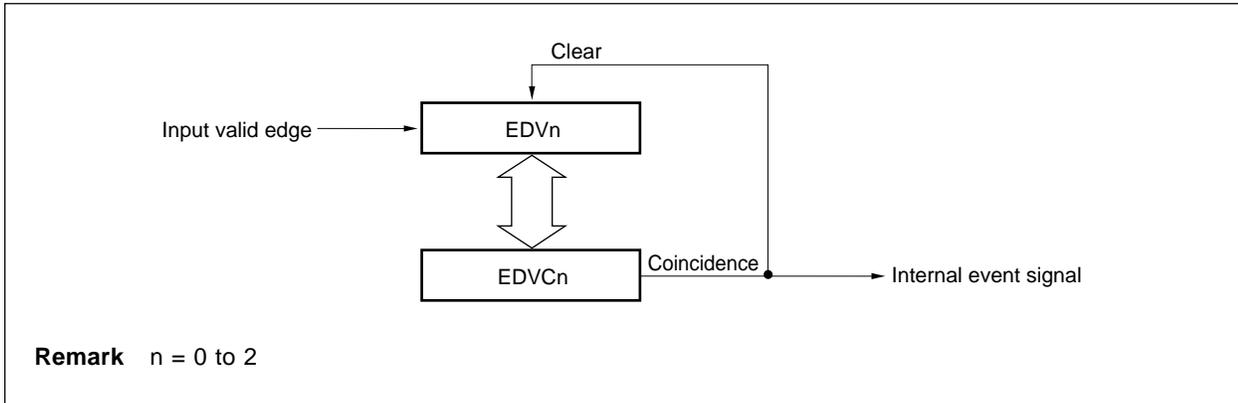
The V854 can internally divide the frequency of the signals input to P11 to P13 (INTP11 to INTP13) pins. The divided result becomes external interrupt request signal or timer capture trigger.

Frequency division ratio is set to event divide control register (EDVC) and comparing it with the value in event divide counter (EDV), and if they coincide, the value becomes the internal event signal, so that the frequency of the INTP signal is divided. 1 to 64 frequency division is possible for INTP11 and INTP12 signal. If INTP12 signal is divided 1 to 64, it can be further divided 1 to 128. However, INTP13 signal cannot be used when this function is used.

The block diagram of the INTP11 to INTP13 input is shown below.



The block diagram of the frequency divider is shown below.



#### (1) Event divide counter 0 to 2 (EDV0 to EDV2)

This register counts the valid edge of the INTP<sub>n</sub> input signal (n = 11 to 13). The EDV0 and EDV1 registers are configured with a 6-bit counter, and EDV2 register is configured with a 7-bit counter.

These registers are cleared at the following timings:

- Coincidence of the value in the event divide control register (EDVC<sub>n</sub>) and the count value
- Write to EDVC<sub>n</sub> register

**Remark** n = 0 to 2

These registers can be only read in 8/1-bit units. Edge inputs may not be counted when changing the specification of the valid edge of the INTM<sub>n</sub> register (n = 1 to 6).

	7	6	5	4	3	2	1	0		
EDV0	0	0	EDV05	EDV04	EDV03	EDV02	EDV01	EDV00	Address FFFFFF1B6H	After reset 00H
EDV1	0	0	EDV15	EDV14	EDV13	EDV12	EDV11	EDV10	FFFFFF1B8H	00H
EDV2	0	EDV26	EDV25	EDV24	EDV23	EDV22	EDV21	EDV20	FFFFFF1BAH	00H

**(2) Event divide control register 0 to 2 (EDVC0 to EDVC2)**

These registers set the frequency division ratio of the valid edge of INTP<sub>n</sub> input signal (n = 11 to 13). The set value as it is becomes the frequency division ratio. However, when 0 is set, the frequency division ratio is the maximum, that is, the frequency division of EDVC0 and EDVC1 is 64 and that of EDVC2 is 128.

Bits 7 and 6 of EDVC0 and bit 7 of EDVC2 are fixed to 0, and if 1 is written, it will be ignored.

These registers can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
EDVC0	0	0	EDVC05	EDVC04	EDVC03	EDVC02	EDVC01	EDVC00	Address FFFFFF1B0H	After reset 01H
EDVC1	0	0	EDVC15	EDVC14	EDVC13	EDVC12	EDVC11	EDVC10	FFFFFF1B2H	01H
EDVC2	0	EDVC26	EDVC25	EDVC24	EDVC23	EDVC22	EDVC21	EDVC20	FFFFFF1B4H	01H

**(3) Event select register (EVS)**

This register selects the signal to be input to EDV2 register.

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
EVS	0	0	0	0	0	0	0	ESE	Address FFFFFF1C0H	After reset 00H

Bit Position	Bit Name	Function
0	ESE	Event Select Selects input signal to EDV2 register 0: INTP13 signal 1: INTP12 signal frequency division result by EDVC1 register

## 5.4 Software Exception

A software exception is generated when the CPU executes the TRAP instruction, and can be always accepted.

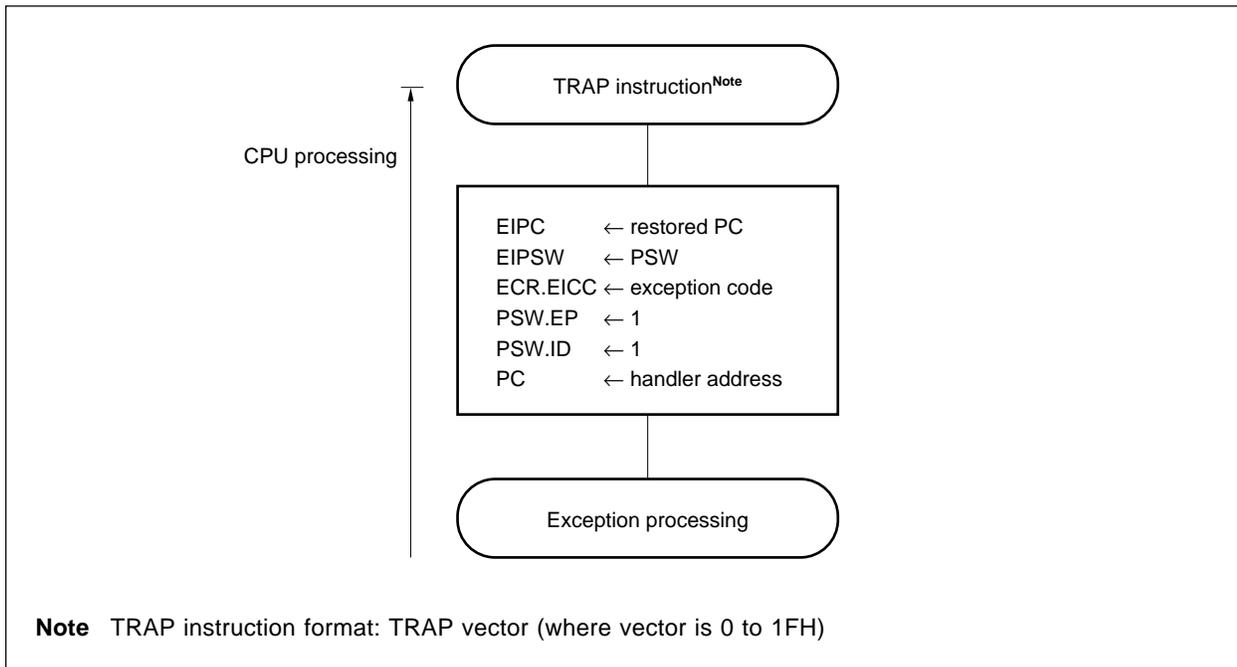
### 5.4.1 Operation

If a software exception occurs, the CPU performs the following processing, and transfers control to the handler routine:

- (1) Saves the restored PC to EIPC.
- (2) Saves the current PSW to EIPSW.
- (3) Writes an exception code to the lower 16 bits (EICC) of ECR (interrupt source).
- (4) Sets the EP and ID bits of PSW.
- (5) Loads the handler address (00000040H or 00000050H) of the software exception routine in the PC, and transfers control.

Figure 5-10 illustrates how a software exception is processed.

**Figure 5-10. Software Exception Processing**



The handler address is determined by the operand of the TRAP instruction (vector). If the vector is 0 to 0FH, the handler address is 00000040H; if the operand is 10H to 1FH, it is 00000050H.

### 5.4.2 Restore

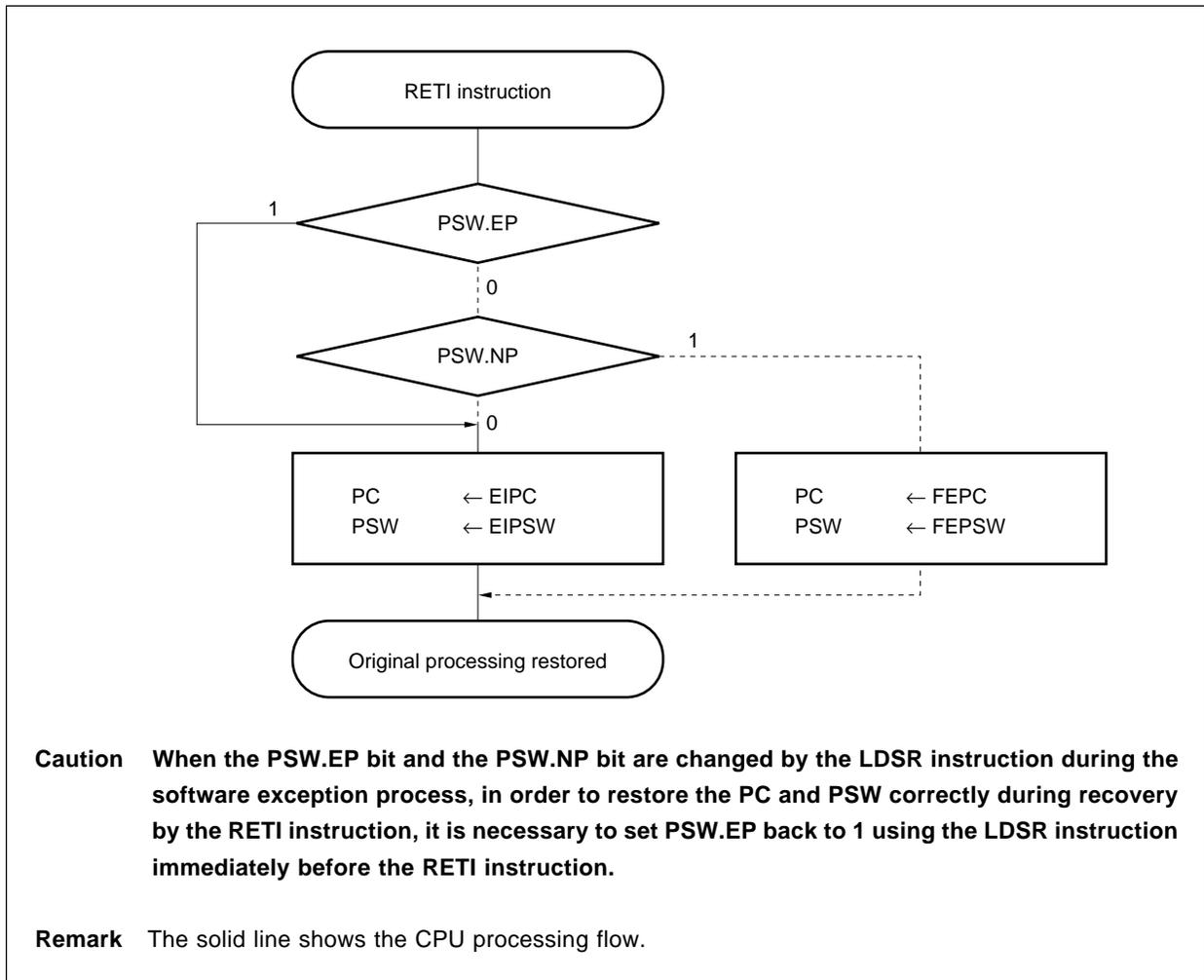
To restore or return execution from the software exception service routine, the RETI instruction is used.

When the RETI instruction is executed, the CPU performs the following steps, and transfers control to the address of the restored PC.

- (1) Restores the restored PC and PSW from EIPC and EIPSW because the EP bit of PSW is 1.
- (2) Transfers control to the restored PC address and PSW status.

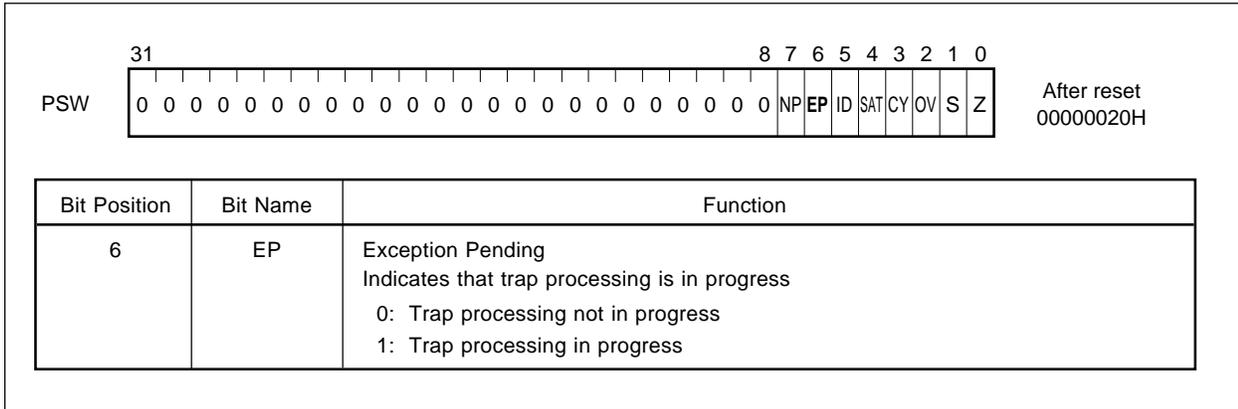
Figure 5-11 illustrates the processing of the RETI instruction.

**Figure 5-11. RETI Instruction Processing**



**5.4.3 Exception status flag (EP)**

The EP flag in PSW is a status flag used to indicate that trap processing is in progress. It is set when a trap occurs.





### 5.5.3 Restore

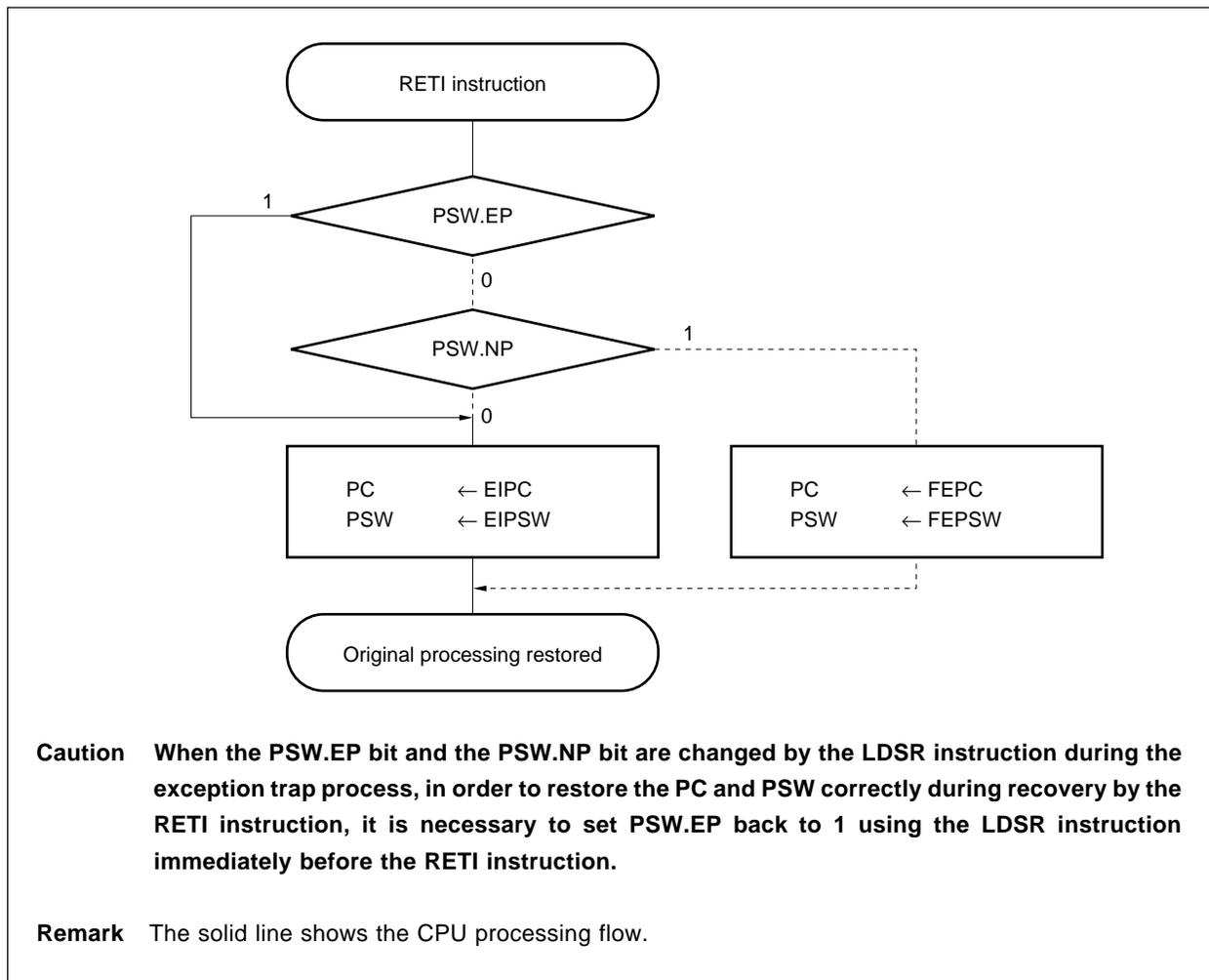
To restore or return execution from the exception TRAP, the RETI instruction is used.

When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

- (1) Restores the restored PC and PSW from EIPC and EIPSW because the EP bit of PSW is 1.
- (2) Transfers control to the restored PC address and PSW status.

Figure 5-13 illustrates the processing of the RETI instruction.

**Figure 5-13. RETI Instruction Processing**



## 5.6 Multiple interrupt processing

Multiple interrupt processing is a function that allows the nesting of interrupts. If a higher priority interrupt is generated and accepted, it will be allowed to stop a current interrupt service routine in progress. Execution of the original routine will resume once the higher priority interrupt routine is completed.

If an interrupt with a lower or equal priority is generated and a service routine is currently in progress, the later interrupt will be pended.

Multiple processing control of maskable interrupts is performed when in the state of interrupt acceptance (ID = 0). To perform maskable interrupt even in an interrupt processing routine, this control must be set in the state of acceptance (ID = 0). If a maskable interrupt acceptance or exception is generated during a service program of maskable interrupt or exception, EIPC and EIPSW must be saved.

The following example shows the procedure of interrupt nesting.

(1) To accept maskable interrupts in service routine

Service routine of maskable interrupt or exception

```

...
...
• Saves EIPC to memory or register
• Saves EIPSW to memory or register
• EI instruction (enables interrupt acceptance)
...
...
...
• DI instruction (disables interrupt acceptance)
• Restores saved value to EIPSW
• Restores saved value to EIPC
• RETI instruction
    
```

← Accepts maskable interrupt.

(2) To generate exception in service program

Service program of maskable interrupt or exception

```

...
...
• Saves EIPC to memory or register
• Saves EIPSW to memory or register
...
• TRAP instruction
• Illegal op code
...
• Restores saved value to EIPSW
• Restores saved value to EIPC
• RETI instruction
    
```

← Accepts exception such as TRAP instruction

← Accepts exception such as illegal op code

Priorities 0 to 7 (0 is the highest priority) can be programmed for each maskable interrupt request for multiple interrupt processing control. To set a priority level, write values to the xxPRn0 to xxPRn2 bits of the interrupt request control register (xxICn) corresponding to each maskable interrupt request. At system reset, the interrupt request is masked by the xxMKn bit, and the priority level is set to 7 by the xxPRn0 to xxPRn2 bits.

The priorities of maskable interrupts are as follows:

(High) Level 0 > Level 1 > Level 2 > Level 3 > Level 4 > Level 5 > Level 6 > Level 7 (Low)

Interrupt processing that has been suspended as a result of multiple interrupt processing is resumed after the interrupt processing of the higher priority has been completed and the RETI instruction has been executed. A pending interrupt request is accepted after the current interrupt processing has been completed and the RETI instruction has been executed.

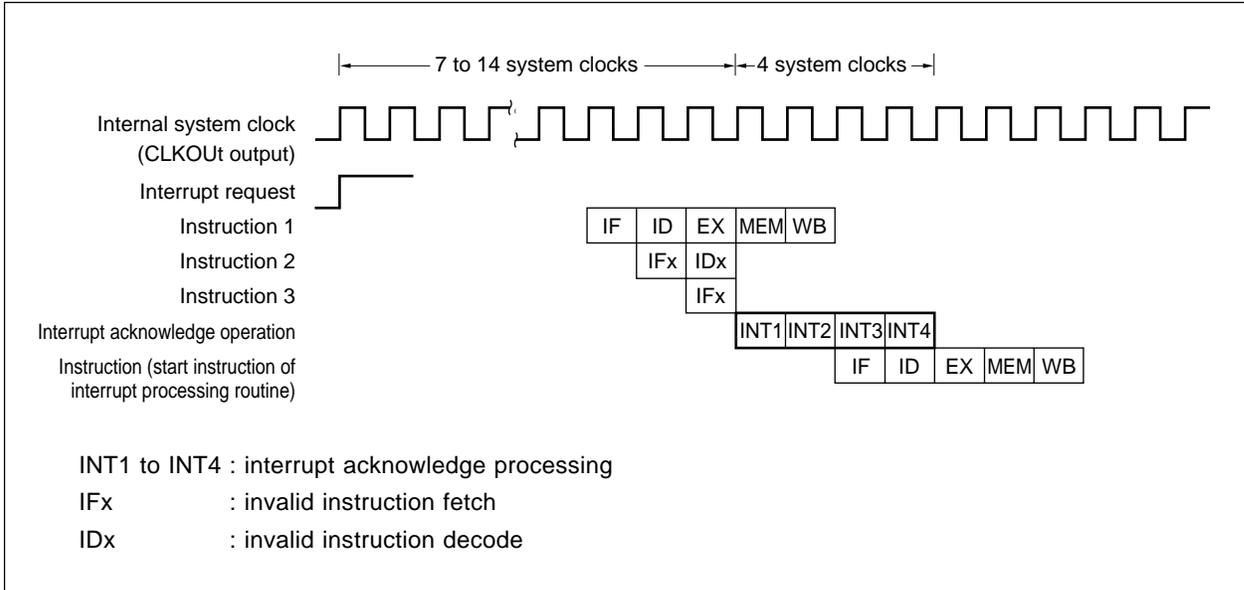
**Caution** In the non-maskable interrupt processing routine (time until the RETI instruction is executed), maskable interrupts are not accepted but are suspended.

**Remarks** xx: Each peripheral unit identifier (OV, CC0, P1, CM1, CM2, CC3, CS, II, SE, SR, ST, AD, P5)  
n: Peripheral unit number (0 to 4)

### 5.7 Interrupt Response Time

Interrupt Response Time from the interrupt request generation to the interrupt processing activation is as follows:

**Figure 5-14. Pipeline Operation at Interrupt Request Acknowledge (General Description)**



Interrupt Response Time (internal system clock)	Condition		
	Internal interrupt	External interrupt	
Minimum	11	13	Except the condition below: • In IDLE/software STOP mode • External bus is accessed • Two or more interrupt request non-sample instructions are executed in succession • Access to interrupt control register
Maximum	18	20	

### 5.8 Periods Where Interrupt is Not Acknowledged

An interrupt request is acknowledged while an instruction is being executed. However, no interrupt request will be acknowledged between an interrupt request non-sample instruction and the next instruction.

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (vs. PSW)

In the following conditions, an interrupt may not be acknowledged.

- (1) Immediately after the RETI instruction execution
- (2) Immediately after the data write to the following registers.
  - Interrupt control register (xxICn)
  - Command register (PRCMD)
  - In-service priority register (ISPR)

**[MEMO]**

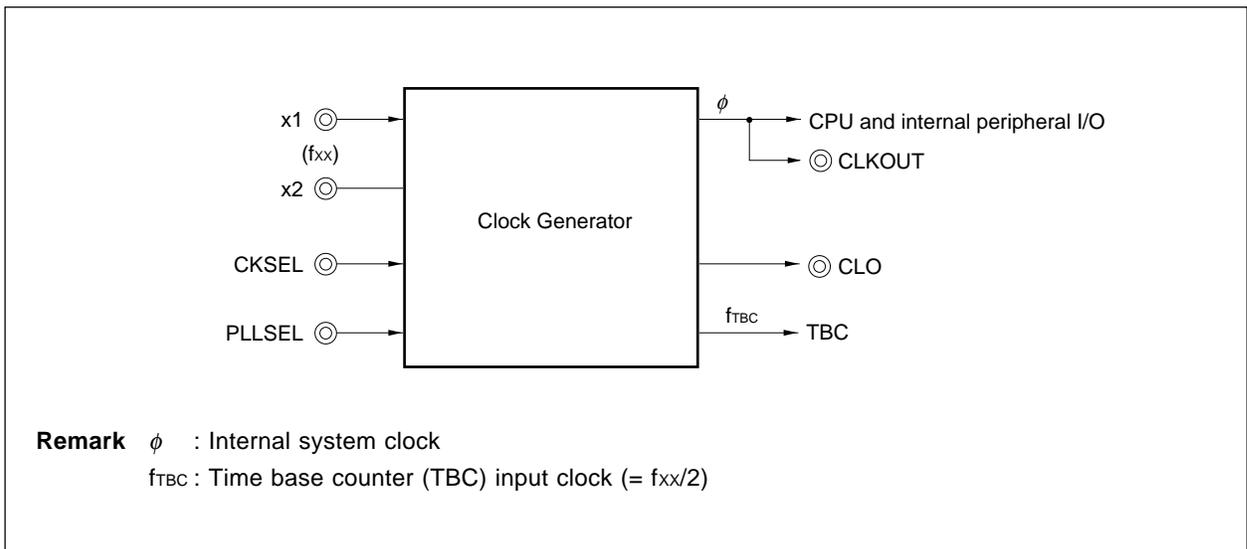
## CHAPTER 6 CLOCK GENERATOR FUNCTION

The clock generator (CG) produces and controls the internal system clock ( $\phi$ ) which is supplied to all the internal hardware units including the CPU.

### 6.1 Features

- Multiplication function by PLL (Phase Locked Loop) synthesizer
- Clock source
  - Oscillation through oscillator connection:  $f_{xx} = \phi, \phi/5$
  - External clock input:  $f_{xx} = \phi, 2 \times \phi, \phi/5$
- Power save control
  - HALT mode
  - IDLE mode
  - Software STOP mode
  - Clock output inhibit function
- Internal system clock output function

### 6.2 Configuration



### 6.3 Selecting Input Clock

The clock generator consists of an oscillator and a PLL synthesizer. It generates, for example, a 32.768 (Max. 33)-MHz internal system clock ( $\phi$ ) when a 6.5536-MHz crystal resonator or ceramic resonator is connected across the X1 and X2 pins at 5-x multiplication.

An external clock can be directly connected to the oscillator circuit. In this case, input the clock signal to the X1 pin, and leave the X2 pin open.

The clock generator is provided with two basic operation modes: PLL mode and direct mode. The operation modes are selected by the CKSEL pin. The CKSEL pin is latched at reset.

CKSEL	Operation Mode
0	PLL mode
1	Direct mode

**Caution** Use CKSEL pin with a fixed input level. Changing the level during operations of this pin may cause erroneous operation.

#### 6.3.1 Direct mode

In the direct mode, external clock with frequency twice higher than that of the internal system clock is input. Because the oscillator circuit and PLL synthesizer do not run, power consumption is significantly reduced. This mode is mainly used for application systems that operate the V854 in a relatively low frequency. Considering EMI measures, the PLL mode is recommended when the external clock frequency ( $f_{xx}$ ) is 32 MHz (internal system clock ( $\phi$ ) = 16 MHz) or more.

#### 6.3.2 PLL mode

In the PLL mode, an external clock is input by connecting an external oscillator, which is multiplied by the PLL synthesizer to generate the internal system clock ( $\phi$ ).

The PLL multiplication number that can be selected is either one or five. The PLL multiplication number can be selected by the PLLSEL pin (refer to **2.3.16 PLLSEL**).

PLLSEL	Multiplication
0	1-x multiplication
1	5-x multiplication

- Cautions**
1. Fix the PLLSEL pin so that the input level does not change (changing the input level of this pin during operation may cause erroneous operation). When this pin is set in the direct mode (CKSEL = 1), the PLLSEL pin has no function. Treat it as an unused pin.
  2. When using an external by a resonator, use this mode with  $f_{xx} = 16$  MHz max.

At reset, with reference to the input clock frequency ( $f_{xx}$ ), an internal system clock ( $\phi$ ) that is either equivalent to the base clock ( $1 \times f_{xx}$ ) or five times the base clock ( $5 \times f_{xx}$ ) is generated depending on whether 1-x or 5-x multiplication is selected.

In the PLL mode, when the clock supply from the external oscillator or external clock source is stopped, the internal system clock ( $\phi$ ) based on the freerunning frequency of the voltage-controlled oscillator circuit (VCO) in the clock generator continues operating. In this case,  $\phi$  is approximately 1 MHz (target). Do not use this mode expecting to obtain the freerunning frequency.

**Examples of the clock used in the PLL mode**

Multiplication	Internal System Clock Frequency ( $\phi$ ) [MHz]	External Oscillator/External Clock Frequency ( $f_{xx}$ ) [MHz]
1-x multiplication	33.000	33.000 <sup>Note</sup>
	25.000	25.000 <sup>Note</sup>
	20.000	20.000 <sup>Note</sup>
	16.000	16.000
5-x multiplication	33.000	6.6000
	25.000	5.0000
	20.000	4.0000
	16.000	3.2000

**Note** Cannot be oscillated by a resonator. Oscillate not higher than 16 MHz.

**6.3.3 Clock control register (CKC)**

This is an 8-bit register which controls the internal system clock frequency. It can be written only by a specific combination of instruction sequences so that its contents are not written by mistake due to erroneous program execution.

This register can be read/written in 8- or 1-bit units.

CKC	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	CKDIV1	CKDIV0	Address FFFFF072H	After reset 00H

Bit Position	Bit Name	Function																																																																				
1, 0	CKDIV1, CKDIV0	Clock Divide Sets the internal system clock frequency in the PLL mode. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th rowspan="2">Operation Mode</th> <th colspan="2">Pins</th> <th colspan="2">CKC</th> <th rowspan="2">Internal System Clock (<math>\phi</math>)</th> </tr> <tr> <th>CKSEL</th> <th>PLLSEL</th> <th>CKDIV1</th> <th>CKDIV0</th> </tr> </thead> <tbody> <tr> <td rowspan="3">Direct mode</td> <td style="text-align: center;">H</td> <td style="text-align: center;"><b>Note</b></td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>f_{xx}/2</math></td> </tr> <tr> <td style="text-align: center;">H</td> <td style="text-align: center;"><b>Note</b></td> <td style="text-align: center;">Any values</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Setting prohibited</td> </tr> <tr> <td style="text-align: center;">H</td> <td style="text-align: center;"><b>Note</b></td> <td style="text-align: center;">1</td> <td style="text-align: center;">Any values</td> <td style="text-align: center;">Setting prohibited</td> </tr> <tr> <td rowspan="4">PLL mode (1-x multiplication)</td> <td style="text-align: center;">L</td> <td style="text-align: center;">L</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>f_{xx}/2</math></td> </tr> <tr> <td style="text-align: center;">L</td> <td style="text-align: center;">L</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Setting prohibited</td> </tr> <tr> <td style="text-align: center;">L</td> <td style="text-align: center;">L</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>f_{xx}/5</math></td> </tr> <tr> <td style="text-align: center;">L</td> <td style="text-align: center;">L</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>f_{xx}/10</math></td> </tr> <tr> <td rowspan="4">PLL mode (5-x multiplication)</td> <td style="text-align: center;">L</td> <td style="text-align: center;">H</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>f_{xx}/5</math></td> </tr> <tr> <td style="text-align: center;">L</td> <td style="text-align: center;">H</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Setting prohibited</td> </tr> <tr> <td style="text-align: center;">L</td> <td style="text-align: center;">H</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>f_{xx}</math></td> </tr> <tr> <td style="text-align: center;">L</td> <td style="text-align: center;">H</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>f_{xx}/2</math></td> </tr> </tbody> </table> <p style="margin-top: 10px;"><b>Note</b> Connect <math>V_{DD}</math> or <math>V_{SS}</math> directly (refer to <b>2.4 Pin I/O Circuit Type and Connection of Unused Pins</b>).</p> <p><b>Remark</b> H : High level input L : Low level input <math>f_{xx}</math>: Input clock</p>	Operation Mode	Pins		CKC		Internal System Clock ( $\phi$ )	CKSEL	PLLSEL	CKDIV1	CKDIV0	Direct mode	H	<b>Note</b>	0	0	$f_{xx}/2$	H	<b>Note</b>	Any values	1	Setting prohibited	H	<b>Note</b>	1	Any values	Setting prohibited	PLL mode (1-x multiplication)	L	L	0	0	$f_{xx}/2$	L	L	0	1	Setting prohibited	L	L	1	0	$f_{xx}/5$	L	L	1	1	$f_{xx}/10$	PLL mode (5-x multiplication)	L	H	0	0	$f_{xx}/5$	L	H	0	1	Setting prohibited	L	H	1	0	$f_{xx}$	L	H	1	1	$f_{xx}/2$
Operation Mode	Pins			CKC		Internal System Clock ( $\phi$ )																																																																
	CKSEL	PLLSEL	CKDIV1	CKDIV0																																																																		
Direct mode	H	<b>Note</b>	0	0	$f_{xx}/2$																																																																	
	H	<b>Note</b>	Any values	1	Setting prohibited																																																																	
	H	<b>Note</b>	1	Any values	Setting prohibited																																																																	
PLL mode (1-x multiplication)	L	L	0	0	$f_{xx}/2$																																																																	
	L	L	0	1	Setting prohibited																																																																	
	L	L	1	0	$f_{xx}/5$																																																																	
	L	L	1	1	$f_{xx}/10$																																																																	
PLL mode (5-x multiplication)	L	H	0	0	$f_{xx}/5$																																																																	
	L	H	0	1	Setting prohibited																																																																	
	L	H	1	0	$f_{xx}$																																																																	
	L	H	1	1	$f_{xx}/2$																																																																	

The sequence of setting data in this register is the same as the power save control register (PSC). However, the limitation items listed in **Cautions 2** for the **3.4.9 Specific registers** do not apply. For details, refer to **6.5.2 Control register**.

**(1) Example of settings**

The example of settings is as below.

Operation Mode	Pins		CKC Register		Input Clock (f <sub>xx</sub> )	Internal System Clock (φ)
	CKSEL	PLLSEL	CKDIV1	CKDIV0		
Direct mode	H	<b>Note</b>	0	0	16 MHz	8 MHz
PLL mode (1-x multiplication)	L	L	0	0	33 MHz	33 MHz
	L	L	1	0	33 MHz	6.6 MHz
	L	L	1	1	33 MHz	3.3 MHz
PLL mode (5-x multiplication)	L	H	0	0	6.6 MHz	33 MHz
	L	H	1	0	6.6 MHz	6.6 MHz
	L	H	1	1	6.6 MHz	3.3 MHz
Other than the above					Setting prohibited	

**Note** Connect directly to V<sub>DD</sub> or V<sub>SS</sub>.

**Remark** H : High level input  
L : Low level input

### 6.4 PLL Lock-up

Following the power-on reset or when existing the software STOP mode is released, a certain length of time will be required for the PLL to stabilize. This required time is called PLL lock-up time. The status in which the frequency is not stable is called unlock status and the status in which it has been stabilized is called lock status.

Two system status registers are available to check the stabilization of the PLL frequency: UNLOCK flag that indicates the stabilization status of the PLL frequency, and PRERR flag that indicates occurrence of a protection error (for the details of the PRERR flag, refer to **3.4.9 (2) System status register (SYS)**).

The SYS register, which contains these UNLOCK and PREERR flags, can be read/written in 8- or 1-bit units.

SYS	7	6	5	4	3	2	1	0		
	0	0	0	PRERR	0	0	0	UNLOCK	Address FFFFF078H	After reset 0000000xB

Bit Position	Bit Name	Function
0	UNLOCK	Unlock Status Flag This is a read-only flag and indicates unlock status of PLL. It holds "0" as long as lock up status is maintained, and is not changed even if system is reset. 0 : Indicates lock status 1 : Indicates unlock status

**Remark** For the description of the PRERR flag, refer to **3.4.9 (2) System status register (SYS)**.

If the unlock status condition should arise, due to a power or clock source failure, the UNLOCK flag should be checked to verify that the PLL has stabilized before performing any execution speed dependent operations, such as real-time processing.

Static processing such as setting of the on-chip hardware units and initialization of the register data and memory data, however, can be executed before the UNLOCK flag is reset.

The following is the relation between the oscillation stabilization time (the time from the oscillator starts oscillating until the input wave form stabilizes) and the PLL lockup time (the time until the frequency stabilizes) when using an oscillator:

$$\text{Oscillation stabilization time} < \text{PLL lockup time}$$

## 6.5 Power Save Control

### 6.5.1 General

The V854 is provided with the following power save or standby modes to reduce power consumption when CPU operation is not required.

#### (1) HALT mode

In this mode, the clock generator (oscillator and PLL synthesizer) continues operation but the operating clock of the CPU stops. The internal peripherals continue to function in reference to the internal system clock. Total power consumption of the system can be reduced through intermittent operation between normal operation and HALT modes.

A dedicated instruction (HALT instruction) transfers to the V854 to the HALT mode.

#### (2) IDLE mode

In this mode, both the CPU clock and the internal system clock are stopped to further reduce power consumption. However, since the clock generator continues to run, normal operation can resume without having to wait for the oscillator and PLL circuits to stabilize.

Setting the PSC register, which is a specific register, transfers the V854 to the IDLE mode.

The IDLE mode is somewhere between the STOP and HALT modes in terms of clock stabilization time and power consumption, and is used in applications where the clock oscillation time should be eliminated but low power consumption is required.

★ **Caution** When inputting external clocks, continue the supply of clocks.

#### (3) Software STOP mode

In this mode, the CPU clock, the internal system clock, and the clock generator are stopped, reducing power consumption to just the leakage current. In this state, power consumption is minimized.

Setting the PSC register, which is a specific register, transfers the V854 to the software STOP mode.

#### ★ (a) PLL mode

Setting the register by software transfers the V854 to the software STOP mode. As soon as the oscillator circuit stops, the clock output of the PLL synthesizer is stopped. After the software STOP mode has been released, it is necessary to allow for stabilization time of the oscillator and system clock. Moreover, the lock up or stabilization time of the PLL may also be necessary, depending on the application.

#### ★ (b) Direct mode

The software STOP mode cannot be used in direct mode.

#### (4) Clock output inhibit

Output of the system clock from the CLKOUT pin is prohibited.

The operations of the clock generator in the normal, HALT, IDLE, and software STOP modes are shown in Table 6-1.

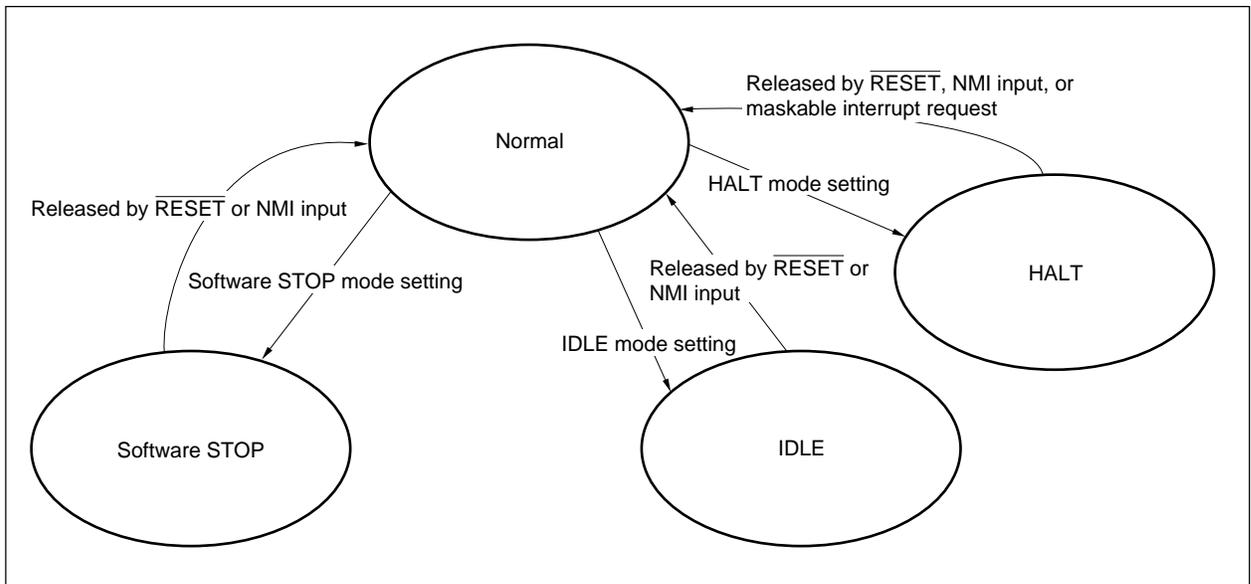
By combining and selecting the mode best suited for a specific application, the power consumption of the system can be effectively reduced.

**Table 6-1. Operation of Clock Generator by Power Save Control**

Clock Source		Standby Mode	Oscillator Circuit (OSC)	PLL Synthesizer	Clock Supply to Peripheral I/O	Clock Supply to CPU
PLL mode	Oscillation by crystal oscillator	Normal	○	○	○	○
		HALT	○	○	○	x
		IDLE	○	○	x	x
		STOP	x	x	x	x
	External clock	Normal	x	○	○	○
		HALT	x	○	○	x
		IDLE	x	○	x	x
		STOP	x	x	x	x
Direct mode	Normal	x	x	○	○	
	HALT	x	x	○	x	
	IDLE	x	x	x	x	
	STOP	x	x	x	x	

○ : Operates  
x : Stops

**Status Transition Diagram**

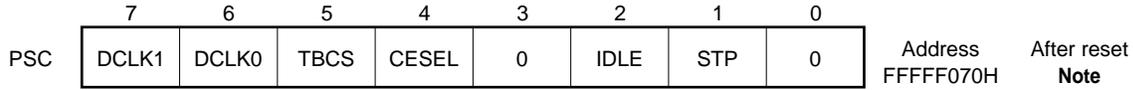


6.5.2 Control registers

(1) Power save control register (PSC)

This is an 8-bit register that controls the power save mode. This is a specific register, and only the access by the specific sequence is valid during write cycles. For details, refer to 3.4.9 Specific registers.

This register can be read/written in 8- or 1-bit units.



Bit Position	Bit Name	Function															
7, 6	DCLKn (n = 1, 0)	Disable CLKOUT Specifies operation mode of CLKOUT pin <table border="1" style="margin: 10px auto; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 15%;">DCLK1</th> <th style="width: 15%;">DCLK0</th> <th style="width: 70%;">Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Normal output mode</td> </tr> <tr> <td>0</td> <td>1</td> <td>RFU (reserved)</td> </tr> <tr> <td>1</td> <td>0</td> <td>RFU (reserved)</td> </tr> <tr> <td>1</td> <td>1</td> <td>Clock output inhibit mode</td> </tr> </tbody> </table>	DCLK1	DCLK0	Mode	0	0	Normal output mode	0	1	RFU (reserved)	1	0	RFU (reserved)	1	1	Clock output inhibit mode
DCLK1	DCLK0	Mode															
0	0	Normal output mode															
0	1	RFU (reserved)															
1	0	RFU (reserved)															
1	1	Clock output inhibit mode															
5	TBCS	Time Base Count Select Selects a number of dividing of time base counter, and specifies oscillation stabilization time. 0: $2^{15}/f_{TBC}$ (s) 1: $2^{16}/f_{TBC}$ (s) For details, refer to explanation of “Time base counter (TBC)” in section 6.6 “Specifying Oscillation Stabilization Time”.															
4	CESEL	Crystal/External Select Specifies functions of X1 and X2 pins 0: Oscillator connected to X1 and X2 pins 1: External clock connected to X1 pin When CESEL = 1, the software STOP mode cannot be used.															
2	IDLE	IDLE Mode Specifies IDLE mode. When “1” is written to this bit, IDLE mode is entered. When IDLE mode is released, this bit is automatically reset to “0”.															
1	STP	STOP Mode Specifies software STOP mode. When “1” is written to this bit, STOP mode is entered. When STOP mode is released, this bit is automatically reset to “0”.															

**Note** 00H (in ROM-less mode 1 and 2, in single-chip mode 2)  
 C0H (in single-chip mode 1 and the PROM mode)

6.5.3 HALT mode

(1) Entering and operation status

In the HALT mode, the clock generator (oscillator circuit and PLL synthesizer) operates, while the operating clock of the CPU stops. The internal peripherals continue to function in reference to the internal system clock. The total lower consumption of the system can be reduced by entering the HALT mode during the idle time of the CPU.

This mode is entered by the HALT instruction.

In the HALT mode, program execution is stopped, but the contents of the registers and internal RAM immediately before entering the HALT mode are retained. The on-chip peripheral functions that are not dependent on the instruction processing of the CPU continue to operate.

Table 6-2 shows the status of each hardware unit in the HALT mode.

Table 6-2. Operating Status in HALT Mode

Function		Operating Status	
Clock Generator		Operates	
Internal System Clock		Operates	
CPU		Stops	
I/O Port		Retained	
Peripheral Function		Operates	
Internal Data		Status of internal data before setting of HALT mode, such as CPU registers, status, data, and internal RAM contents, are retained.	
External Expansion Mode	AD0 to AD15	High impedance <sup>Note</sup>	
	A16 to A23	Retained <sup>Note</sup>	High-impedance when $\overline{\text{HLDAK}} = 0$
	$\overline{\text{LBEN}}, \overline{\text{UBEN}}$		
	R/ $\overline{\text{W}}$	High level output <sup>Note</sup>	
	$\overline{\text{DSTB}}, \overline{\text{WRL}}, \overline{\text{WRH}}, \overline{\text{RD}}$		
	ASTB		
	HLD $\overline{\text{AK}}$	Operates	
CLKOUT		Clock output (when clock output is not disabled)	
CLO		Retained (CLE bit of the CLOM register = 0) Clock output (CLE bit of the CLOM register = 1)	

**Note** The instruction fetch operation continues even after the HALT instruction has been executed, until the internal instruction prefetch queue becomes full. After the queue has become full, the operation is stopped in the status indicated in the above Table 6-2.

**(2) Releasing HALT mode**

The HALT mode can be released by the non-maskable interrupt request, an unmasked maskable interrupt request, or a  $\overline{\text{RESET}}$  signal input.

**(a) Releasing by interrupt request**

The HALT mode is unconditionally released by the NMI request or an unmasked maskable interrupt request, regardless of the priority. However, if the HALT mode is set in an interrupt processing routine, the operation will differ as follows:

- (i) If an interrupt request with a priority lower than that of the interrupt request under execution is generated, the HALT mode is released, but the newly generated interrupt request is not accepted. The new interrupt request will be kept pending.
- (ii) If an interrupt request with a priority higher (including NMI request) than the interrupt request under execution is generated, the HALT mode is released, and the interrupt request is also accepted.

**Operation after HALT mode has been released by interrupt request**

Releasing Source	Interrupt Enable (EI) Status	Interrupt disable (DI) Status
NMI request	Branches to handler address	
Maskable interrupt request	Branches to handler address or executes next instruction	Executes next instruction

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

The same operation as the normal reset operation is performed.

6.5.4 IDLE mode

(1) Entering and operation status

In this mode, both the CPU clock and the internal system clock are stopped to further reduce power consumption. However, since the clock generator continues to run, normal operation can resume without having to wait for the oscillator and PLL circuit to stabilize.

The IDLE mode is entered when the PSC register is programmed by the store (ST/SST) instruction or bit manipulation (SET1/CLR1/NOT1) instruction.

Execution of the program is stopped in the IDLE mode, but the contents of the registers and internal RAM immediately before entering the IDLE mode are retained. The on-chip peripheral functions are stopped in this mode. External bus hold request ( $\overline{\text{HLDRQ}}$ ) is not accepted.

Table 6-3 shows the hardware status in the IDLE mode.

Table 6-3. Operating Status in IDLE Mode

Function		Operating Status
Clock Generator		Operates
Internal System Clock		Stops
CPU		Stops
I/O Port		Retained
Peripheral Function		Stops
Internal Data		Status of all internal data immediately before IDLE mode is entered, such as CPU registers, status, data, and internal RAM contents, are retained.
External Expansion Mode	AD0 to AD15	High-impedance
	A16 to A23	
	$\overline{\text{LBEN}}$ , $\overline{\text{UBEN}}$	
	$\overline{\text{R/W}}$	
	$\overline{\text{DSTB}}$ , $\overline{\text{WRL}}$ , $\overline{\text{WRH}}$ , $\overline{\text{RD}}$	
	ASTB	
	HLDK	
CLKOUT		Low level output
CLO		Retained <sup>Note</sup>

**Note** Set the CLE bit of the CLOM register to 0 before switching over to the IDLE mode.

**(2) Releasing IDLE mode**

The IDLE mode is released by the NMI signal input or  $\overline{\text{RESET}}$  signal input.

**(a) Releasing by NMI signal input**

The NMI request is accepted and serviced as soon as the IDLE mode has been released.

If the IDLE mode is entered in the NMI processing routine, however, only the IDLE mode is released, and the interrupt will not be accepted. The interrupt request will be retained and kept pending.

The interrupt processing that is started by the NMI signal input when the IDLE mode is released is treated in the same manner as a normal NMI interrupt that is processed (because there is only one vector address of the NMI interrupt). Therefore, if it is necessary to distinguish between the two types of NMI interrupts, a software flag should be defined in advance, and the flag must be set before setting the IDLE flag by the store/bit manipulation instruction. By checking this flag during the NMI interrupt processing, the NMI used to released the IDLE mode can be distinguished from the normal NMI.

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

The same operation as the normal reset operation is performed.

6.5.5 Software STOP mode

(1) Entering and operation status

In this mode, the CPU clock, the internal system clock, and the clock generator are stopped, reducing power consumption to only leakage current. In this state, power consumption is minimized.

The software STOP mode is entered by programming the PSC register (Specific register) using the store (ST/SST) or bit manipulation (SET1/CLR1/NOT1) instruction (refer to 3.4.9 Specific register).

It is necessary to ensure the oscillation stabilization time of the oscillator circuit after the software STOP mode has been released, when the oscillator connection mode (CESEL bit = "0") are set.

In the software STOP mode, program execution is stopped, but all the contents of the registers and internal RAM immediately before entering the STOP mode are retained. The on-chip peripheral function also stops operation.

Table 6-4 shows the hardware status in the software STOP mode.

Table 6-4. Operating Status in Software STOP Mode

Function		Operating Status
Clock Generator		Stops
Internal System Clock		Stops
CPU		Stops
I/O Port <sup>Note 1</sup>		Retained
Peripheral Function		Stops
Internal Data <sup>Note 1</sup>		Status of all internal data immediately before software STOP mode is set, such as CPU registers, status, data, and internal RAM contents, are retained.
External Expansion Mode	AD0 to AD15	High-impedance
	A16 to A23	
	LBEN, UBEN	
	R/W	
	DSTB, WRL, WRH, RD	
	ASTB	
	HLDK	
CLKOUT		Low level output
CLO		Retained <sup>Note 2</sup>

- Notes**
1. When the value of  $V_{DD}$  is within the operating range. Even if  $V_{DD}$  drops below the minimum operating voltage, the contents of the internal RAM can be retained if the data retention voltage  $V_{DDDR}$  is maintained.
  2. Set the CLE bit of the CLOM register to 0 before switching over to the software STOP mode.

**(2) Releasing software STOP mode**

The STOP mode is released by the NMI signal input or  $\overline{\text{RESET}}$  signal input.

It is necessary to ensure the oscillation stabilization time when releasing from the STOP mode in the PLL mode (CKSEL bit = "0") and oscillator connection mode (CESEL bit = "0").

Moreover, the lock up time of the PLL may also be necessary, depending on the application. For details, refer to section **6.4 PLL Lock-up**.

**(a) Releasing by NMI signal input**

When the STOP mode is released by the NMI signal, the NMI request is also accepted.

If the STOP mode is set in an NMI processing routine, however, only the STOP mode is released, and the interrupt is not accepted. The interrupt request is retained and kept pending.

**NMI interrupt processing on releasing STOP mode**

The interrupt processing that is started by the NMI signal input when the STOP mode is released is treated in the same manner as a normal NMI interrupt that is processed (because there is only one handler address of the NMI interrupt). Therefore, if it is necessary to distinguish between the two types of NMI interrupts, a software flag should be defined in advance, and the flag must be set before setting the STOP flag by the store/bit manipulation instruction. By checking this flag during the NMI interrupt processing, the NMI used to released the STOP mode can be distinguished from the normal NMI.

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

The operation same as the normal reset operation is performed.

## 6.6 Specifying Oscillation Stabilization Time

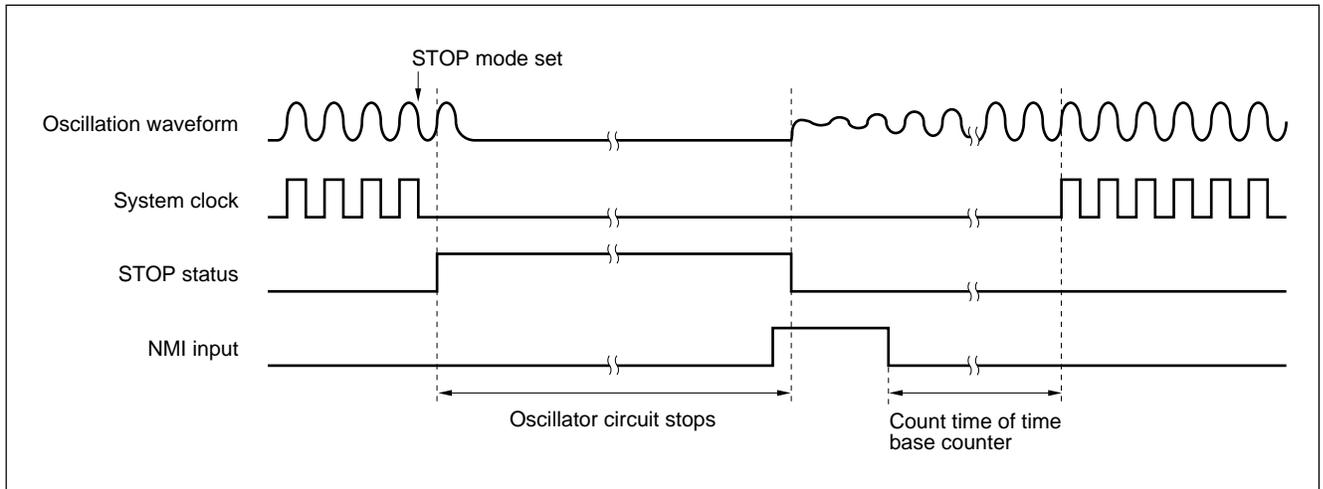
The time required for the oscillator circuit to become stabilized after the STOP mode has been released can be specified in the following two ways:

### ★ (1) Securing time using internal time base counter (NMI pin input)

When the valid edge is input to the NMI pin, the STOP mode is released. When the inactive edge is input to the pin, the time base counter (TBC) starts counting, and the time required for the clock output from the oscillator circuit to become stabilized is specified by that count time.

Oscillation stabilization time  $\approx$  (Active level width after valid edge of NMI input has been detected) + (Count time of TBC)

After a specific time has elapsed, the system clock output is started, and execution branches to the handler address of the NMI interrupt.



During inactivity, the NMI pin should be kept at the inactive level (e.g. when the valid edge is specified to be the falling edge).

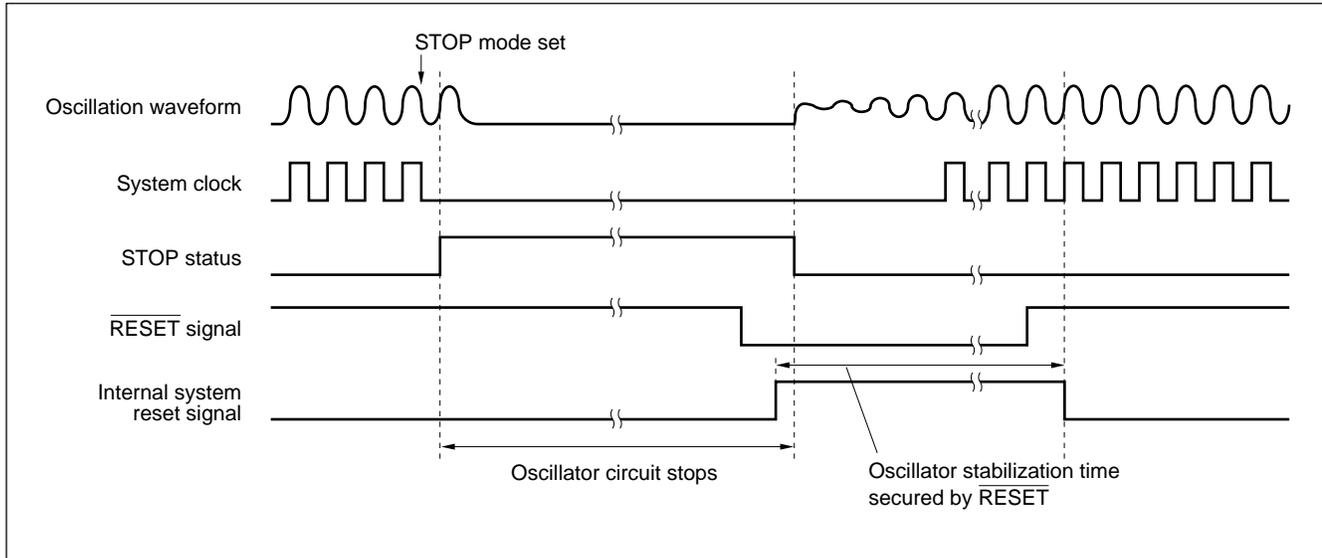
If an operation to enter the STOP mode is performed while a valid edge has been input to the NMI pin before the CPU accepts the interrupt, the STOP mode will immediately be released. If the clock generator is driven by a PLL (CKSEL = 0) and an oscillator (CESEL = 0), program execution is started after the oscillation stabilization time specified in the time base counter has elapsed.

**(2) Securing time by signal level width ( $\overline{\text{RESET}}$  pin input)**

The STOP mode is released when the falling edge is input to the  $\overline{\text{RESET}}$  pin.

The time required for the clock output from the oscillator circuit to become stabilized is specified by the low-level width of the signal input to the  $\overline{\text{RESET}}$  pin.

After the rising edge has been input to the  $\overline{\text{RESET}}$  pin, operation of the internal system clock begins, and execution branches to the vector address that is used when the system is reset.



★ **Time base counter (TBC)**

The time base counter (TBC) is used to secure the oscillation stabilization time of the oscillator circuit when the software STOP mode is released.

- In oscillator connecting mode (CESEL = 0)

TBC counts the oscillation stabilization time after the STOP mode is released, and the execution of a program is started after the counting is completed.

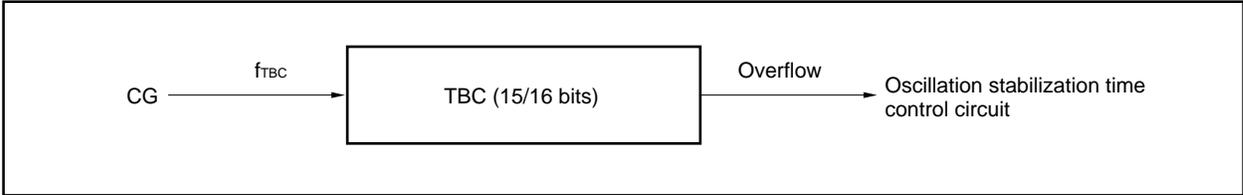
- ★ In both the PLL mode (CKSEL = 0) and oscillator connecting mode (CESEL = 0), the count clock of TBC is selected by the TBCS bit of the PSC register and the following count time can be set.

**Table 6-5. Example of Count Time**

TBCS	Frequency Division	Count Time			
		$f_{xx} = 5.0 \text{ MHz}$	$f_{xx} = 6.6 \text{ MHz}$	$f_{xx} = 25.0 \text{ MHz}$	$f_{xx} = 33.0 \text{ MHz}$
0	$2^{15}/f_{TBC}$	13.1 ms	9.8 ms	2.6 ms	2.0 ms
1	$2^{16}/f_{TBC}$	26.2 ms	19.6 ms	5.2 ms	4.0 ms

$f_{TBC}: f_{xx}/2$

**Figure 6-1. Block Configuration**



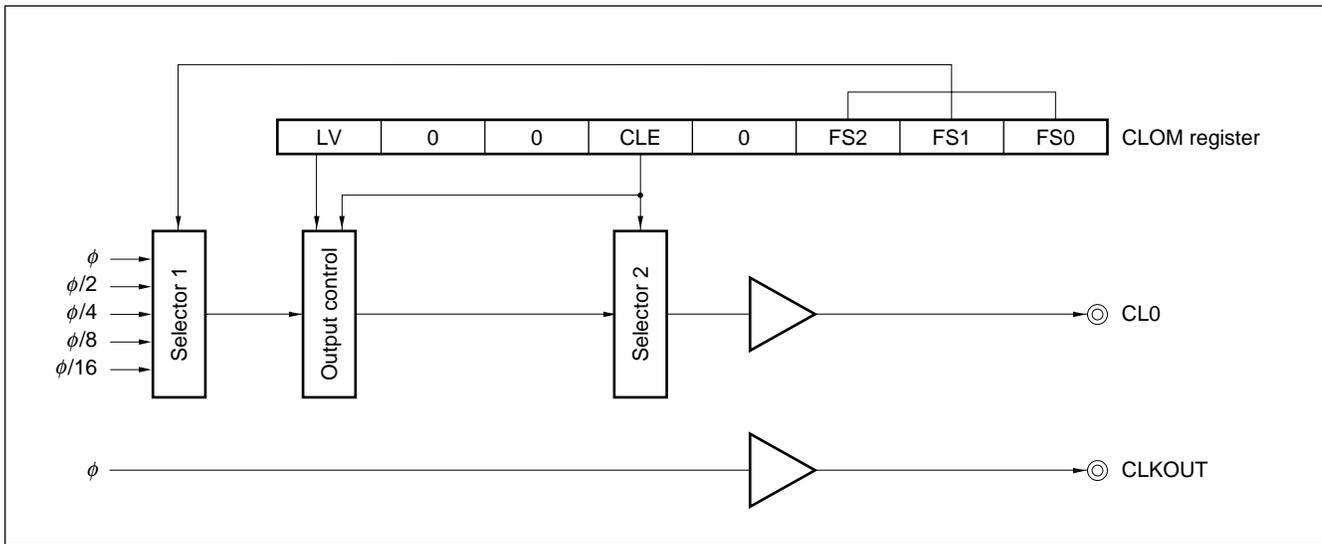
### 6.7 Clock Output Control

The V854 can output CLKOUT signal which has the same frequency as that of the system clock and CLO signal which is the frequency division of the system clock.

- CLKOUT signal =  $\phi$  (system clock)
- CLO signal =  $\phi, \phi/2, \phi/4, \phi/8, \phi/16$

The V854 can also be used as a 1-bit output port.

#### 6.7.1 Configuration



#### 6.7.2 CLKOUT signal output control

The operation mode of the CLKOUT pin can be selected by the DCLK0 and DCLK1 bits of the PSC register.

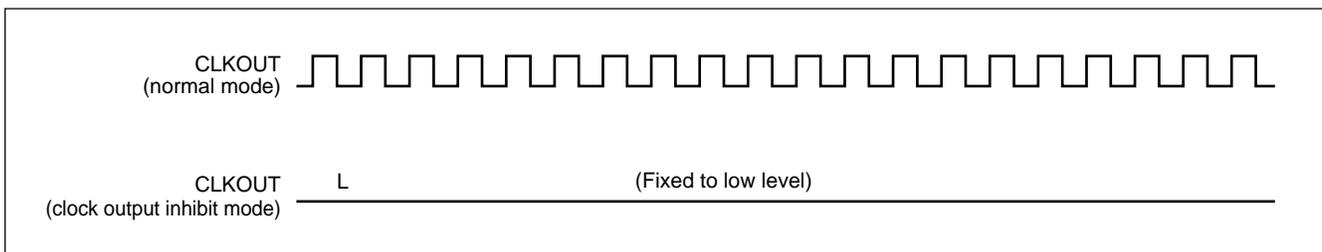
By using this operation mode in combination with the HALT, IDLE, or software STOP mode, the power dissipation can be effectively reduced (for how to write these bits, refer to **6.5.2 Control registers**).

##### Clock output inhibit mode

The clock output from the CLKOUT pin is inhibited.

This mode is ideal for single-chip mode systems or systems that fetch instructions to external expansion devices or asynchronously accesses data.

Because the operation of CLKOUT is completely stopped in this mode, the power dissipation can be minimized and radiation noise from the CLKOUT pin can be suppressed.



The PSC register reset value is 00H in ROMless mode 1 and 2 and single-chip mode 2 and C0H in single-chip mode 1 and PROM mode. Therefore, the CLKOUT signal is output during the reset period in ROMless mode 1 and 2 and single-chip mode 2. In single-chip mode 1, the CLKOUT signal is not output until the DCLK1 and DCLK0 bits of the PSC register are set to "11" after reset is released (low level output).

In the PROM mode, the CLKOUT signal is not output (low level output).

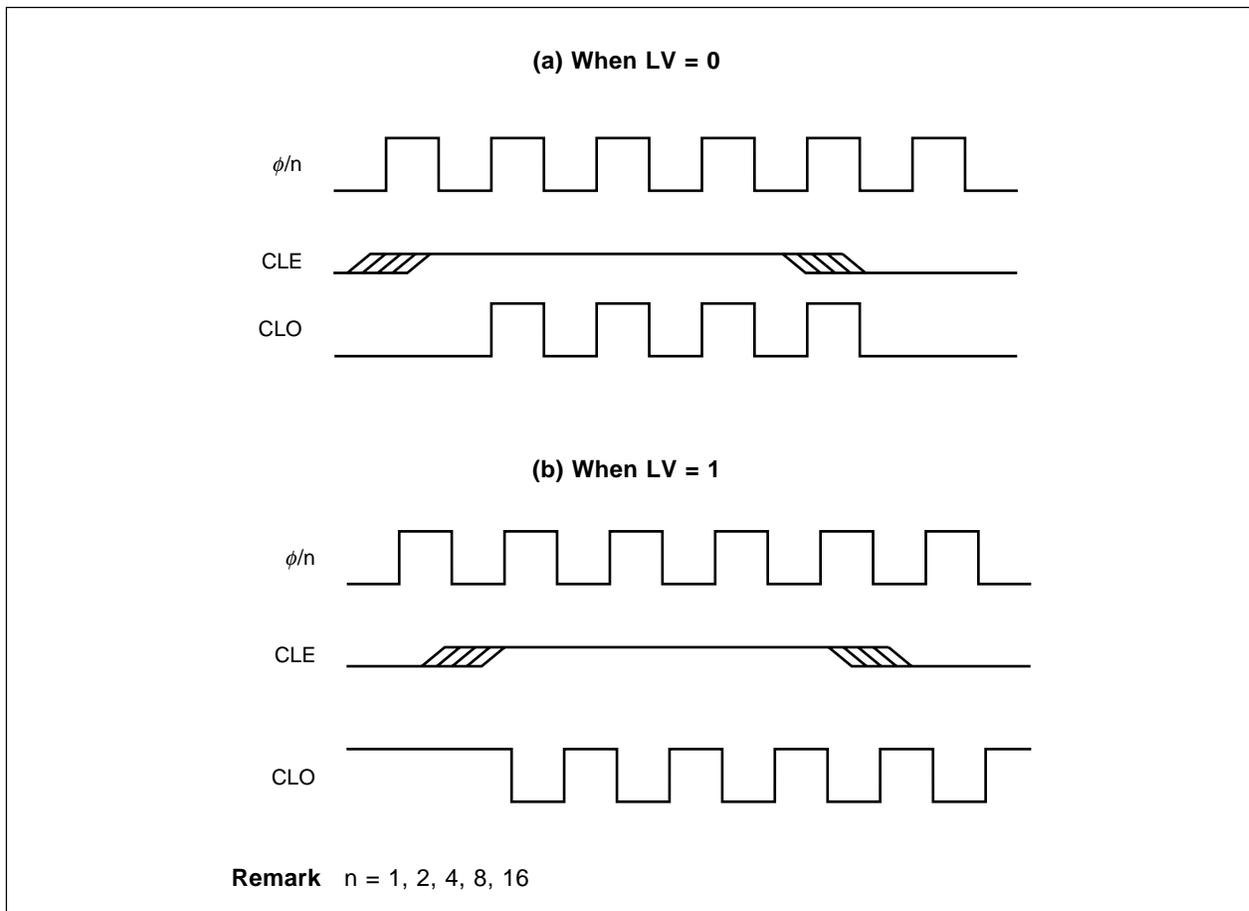
### 6.7.3 CLO signal output control

The clock to be output to CLO pin can be selected by the FS bit of the CLOM register.

CLO pin outputs a signal which has the same level as that of the LV bit when the CLE bit is 0. Signal is output synchronized with the clock (the frequency selected by the FS bit ) immediately after the CLE bit is set to 1.

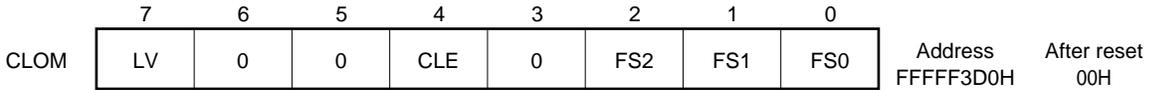
Then, if the CLE bit is set to 0, the same level as that of the LV bit is output, and the output operation thereafter is stopped.

Figure 6-2. CLO Signal Output Timing



**(1) Clock output mode register (CLOM)**

This register controls clock output function. This register can be read/written in 8- or 1-bit units.

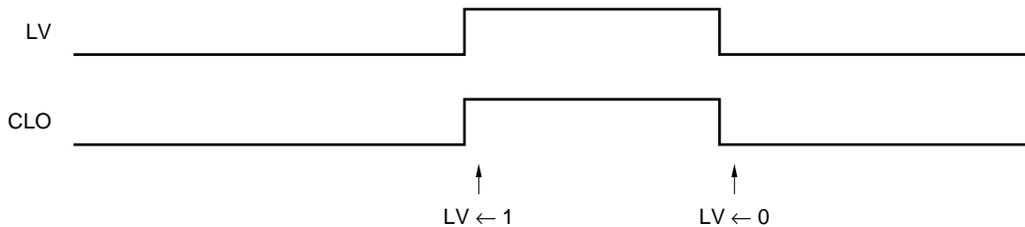


Bit Position	Bit Name	Function																												
7	LV	Level Selects output level of CLO signal 0 : Low-level output 1 : High-level output																												
4	CLE	Clock Enable Controls clock output of CLO signal 0 : Outputs the contents of LV bit 1 : Outputs the clock selected by FS2 to FS0 bits																												
2 to 0	FS2 to FS0	Frequency Select Selects the frequency of CLO signal <table border="1" style="margin: 10px auto; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 10%;">FS2</th> <th style="width: 10%;">FS1</th> <th style="width: 10%;">FS0</th> <th style="width: 70%;">Selection of Frequency</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td><math>\phi</math></td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td><math>\phi/2</math></td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td><math>\phi/4</math></td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td><math>\phi/8</math></td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td><math>\phi/16</math></td> </tr> <tr> <td colspan="3">Others</td> <td>Setting prohibited</td> </tr> </tbody> </table> <p><b>Remark</b> <math>\phi</math>: Internal system clock</p>	FS2	FS1	FS0	Selection of Frequency	0	0	0	$\phi$	0	0	1	$\phi/2$	0	1	0	$\phi/4$	0	1	1	$\phi/8$	1	0	0	$\phi/16$	Others			Setting prohibited
FS2	FS1	FS0	Selection of Frequency																											
0	0	0	$\phi$																											
0	0	1	$\phi/2$																											
0	1	0	$\phi/4$																											
0	1	1	$\phi/8$																											
1	0	0	$\phi/16$																											
Others			Setting prohibited																											

**Caution** Do not change the values of the other bits (LV, FS2 to FS0) during setting of the CLE bit (1). Do not change the values of other bits (LV, FS2 to FS0) simultaneously with changing the value of the CLE bit.

**(2) 1-bit output port**

When the CLE bit is 0, CLO pin outputs the signals of the same level as that of the LV bit. When the contents of the LV bit is changed, CLO signal changes immediately.



**(3) Operations in standby mode****(a) HALT mode**

The status before setting HALT mode is maintained. Clock continues to be output while clock is being output. When clock output is disabled, the signal of the same level that of the LV bit before HALT mode is set is output.

**(b) Software STOP mode/IDLE mode**

Disables clock output before setting these modes (the CLE bit is cleared by software). The signal of the same level as that of the LV bit before these modes are set is output from the CLO pin.

**[MEMO]**

## CHAPTER 7 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)

### 7.1 Features

- Timer 0: 24-bit timer/event counter (1 channel)
  - Capture/compare registers: 4
  - Can be used as a trigger of A/D converter (CC03 coincidence)
  - Set/reset outputs: 2
  - Clearing and starting timer
  - External input pulse measurement
  - Overflow interrupt request and overflow flag
  - Applications: measurement of pulse interval and frequency, output of pulses with various waveforms
- Timer 1: 24-bit timer/event counter (1 channel)
  - Capture registers: 4
  - Compare registers: 2
  - INTP edge detection circuit with 1 to 64/1 to 128 frequency divider
  - Can be used as a trigger of real time output port (CM10 coincidence)
  - Overflow interrupt request and overflow flag
  - Clearing and starting timer
  - Applications: measurement of pulse interval and frequency of software servo, etc.
- Timer 2: 16-bit interval timer counter (5 channels)
  - Compare registers: 1
  - Toggle output: 1
  - External input pulse measurement
  - Clearing and starting timer
  - Applications: interval timer, pulse counter, and pulse output of constant period for software
- Timer 3: 16-bit interval timer (1 channel)
  - Dedicated capture register: 1
  - Capture/compare register: 1
  - INTP edge detection circuit with digital noise elimination
  - Clearing and starting timer
  - Applications: measurement of pulse interval and frequencies such as pulse width measurement of remote controllers

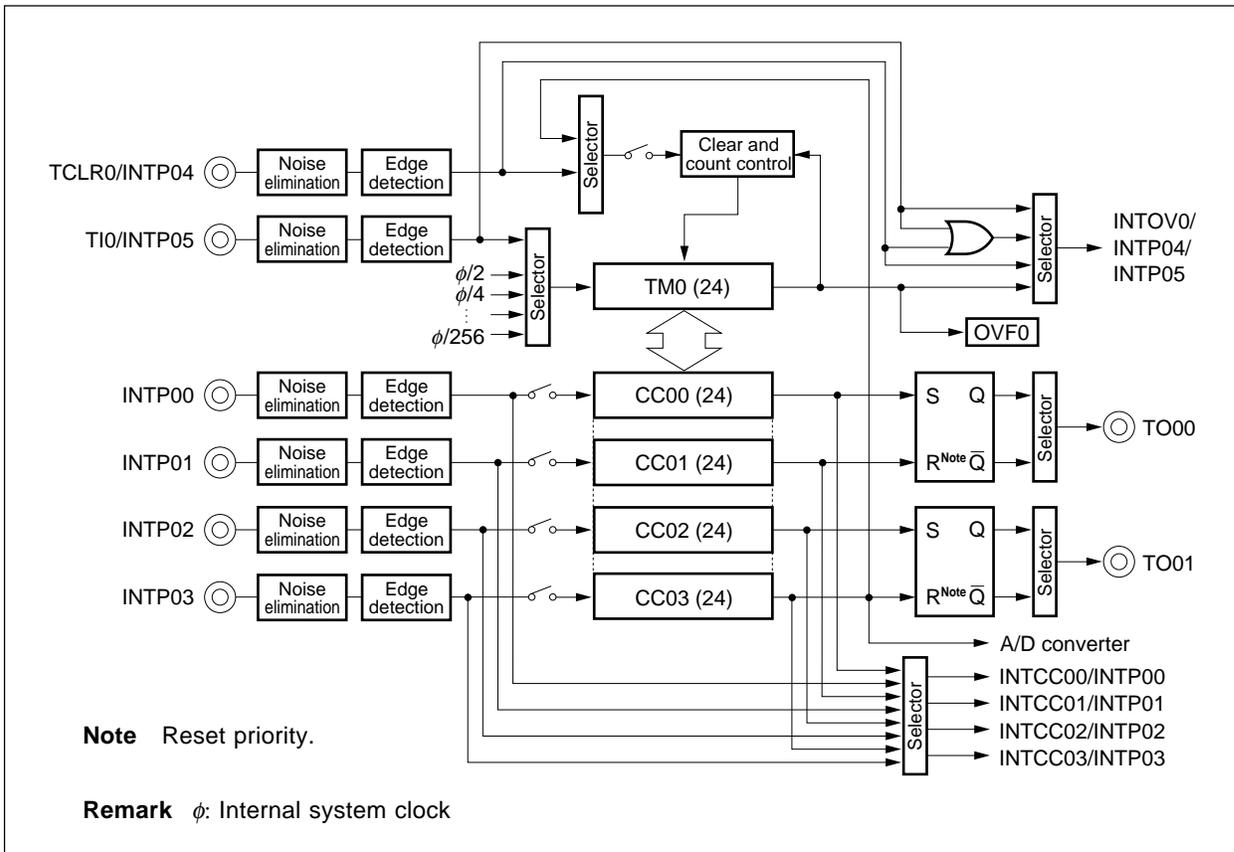
7.2 Basic Configuration

Table 7-1. List of Real-Time Pulse Unit (RPU) Configuration

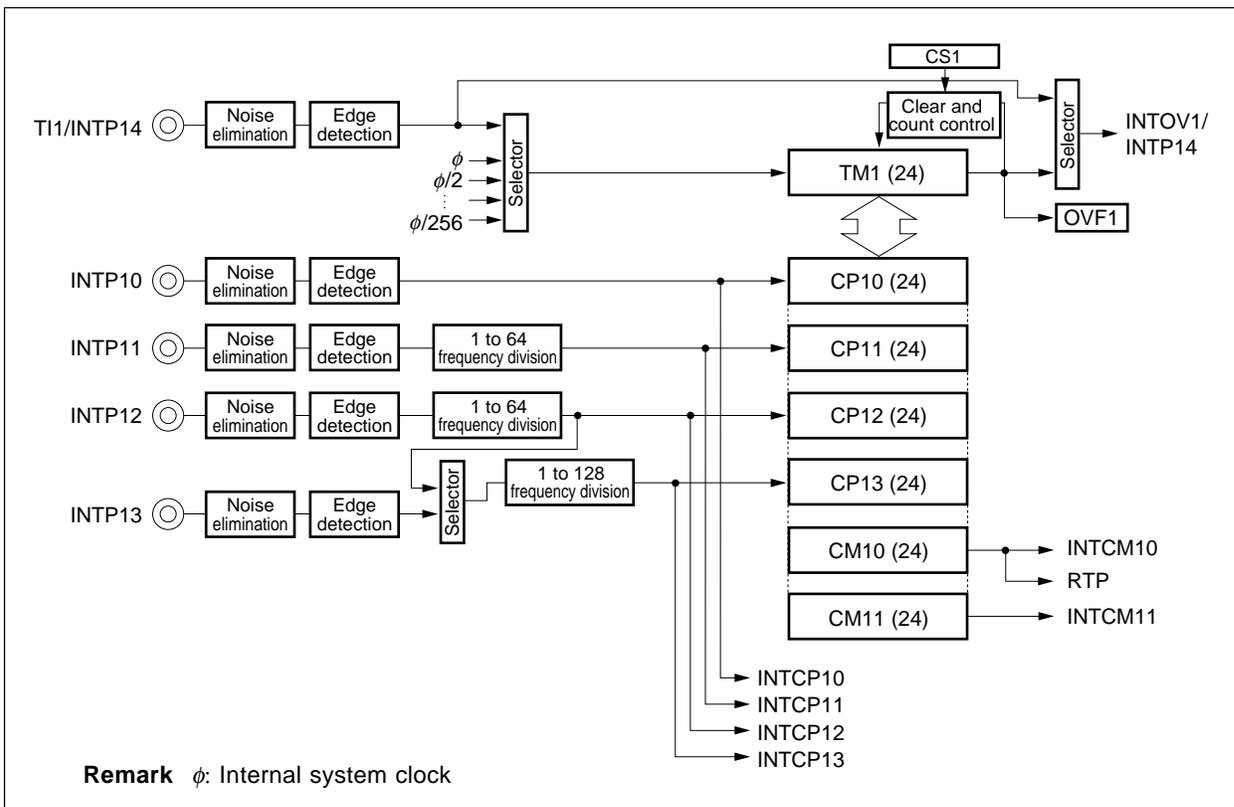
Timer	Bit Width	Count Clock	Register	Read/Write (R/W)	Clear Condition	Generated Interrupt Signal	Capture Trigger	Compare Match Trigger
Timer 0	24	$\phi/2, \phi/4, \phi/8, \phi/16, \phi/32, \phi/64, \phi/128, \phi/256,$ T10 pin input	TM0	R	<ul style="list-style-type: none"> <li>• INTOV0 input</li> <li>• TCLR0 input</li> <li>• INTCC03 input</li> </ul>	INTOV0	—	—
			CC00	R/W		INTCC00	INTP00	TO00 (S)
			CC01			INTCC01	INTP01	TO00 (R)
			CC02			INTCC02	INTP02	TO01 (S)
			CC03			INTCC03	INTP03	TO01 (R), A/D converter
Timer 1	24	$\phi, \phi/2, \phi/4, \phi/8, \phi/16, \phi/32, \phi/64, \phi/128, \phi/256,$ T11 pin input	TM1	R	<ul style="list-style-type: none"> <li>• INTOV1 input</li> <li>• Software clear</li> </ul>	INTOV1	—	—
			CP10			INTCP10	INTP10	—
			CP11			INTCP11	Frequency division of INTP11	—
			CP12			INTCP12	Frequency division of INTP12	—
			CP13			INTCP13	Frequency division of INTP12/INTP13	—
			CM10	R/W		INTCM10	—	Real time output port
			CM11			INTCM11	—	—
Timer 2	16	$\phi/2, \phi/4, \phi/8, \phi/16, \phi/32, \phi/64, \phi/128, \phi/256,$ T12n pin input (n = 0 to 4)	TM20	R	<ul style="list-style-type: none"> <li>• INTCM2n input</li> <li>• Software clear</li> </ul>	—	—	—
			CM20	R/W		INTCM20	—	TO20 (T)
			TM21	R		—	—	—
			CM21	R/W		INTCM21	—	TO21 (T)
			TM22	R		—	—	—
			CM22	R/W		INTCM22	—	TO22 (T)
			TM23	R		—	—	—
			CM23	R/W		INTCM23	—	TO23 (T)
			CM24	R/W		INTCM24	—	TO24 (T)
Timer 3	16	$\phi/2, \phi/4, \phi/8, \phi/16, \phi/32, \phi/64, \phi/128, \phi/256$	TM3	R	<ul style="list-style-type: none"> <li>• INTCC3 input</li> <li>• Capture trigger of CC3</li> <li>• Software clear</li> </ul>	—	—	—
			CC3	R/W		INTCC3	INTP30	—
			CP3	R		—	—	—

**Remark**  $\phi$  : Internal system clock  
 S/R: Set/reset  
 T : Toggle output

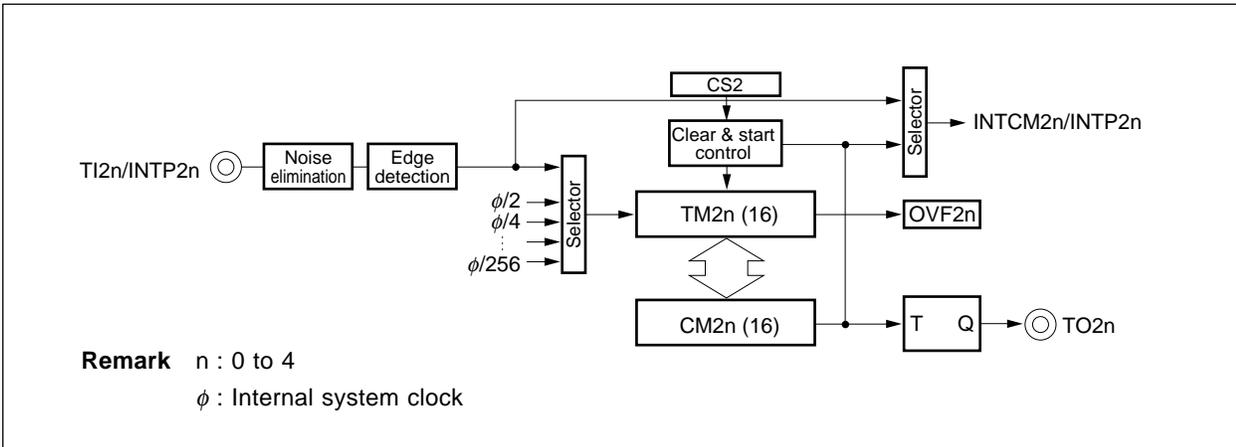
(1) Timer 0 (24-bit timer/event counter)



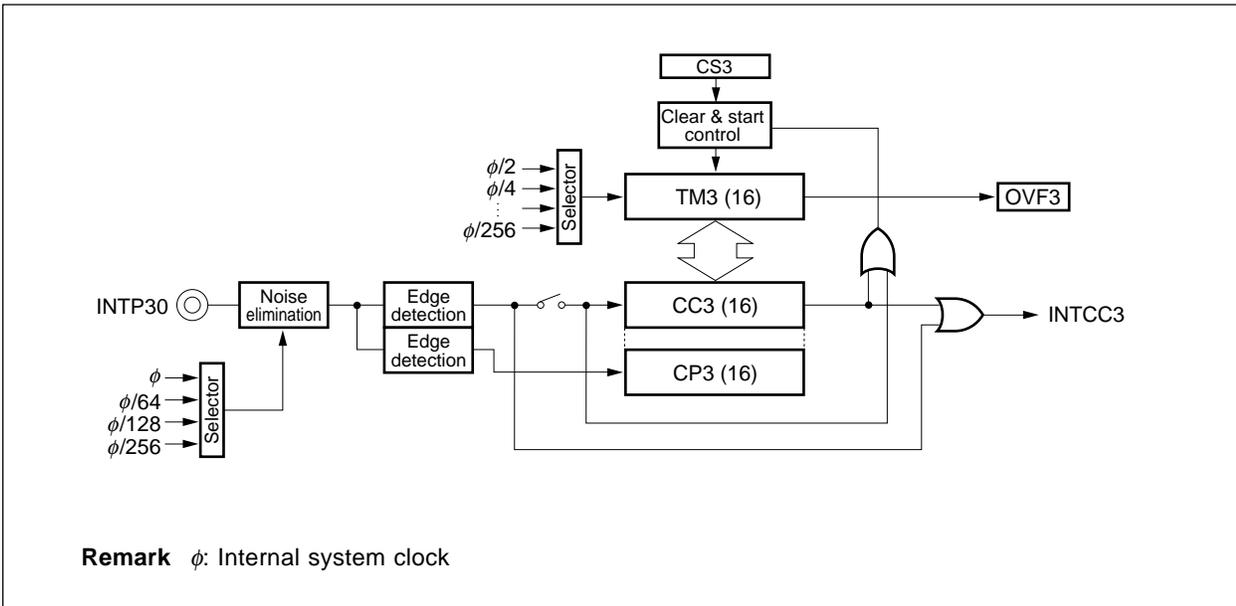
(2) Timer 1 (24-bit timer/event counter)



(3) Timer 2 (16-bit interval timer counter)



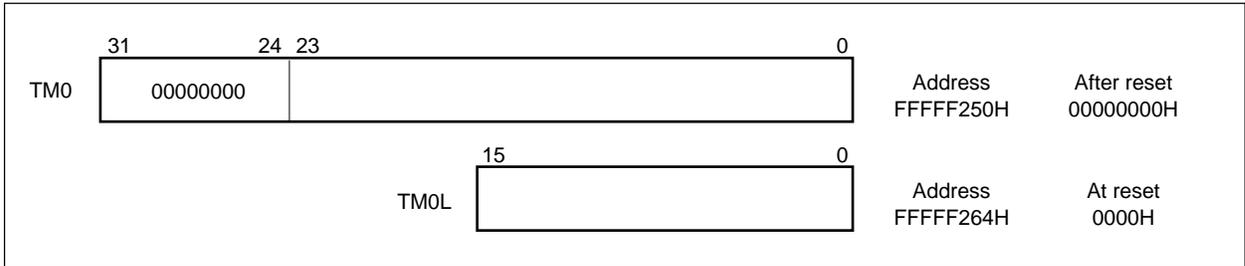
(4) Timer 3 (16-bit interval timer)



7.2.1 Timer 0

(1) Timers 0, 0L (TM0, TM0L)

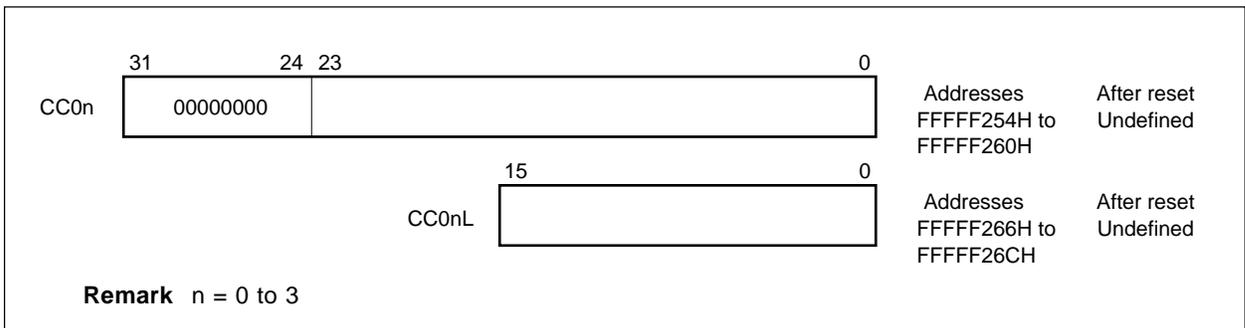
- ★ TM0 functions as a 24-bit interval timer, free-running timer or event counter for external signals. It is used to measure cycles and frequency, and also for pulse generation.
- Only 32-bit read access is enabled for TM0 (however, the high-order 8 bits are fixed to 0), and only 16-bit read access is enabled for TM0L.
- To read the low-order 24 bits of timer 0, specify TM0 with word access, and to read the low-order 16 bits only, specify TM0 with half-word access.



TM0 counts up the internal count clock or external count clock. The timer is started or stopped by the CEO bit of timer control register 00 (TMC00).

(2) Capture/compare registers 00 to 03 (CC00 to CC03, CC00L to CC03L)

- The capture/compare registers are 24-bit registers connected to TM0. They can be used as capture registers or compare registers depending on the specification of timer control register 01 (TMC01).
- 32-bit read/write access is enabled for CC0n (however, the high-order 8 bits are fixed to 0, and are ignored during write operation), and 16-bit read/write access is enabled for CC0nL.
- To access the low-order 24 bits or the low-order 16 bits of these registers, specify CC0n and CC0nL, respectively.



(a) When used as a capture register

When a capture/compare register is used as a capture register, it detects the valid edge of the corresponding external interrupt INTPn (n = 10 to 13) as a capture trigger. Timer 0 latches the count value in synchronization with the capture trigger (capture operation). The latched value is held by the capture register until the next capture operation is performed.

(b) When used as a compare register

When a capture/compare register is used as a compare register, it compares its contents with the value of the timer at each clock tick.

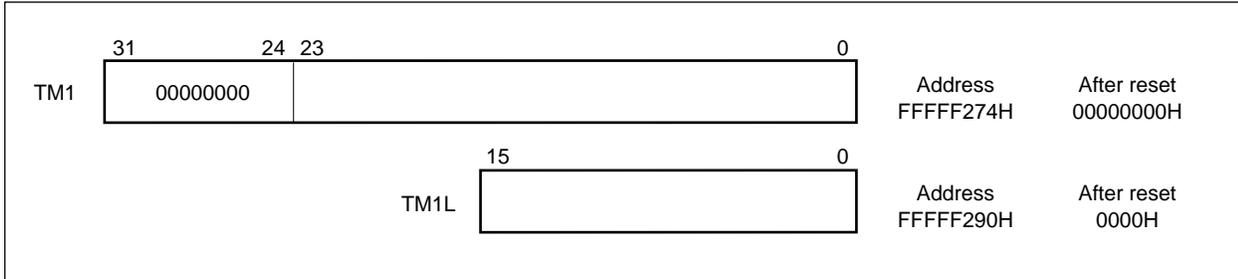
Compare registers support the set/reset output function. In other words, they set or reset the corresponding timer output synchronously with the coincidence signal generation.

7.2.2 Timer 1

(1) Timers 1, 1L (TM1, TM1L)

TM1 functions as a 24-bit free-running timer or event counter. Timers 1 to 14 are used to measure cycles and frequency, and also for programmable pulse generation.

TM1 is specified in 32-bit access, and TM1L is specified in lower 16-bit access. Only 32-bit read access is enabled for TM1, and only 16-bit read access is enabled for TM1L.

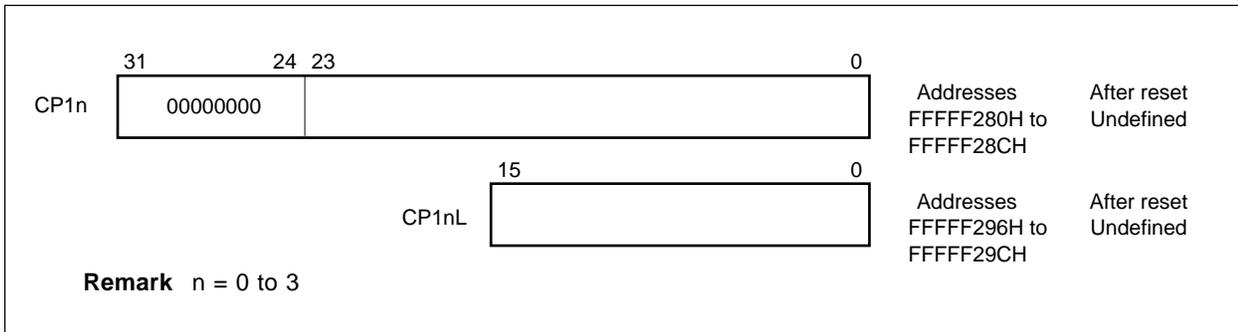


TM1 counts up the internal count clock or external count clock. The timer is started or stopped by the CE1 bit of timer control register 1 (TMC1).

(2) Capture registers 10 to 13 (CP10 to CP13, CP10L to CP13L)

The capture registers are 24-bit registers connected to TM1. These registers can be only read in 32-bit units. CP1n is specified in 32-bit access to this register. CP1nL is specified in lower 16-bit access.

Only 32-bit read access is enabled for CP1n, and only 16-bit read access is enabled for CP1nL.



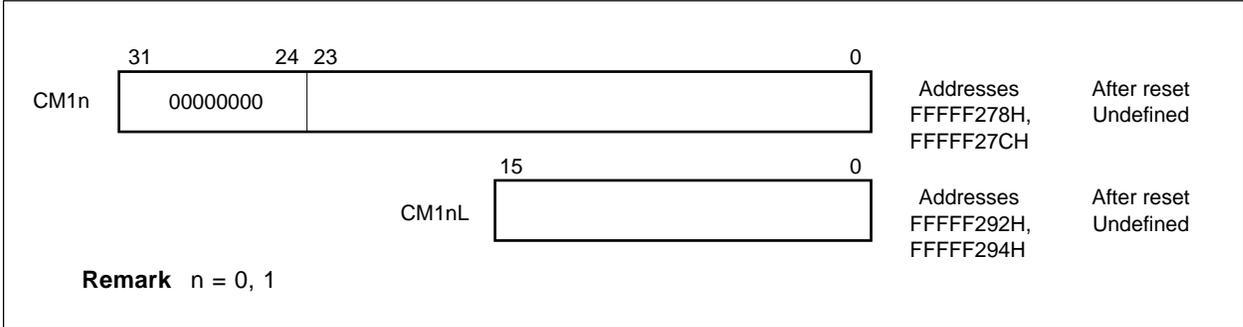
**(3) Compare registers 10, 11 (CM10, CM11, CM10L, CM11L)**

The compare registers are 24-bit registers connected to TM1. These registers can be read or written in 32-bit units.

CM1n is specified in 32-bit access to this register. CM1nL is specified in lower 16-bit access.

CM1n can be read or written in 32-bit units. The values written in bits 24 to 31 are ignored.

CM1nL can be read or written in 16-bit units. 00H is written in the higher bits (bits 16 to 23) in write access to CM1nL.

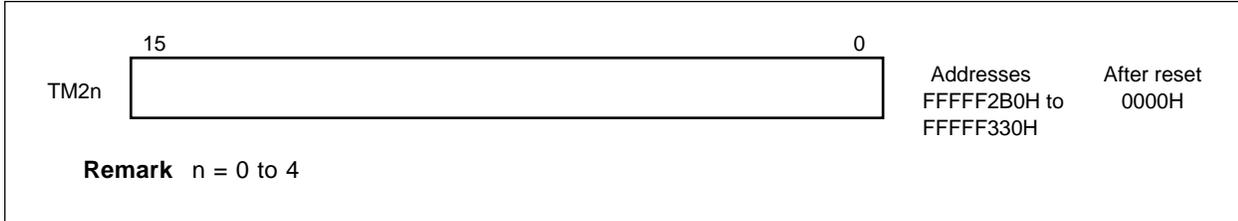


7.2.3 Timer 2

(1) Timers 20 to 24 (TM20 to TM24)

TM2n is a 16-bit timer and is mainly used as an interval timer for software.

TM2n can be only read in 16-bit units.



TM2n is started or stopped by the CE2n bit of timer control register 2n (TMC2n).

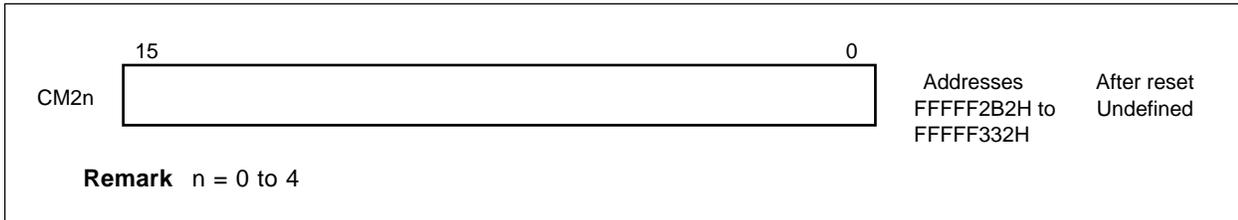
The count clock is selected by the TMC2n register from  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ ,  $\phi/256$ , and TI2n input.

**Caution** When the value of the timer coincides with the value of the compare register (CM2n), the timer is cleared by the next clock tick. If the division ratio is large and results in a slow clock period, the timer value may not be cleared to zero yet, if the timer is read immediately after the occurrence of the coincidence signal interrupt.

The count clock cannot be changed during timer operations.

(2) Compare registers 20 to 24 (CM20 to CM24)

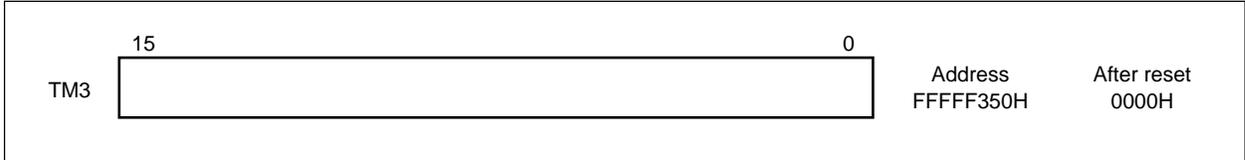
The compare registers are 16-bit registers connected to TM2n. These registers can be read or written in 16-bit units.



7.2.4 Timer 3

(1) Timer 3 (TM3)

TM3 is a 16-bit timer and is mainly used as an interval timer for software.  
This timer can be only read in 16-bit units.

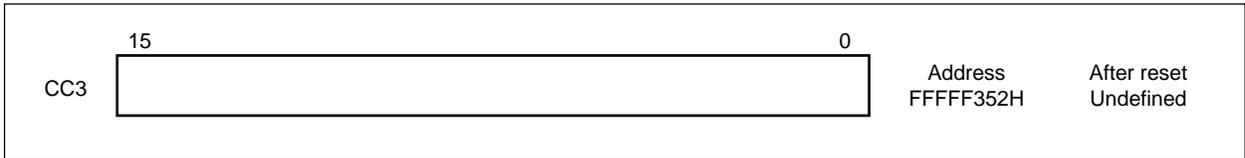


TM3 is started or stopped by the CE3 bit of timer control register 3 (TMC3).  
The count clock is selected by the TMC3 register from  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ , or  $\phi/256$ .

**Caution** When the value of the timer matches the value of the compare register (CM4), the timer is cleared by the next clock tick. If the division ratio is large and results in a slow clock period, the timer value may not be cleared to zero yet, if the timer is read immediately after the occurrence of the coincidence signal interrupt.  
The count clock cannot be changed during timer operations.

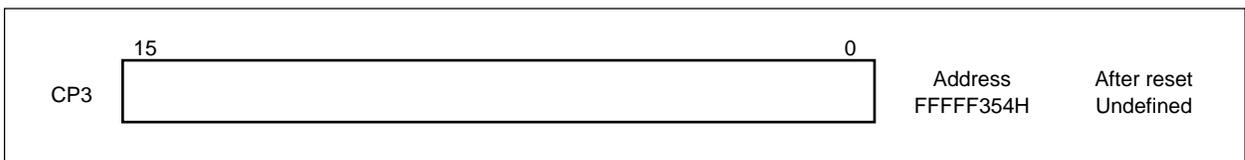
(2) Capture compare register 3 (CC3)

CC3 is a 16-bit register connected to TM3. This register can be read/written in 16-bit units.



(3) Capture register 3 (CP3)

CP3 is a 16-bit register connected to TM3. This register can be only read in 16-bit units.



7.3 Control Register

(1) Timer control register 00 (TMC00)

TMC00 specifies control of count enable/disable and the count clock of TM0.

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
TMC00	CE0	OST0	0	0	PRM03	PRM02	PRM01	PRM00	Address FFFFFF240H	After reset 01H

Bit Position	Bit Name	Function																																																							
7	CE0	<p>Count Enable Specifies enable/disable of timer/count.</p> <p>0 : Timer count disabled (stops at TM0 = 000000H) 1 : Timer count enabled</p> <p>When TMC02. ECLR0 = 1, the timer does not start counting up until TCLR0 input. When TMC02. ECLR0 = 0, writing "1" to the CE0 bit triggers count start of the timer. Therefore, even ECLR0 = 0 is set after setting CE0 with ECLR0 = 1, the timer does not start.</p>																																																							
6	OST0	<p>Overflow Stop Specifies the operation after overflow of timer.</p> <p>0 : The timer continues counting after overflow has occurred 1 : The timer retains 000000H and stops after overflow has occurred</p> <p>The timer resumes counting when the following operation is performed. When ECLR0 = 0: writing 1 to CE0 bit When ECLR0 = 1: trigger inputting to timer clear pin (TCLR0)</p>																																																							
3 to 0	PRM03 to PRM00	<p>Prescaler Clock Mode Selects the count clock frequency.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>PRM03</th> <th>PRM02</th> <th>PRM01</th> <th>PRM00</th> <th>Count Clock</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td><math>\phi/2</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td><math>\phi/4</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td><math>\phi/8</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td><math>\phi/16</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td><math>\phi/32</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td><math>\phi/64</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td><math>\phi/128</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td><math>\phi/256</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>TIO input</td> </tr> <tr> <td colspan="4">Others</td> <td>Setting prohibited</td> </tr> </tbody> </table> <p><b>Caution</b> Do not change the count clock frequency while the timer operates. <b>Remark</b> <math>\phi</math>: Internal system clock</p>	PRM03	PRM02	PRM01	PRM00	Count Clock	0	0	0	1	$\phi/2$	0	0	1	0	$\phi/4$	0	0	1	1	$\phi/8$	0	1	0	0	$\phi/16$	0	1	0	1	$\phi/32$	0	1	1	0	$\phi/64$	0	1	1	1	$\phi/128$	1	0	0	0	$\phi/256$	1	1	1	1	TIO input	Others				Setting prohibited
PRM03	PRM02	PRM01	PRM00	Count Clock																																																					
0	0	0	1	$\phi/2$																																																					
0	0	1	0	$\phi/4$																																																					
0	0	1	1	$\phi/8$																																																					
0	1	0	0	$\phi/16$																																																					
0	1	0	1	$\phi/32$																																																					
0	1	1	0	$\phi/64$																																																					
0	1	1	1	$\phi/128$																																																					
1	0	0	0	$\phi/256$																																																					
1	1	1	1	TIO input																																																					
Others				Setting prohibited																																																					

**(2) Timer control register 01 (TMC01)**

TMC01 selects the function of the capture/compare register and sets enable/disable of the timer clear function. The contents of the register and the timer count operation are not affected even if the contents of TMC01 is rewritten during timer 0 operation.

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
TMC01	CMS03	CMS02	CMS01	CMS00	IMS03	IMS02	IMS01	IMS00	Address FFFFFF242H	After reset 00H

Bit Position	Bit Name	Function															
7 to 4	CMS03 to CMS00	<p>Capture/Compare Mode Select Selects the operation mode of the capture/compare register (CC0n). Set in combination with the IMS bit. For the contents of the setting, refer to the explanation of the IMS bit.</p>															
3 to 0	IMS03 to IMS00	<p>Interrupt Mode Select Selects the interrupt source. Set in combination with the CMS bit.</p> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th style="width: 10%;">CMS0n</th> <th style="width: 10%;">IMS0n</th> <th style="width: 80%;">CC0n Register Operation Mode/Interrupt Source Selection</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Operates as a capture register. Interrupt is generated at capture timing.</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Setting prohibited</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Operates as a compare register. Interrupt is generated by coincidence signal of compare register. Capture trigger from INTP0n is ignored.</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Operates as a compare register. Interrupt is generated at INTP0n signal input timing.</td> </tr> </tbody> </table>	CMS0n	IMS0n	CC0n Register Operation Mode/Interrupt Source Selection	0	0	Operates as a capture register. Interrupt is generated at capture timing.	0	1	Setting prohibited	1	0	Operates as a compare register. Interrupt is generated by coincidence signal of compare register. Capture trigger from INTP0n is ignored.	1	1	Operates as a compare register. Interrupt is generated at INTP0n signal input timing.
CMS0n	IMS0n	CC0n Register Operation Mode/Interrupt Source Selection															
0	0	Operates as a capture register. Interrupt is generated at capture timing.															
0	1	Setting prohibited															
1	0	Operates as a compare register. Interrupt is generated by coincidence signal of compare register. Capture trigger from INTP0n is ignored.															
1	1	Operates as a compare register. Interrupt is generated at INTP0n signal input timing.															

**Remark** n = 0 to 3

**(3) Timer control register 02 (TMC02)**

TMC02 selects the function of the capture/compare register and sets enable/disable of the timer clear function.

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
TMC02	0	0	IMS05	IMS04	0	0	ECLR0	CCLR0	Address	After reset
									FFFFFF244H	00H

Bit Position	Bit Name	Function															
5, 4	IMS05, IMS04	Interrupt Mode Select Selects the interrupt source. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>IMS05</th> <th>IMS04</th> <th>Selection of Interrupt Source</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Overflow interrupt is generated by TM0</td> </tr> <tr> <td>0</td> <td>1</td> <td>Interrupt is generated by INTP04</td> </tr> <tr> <td>1</td> <td>0</td> <td>Interrupt is generated by INTP05</td> </tr> <tr> <td>1</td> <td>1</td> <td>Interrupt is generated by OR of INTP04 and INTP05</td> </tr> </tbody> </table>	IMS05	IMS04	Selection of Interrupt Source	0	0	Overflow interrupt is generated by TM0	0	1	Interrupt is generated by INTP04	1	0	Interrupt is generated by INTP05	1	1	Interrupt is generated by OR of INTP04 and INTP05
IMS05	IMS04	Selection of Interrupt Source															
0	0	Overflow interrupt is generated by TM0															
0	1	Interrupt is generated by INTP04															
1	0	Interrupt is generated by INTP05															
1	1	Interrupt is generated by OR of INTP04 and INTP05															
1	ECLR0	External Input Timer Clear Controls clear and start of TM0 by external clear input (TCLR0). <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>ECLR0</th> <th>Clear and Start of TM0</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>TM0 is not cleared</td> </tr> <tr> <td>1</td> <td>TM0 is cleared and count up starts</td> </tr> </tbody> </table>	ECLR0	Clear and Start of TM0	0	TM0 is not cleared	1	TM0 is cleared and count up starts									
ECLR0	Clear and Start of TM0																
0	TM0 is not cleared																
1	TM0 is cleared and count up starts																
0	CCLR0	Compare Input Timer Clear Controls clear and start of TM0 by CC03 match. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>CCLR0</th> <th>Clear and Start of TM0</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>TM0 is not cleared</td> </tr> <tr> <td>1</td> <td>TM0 is cleared and count up starts</td> </tr> </tbody> </table>	CCLR0	Clear and Start of TM0	0	TM0 is not cleared	1	TM0 is cleared and count up starts									
CCLR0	Clear and Start of TM0																
0	TM0 is not cleared																
1	TM0 is cleared and count up starts																

**(4) Timer control register 1 (TMC1)**

TMC1 specifies count enable/disable and controls the count clock of TM1.

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
TMC1	CE1	OST1	CS1	IMS1	PRM13	PRM12	PRM11	PRM10	Address FFFFF270H	After reset 01H

Bit Position	Bit Name	Function																																																							
7	CE1	Count Enable Specifies enable/disable of timer count. 0 : Timer count disabled (stops at TM1 = 000000H) 1 : Timer count enabled																																																							
6	OST1	Overflow Stop Specifies the operation after overflow of timer. 0 : The timer continues counting after overflow has occurred 1 : The timer retains 000000H and stops after overflow has occurred The timer resumes counting when the following operation is performed. Writing 1 to CE1 bit Writing 1 to CS1 bit																																																							
5	CS1	Clear & Start Controls clear/start of TM1 by software. This bit is always 0 when writing. 0 : Continues counting 1 : Clears TM1 and resumes counting																																																							
4	IMS1	Interrupt Mode Select Selects interrupt source. 0 : Interrupt occurs by overflow of TM1 1 : Interrupt occurs by INTP14 signal																																																							
3 to 0	PRM13 to PRM10	Prescaler Clock Mode Selects the count clock frequency. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th style="width: 10%;">PRM13</th> <th style="width: 10%;">PRM12</th> <th style="width: 10%;">PRM11</th> <th style="width: 10%;">PRM10</th> <th style="width: 60%;">Count Clock</th> </tr> </thead> <tbody> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;"><math>\phi</math></td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;"><math>\phi/2</math></td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;"><math>\phi/4</math></td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;"><math>\phi/8</math></td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;"><math>\phi/16</math></td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;"><math>\phi/32</math></td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;"><math>\phi/64</math></td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;"><math>\phi/128</math></td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;"><math>\phi/256</math></td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">T11 input</td></tr> </tbody> </table> <p style="margin-top: 10px;"><b>Caution</b> Do not change the count clock frequency while the timer operates.  <b>Remark</b> <math>\phi</math>: Internal system clock</p>	PRM13	PRM12	PRM11	PRM10	Count Clock	0	0	0	0	$\phi$	0	0	0	1	$\phi/2$	0	0	1	0	$\phi/4$	0	0	1	1	$\phi/8$	0	1	0	0	$\phi/16$	0	1	0	1	$\phi/32$	0	1	1	0	$\phi/64$	0	1	1	1	$\phi/128$	1	0	0	0	$\phi/256$	1	1	1	1	T11 input
PRM13	PRM12	PRM11	PRM10	Count Clock																																																					
0	0	0	0	$\phi$																																																					
0	0	0	1	$\phi/2$																																																					
0	0	1	0	$\phi/4$																																																					
0	0	1	1	$\phi/8$																																																					
0	1	0	0	$\phi/16$																																																					
0	1	0	1	$\phi/32$																																																					
0	1	1	0	$\phi/64$																																																					
0	1	1	1	$\phi/128$																																																					
1	0	0	0	$\phi/256$																																																					
1	1	1	1	T11 input																																																					

(5) Timer control register 20 to 24 (TMC20 to TMC24)

TMC2n specifies count enable/disable and controls the count clock of TM2n.

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
TMC20	CE20	IMS20	CS20	0	PRM203	PRM202	PRM201	PRM200	Address FFFFFF2A0H	After reset 01H
TMC21	CE21	IMS21	CS21	0	PRM213	PRM212	PRM211	PRM210	Address FFFFFF2C0H	After reset 01H
TMC22	CE22	IMS22	CS22	0	PRM223	PRM222	PRM221	PRM220	Address FFFFFF2E0H	After reset 01H
TMC23	CE23	IMS23	CS23	0	PRM233	PRM232	PRM231	PRM230	Address FFFFFF300H	After reset 01H
TMC24	CE24	IMS24	CS24	0	PRM243	PRM242	PRM241	PRM240	Address FFFFFF320H	After reset 01H

Bit Position	Bit Name	Function																																																							
7	CE2n	Count Enable Specifies enable/disable of timer count. 0 : Timer count disabled (stops at TM2n = 0000H) 1 : Timer count enabled																																																							
6	IMS2n	Interrupt Mode Select Selects interrupt source. 0 : Interrupt is generated by coincidence signal of the compare register (CM2n) 1 : Interrupt is generated by INTP2n signal																																																							
5	CS2n	Clear & Start Controls clear/start of TM2n by software. This bit is set to 0 when reading. 0 : Continues counting 1 : Clears TM2n and resumes counting																																																							
3 to 0	PRM2n3 to PRM2n0	Prescaler Clock Mode Selects the count clock frequency. <table border="1" style="margin-top: 10px;"> <thead> <tr> <th>PRM2n3</th> <th>PRM2n2</th> <th>PRM2n1</th> <th>PRM2n0</th> <th>Count Clock</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td><math>\phi/2</math></td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td><math>\phi/4</math></td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td><math>\phi/8</math></td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>0</td> <td><math>\phi/16</math></td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>1</td> <td><math>\phi/32</math></td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>0</td> <td><math>\phi/64</math></td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>1</td> <td><math>\phi/128</math></td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td><math>\phi/256</math></td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>TI2n input</td> </tr> <tr> <td colspan="4">Others</td> <td>Setting prohibited</td> </tr> </tbody> </table> <p><b>Caution</b> Do not change the count clock frequency while the timer operates. <b>Remark</b> <math>\phi</math>: Internal system clock</p>	PRM2n3	PRM2n2	PRM2n1	PRM2n0	Count Clock	0	0	0	1	$\phi/2$	0	0	1	0	$\phi/4$	0	0	1	1	$\phi/8$	0	1	0	0	$\phi/16$	0	1	0	1	$\phi/32$	0	1	1	0	$\phi/64$	0	1	1	1	$\phi/128$	1	0	0	0	$\phi/256$	1	1	1	1	TI2n input	Others				Setting prohibited
PRM2n3	PRM2n2	PRM2n1	PRM2n0	Count Clock																																																					
0	0	0	1	$\phi/2$																																																					
0	0	1	0	$\phi/4$																																																					
0	0	1	1	$\phi/8$																																																					
0	1	0	0	$\phi/16$																																																					
0	1	0	1	$\phi/32$																																																					
0	1	1	0	$\phi/64$																																																					
0	1	1	1	$\phi/128$																																																					
1	0	0	0	$\phi/256$																																																					
1	1	1	1	TI2n input																																																					
Others				Setting prohibited																																																					

**Remark** n = 0 to 4

**(6) Timer control register 3 (TMC3)**

TMC3 specifies count enable/disable and controls the count clock of TM3.

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
TMC3	CE3	0	CS3	CMS3	PRM33	PRM32	PRM31	PRM30	Address FFFFF340H	After reset 01H

Bit Position	Bit Name	Function																																																		
7	CE3	Count Enable Specifies enable/disable of timer count. 0 : Timer count disabled (stops at TM3 = 0000H) 1 : Timer count enabled																																																		
5	CS3	Clear & Start Controls clear/start of TM3 by software. This bit is always 0 when reading. 0 : Continues counting 1 : Clears TM3 and resumes counting																																																		
4	CMS3	Capture/Compare Mode Select Selects capture/compare mode. 0 : Operates as a capture register 1 : Operates as a compare register																																																		
3 to 0	PRM33 to PRM30	Prescaler Clock Mode Selects the count clock frequency. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">PRM33</th> <th style="width: 10%;">PRM32</th> <th style="width: 10%;">PRM31</th> <th style="width: 10%;">PRM30</th> <th style="width: 60%;">Count Clock</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>\phi/2</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>\phi/4</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>\phi/8</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>\phi/16</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>\phi/32</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>\phi/64</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>\phi/128</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>\phi/256</math></td> </tr> <tr> <td colspan="4" style="text-align: center;">Others</td> <td style="text-align: center;">Setting prohibited</td> </tr> </tbody> </table> <p><b>Caution</b> Do not change the count clock frequency while the timer operates.  <b>Remark</b> <math>\phi</math>: Internal system clock</p>	PRM33	PRM32	PRM31	PRM30	Count Clock	0	0	0	1	$\phi/2$	0	0	1	0	$\phi/4$	0	0	1	1	$\phi/8$	0	1	0	0	$\phi/16$	0	1	0	1	$\phi/32$	0	1	1	0	$\phi/64$	0	1	1	1	$\phi/128$	1	0	0	0	$\phi/256$	Others				Setting prohibited
PRM33	PRM32	PRM31	PRM30	Count Clock																																																
0	0	0	1	$\phi/2$																																																
0	0	1	0	$\phi/4$																																																
0	0	1	1	$\phi/8$																																																
0	1	0	0	$\phi/16$																																																
0	1	0	1	$\phi/32$																																																
0	1	1	0	$\phi/64$																																																
0	1	1	1	$\phi/128$																																																
1	0	0	0	$\phi/256$																																																
Others				Setting prohibited																																																

**(7) Timer output control registers 0 and 1 (TOC0, TOC1)**

TOC0, TOC1 control the timer outputs from the TO00, TO01, and TO20 to TO24 pins.

These registers can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
TOC0	0	0	ENTO20	ALV20	ENTO01	ALV01	ENTO00	ALV00	Address FFFFFF232H	After reset 00H
TOC1	ENTO24	ALV24	ENTO23	ALV23	ENTO22	ALV22	ENTO21	ALV21	Address FFFFFF234H	After reset 00H

Bit Position	Bit Name	Function
7, 5, 3, 1	ENTOn	Enable TO pin Enables corresponding timer output (TO <sub>n</sub> ). 0: Timer output is disabled. The anti-phase levels of ALV <sub>n</sub> bit (inactive levels) are output from TO <sub>n</sub> pins. Even if coincidence signal is generated from corresponding compare register, levels of TO <sub>n</sub> pins do not change. 1: Timer output function is enabled. Timer output changes when coincidence signal is generated from corresponding compare register. After the timer output has been enabled before the first coincidence signal is generated, the anti-phase levels of ALV <sub>n</sub> bit (inactive levels) are output.
6, 4, 2, 0	ALVn	Active Level TO pin Specifies the active level of timer output. 0: Active-low 1: Active-high

- Remarks**
1. The flip-flops of the TO00 and TO01 outputs is a reset priority.
  2. n = 00, 01, 20 to 24

**Caution** The TO00, TO01 outputs are not changed by the external interrupt signals (INTP00 to INTP03). When using the TO00 and TO01 signals, specify the capture/compare registers as compare registers CMS00 to CMS03 (set CMS00 to CMS03 to "1").

**(8) Timer overflow status register (TOVS)**

The overflow flags TM0 to TM3 are assigned.

This register can be read/written in 8- or 1-bit units.

By testing and resetting the TOVS register via software, occurrence of an overflow can be polled.

TOVS	7	6	5	4	3	2	1	0	Address FFFFF230H	After reset 00H
	OVF3	OVF24	OVF23	OVF22	OVF21	OVF20	OVF1	OVF0		

Bit Position	Bit Name	Function
7 to 0	OVFn	<p>Overflow Flag TMn overflow flag. 0: No overflow 1: Overflow</p> <p>From TM0 and TM1, the interrupt requests (INTOV0 and INTOV1) are generated for the interrupt controller in synchronization with the overflow. However, the interrupt operation and TOVS are independent from each other, and the overflow flags (OVF0 and OVF1) from TM0 and TM1 can be rewritten in the same manner as in other overflow flags. At this time, the interrupt request flags (OVF0 and OVF1) corresponding to INTOV0 and INTOV1 are not affected.</p> <p>No transmission is executed to the TOVS register during access from the CPU. Therefore, even if an overflow occurs when the TOVS register is being read, the overflow flag value will not be updated and this overflow condition will be reflected the next time the TOVS register is read.</p>

**Remark** n = 0, 1, 20 to 24, and 3

**(9) External interrupt mode registers 1 to 4, 7 (INTM1 to INTM4, INTM7)**

These registers set the following 3 types of valid edges:

- Sets valid edge of external interrupt request signal (INTP) when using CP10 to CP13 (timer 1), CC3, CP3 (timer 3) as capture registers.
- Sets valid edges at external count clock input (TI) of timer 0, timer 1, and timer 2.
- Sets valid edges at timer clear (TCLR0) of timer 0.

For the details, refer to **CHAPTER 5 INTERRUPT/EXCEPTION PROCESSING FUNCTION**.

**(10) Event divide counter 0 to 2 (EDV0 to EDV2)**

Counts valid edges detected by the INTM1 to INTM3 registers. For the details, refer to **5.3 Maskable Interrupt**.

**(11) Event divide control register 0 to 2 (EDVC0 to EDVC2)**

Sets the frequency division ratio of event divide counter (EDV0 to EDV2). For the details, refer to **5.3. Maskable Interrupt**.

**(12) Event selection register (EVS)**

Selects INTP signal to input to the EDV2 register. For the details, refer to **5.3 Maskable Interrupt**.

## 7.4 Timer 0 Operation

### 7.4.1 Count operation

Timer 0 functions as a 24-bit interval timer or event counter, as specified by timer control registers 00 to 02 (TMC00 to TMC02).

Timer 0 performs counting up by count clock. Start/stop of counting is controlled by the CE0 bit of timer control register 00 (TMC00).

#### (1) Start counting

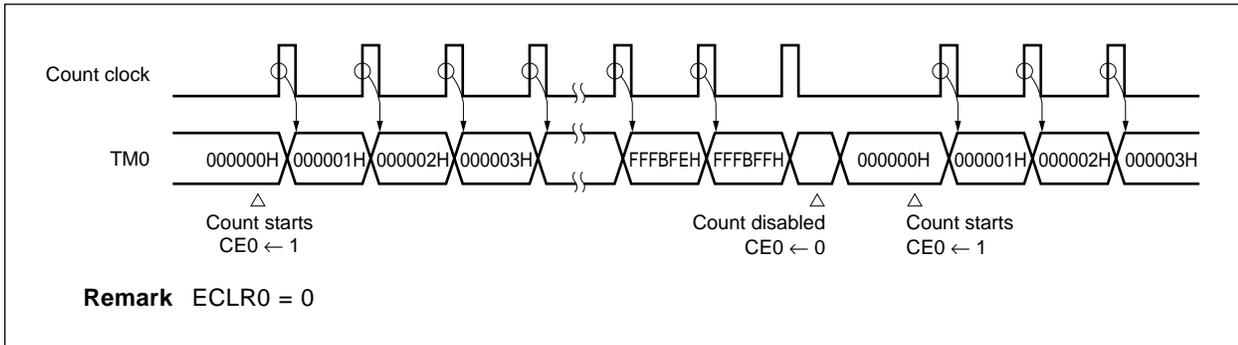
Timer 0 starts counting by setting the CE0 bit to 1 while the ECLR0 bit of the TMC02 register is 0. However, when the ECLR0 bit is 1, timer 0 does not start counting until the TCLR0 signal is input. Therefore, it does not start counting by setting ECLR = 0 after setting CE0 = 1 while ECLR0 = 1.

Writing 1 to TM0 during counting operations (CE0 = 1) does not clear the TM0 register, and timer 0 continues counting.

#### (2) Stop counting

Timer 0 stops counting by setting the CE0 bit to 0. If the OST bit of the TMC00 register is set to 1, timer 0 stops operation after occurrence of overflow. However, the value of the timer register can immediately be cleared by setting CE0 = 0.

Figure 7-1. Basic Operation of Timer 0



### 7.4.2 Count clock selection

An internal or external count clock frequency can be input to timer 0. Which count clock frequency is used is selected by the PRM00 to PRM03 bits of the TMC00 register.

**Caution** Do not change the count clock frequency while the timer operates.

#### (1) Internal count clock

An internal count clock frequency is selected by the PRM bit of the TMC00 register, from  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ , and  $\phi/256$ .

PRM03	PRM02	PRM01	PRM00	Internal Count Clock
0	0	0	1	$\phi/2$
0	0	1	0	$\phi/4$
0	0	1	1	$\phi/8$
0	1	0	0	$\phi/16$
0	1	0	1	$\phi/32$
0	1	1	0	$\phi/64$
0	1	1	1	$\phi/128$
1	0	0	0	$\phi/256$

#### (2) External count clock

The signal input to the TI0 pin is counted. At this time, timer 0 operates as an event counter. To set an external count clock see the table as follows.

PRM03	PRM02	PRM01	PRM00	External Count Clock
1	1	1	1	TI0 input

The valid edge of TI0 is specified by the ES bit of the INTM2 register.

ES051	ES050	Valid Edge
0	0	Falling edge
0	1	Rising edge
1	0	RFU (reserved)
1	1	Both rising and falling edges

**7.4.3 Overflow**

If the TM0 register overflows as a result of counting the count clock frequency to FFFFFFFH, a flag is set to the OVF0 flag of the TOVS registers, and an overflow interrupt (INTOV0) is generated. The value of the OVF0 flag is retained until it is changed by user application.

The operation of the TM0 register after occurrence of overflow is determined by the OST0 bit.

**(1) Operation after occurrence of overflow when OST0 = 0**

The TM0 register continues counting.

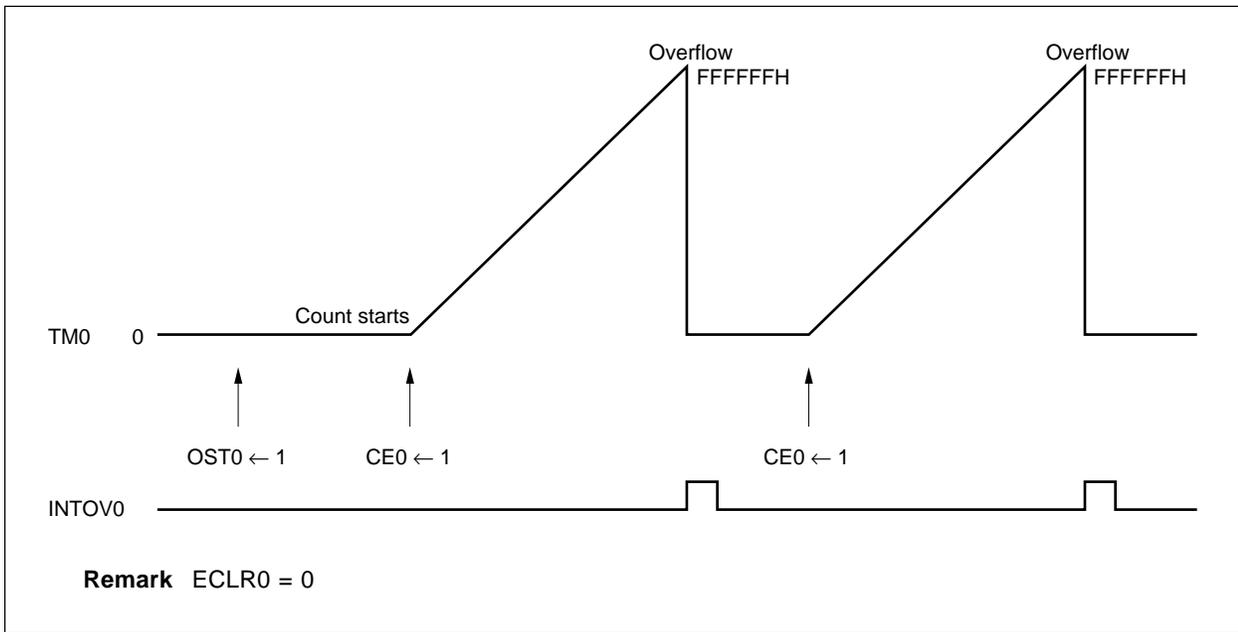
**(2) Operation after occurrence of overflow when OST0 = 1**

TM0 = 000000H is retained, and the TM0 register stops counting. At this time TM0 stops with CE0 = 1. Perform the following to resume counting.

- When ECLR0 = 0 : write 1 to CE0 bit
- When ECLR0 = 1 : trigger input to timer clear pin (TCLR0)

The operation is not affected even if the CE0 bit is set to 1 during count operation.

**Figure 7-2. Operation after Occurrence of Overflow (when ECLR0 = 0, OST0 = 1)**



#### 7.4.4 Clearing/starting timer

There are three methods of clearing/starting timer 0: by overflow, by TCLR0 signal input, and by CC03 coincidence.

##### (1) Clearing/starting by overflow

For the details of the operation, refer to **7.4.3 Overflow**.

##### (2) Clearing/starting by TCLR0 signal input

Timer 0 usually starts the count operation when the CE0 bit of the TMC00 register is set to 1. It is also possible to clear TM0 and start the count operation by using external input TCLR0.

When the valid edge is input to TCLR0 after ECLR0 = 1, OST0 = 0, and the CE0 bit is set to 1, the count operation is started. If the valid edge is input to TCLR0 during operation, TM0 clears its value and then resume the count operation (refer to **Figure 7-3**).

When the valid edge is input to the TCLR0 signal after ECLR0 = 0, OST0 = 1, and the CE0 bit is set to 1, the count operation is started. When TM0 overflows, the count operation is stopped once and is not resumed until the valid edge is input to TCLR0. If the valid edge of TCLR0 is detected during count operation, TM0 is cleared and continues counting (refer to **Figure 7-4**). Timer 0 does not resume counting even if the CE0 bit is set to 1 after occurrence of overflow. When CE0 = 0, TCLR0 input is invalid.

**Figure 7-3. Clearing/Starting Timer by TCLR0 Signal Input (when ECLR0 = 1, CCLR0 = 0, OST0 = 0)**

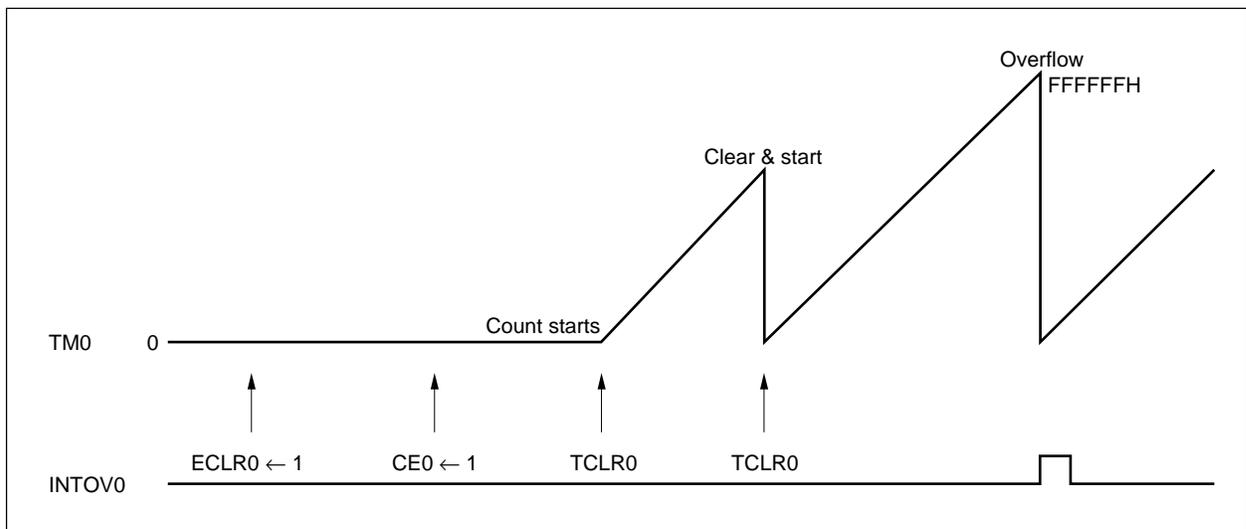
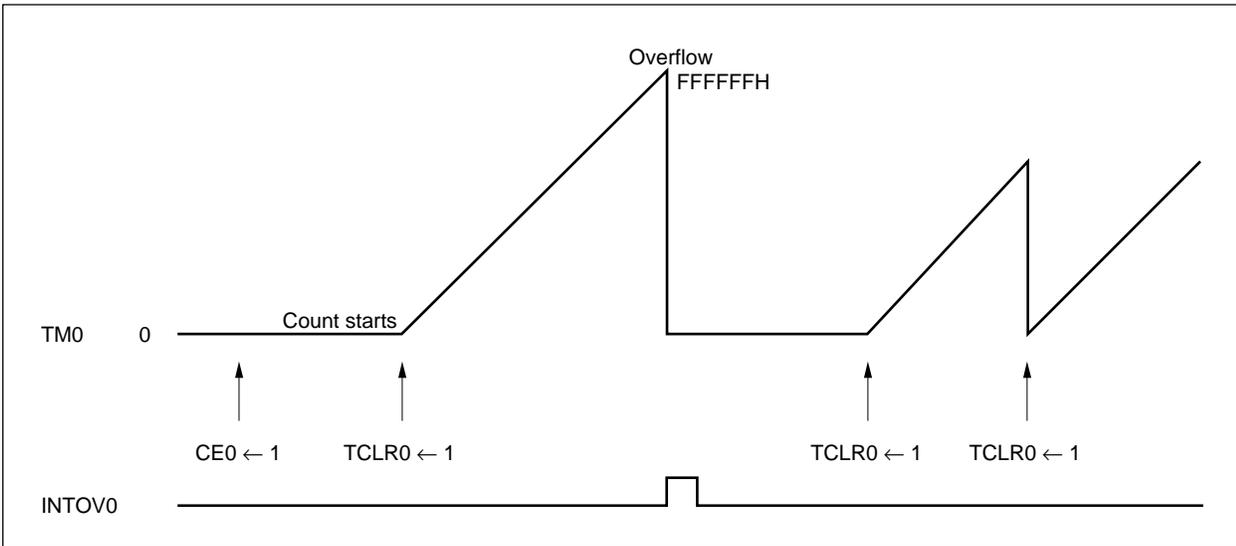


Figure 7-4. Relations between Clear/Start by TCLR0 Signal Input and Overflow (when ECLR0 = 1, OST0 = 1)



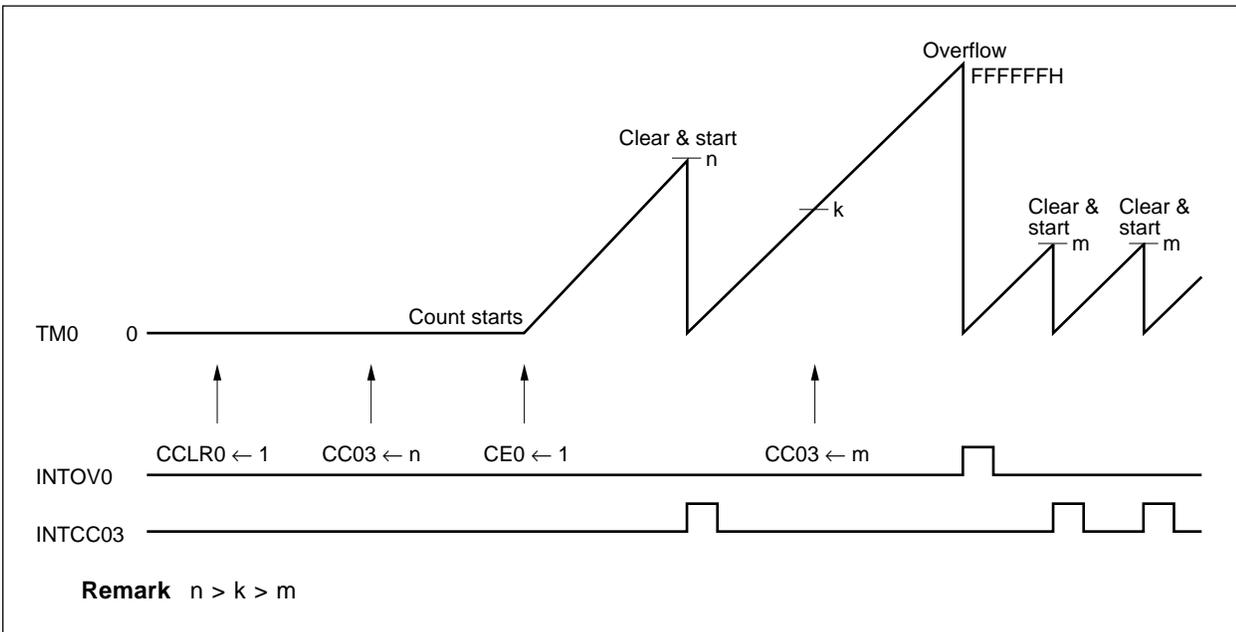
**(3) Clearing/starting by CC03 match**

Timer 0 usually starts the count operation when CCLR = 1, CMS03 = 1, and the CE0 bits of the TMC00 registers are set to 1. It is also possible to clear TM0 and start the count operation by generation of CC03 match (INTCC03).

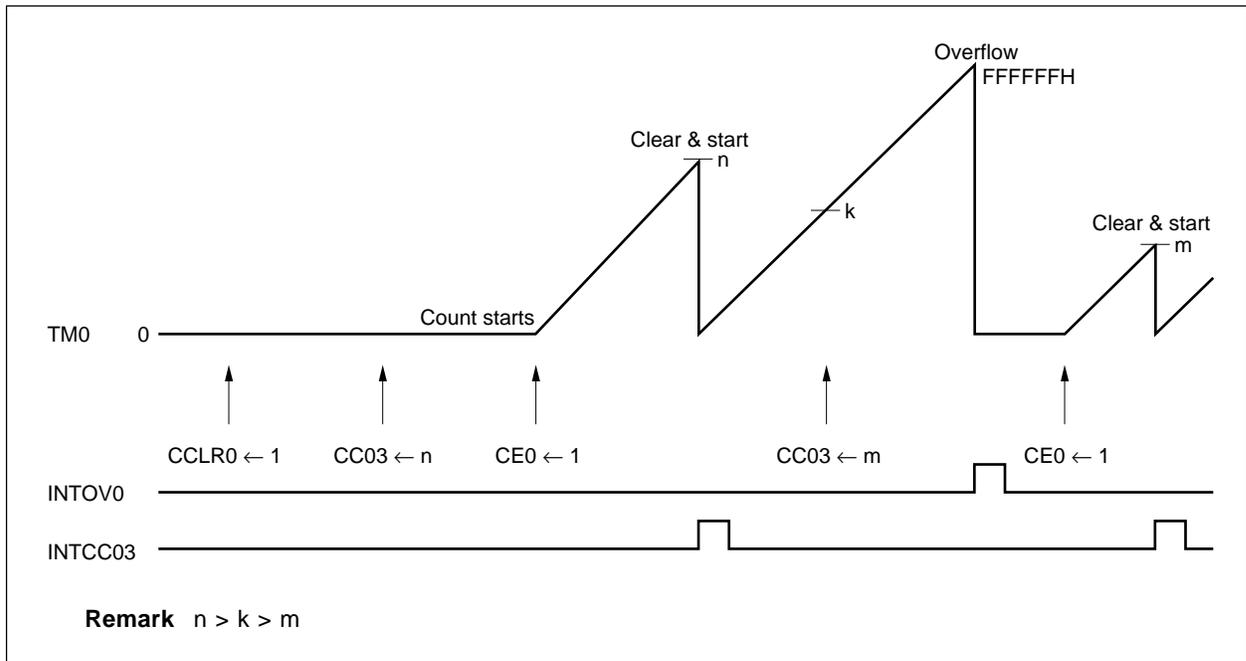
When CCLR0 = 1, CMS03 = 1, and the CE0 bit is set to 1, the count operation is started. If CC03 match is generated during operation, TM0 clears its value and then resumes the count operation (refer to **Figure 7-5**).

If a value smaller than the current count value of TM0 is set to CC03 during count operation, overflow of TM0 occurs (refer to **Figure 7-6**). For the operation after occurrence of overflow, refer to **7.4.3 Overflow**.

Figure 7-5. Clearing/Starting Timer by CC03 Match (when CCLR0 = 1, OST0 = 0)



**Figure 7-6. Relations between Clear/Start by CC03 Coincidence and Overflow Operation (when CCLR0 = 1, OST0 = 1)**



#### 7.4.5 Capture operation

When the TMC01 register is set to a capture register, the capture/compare registers (CC00 to CC03) perform capture operations that capture and hold the count values of TM0 and load them to a capture register in asynchronization with an external trigger. The valid edge from the external interrupt request input pin INTP0n is used as the capture trigger. In synchronization with this capture trigger signal, the count values of TM0 during counting are captured and loaded to the capture register and the interrupt request INTCC0n is simultaneously issued. The value of the capture register is retained until the next capture trigger is generated.

When the capture timing to a capture register and write operation to a register by instruction are in contention, the latter is given priority and the capture operation is ignored.

**Table 7-2. Capture Trigger Signal to 24-Bit Capture Register**

Capture Trigger Signal	Capture Register	Interrupt Request
INTP00	CC00/CC00L	INTCC00
INTP01	CC01/CC01L	INTCC01
INTP02	CC02/CC02L	INTCC02
INTP03	CC03/CC03L	INTCC03

The valid edge of the capture trigger is set by the external interrupt mode register (INTM1).

When both the rising and falling edges are specified as the capture trigger, the width of an externally input pulse can be measured. If either the rising or falling edge is specified as the capture trigger, the frequency of the input pulse can be measured.

Figure 7-7. Example of TM0 Capture Operation

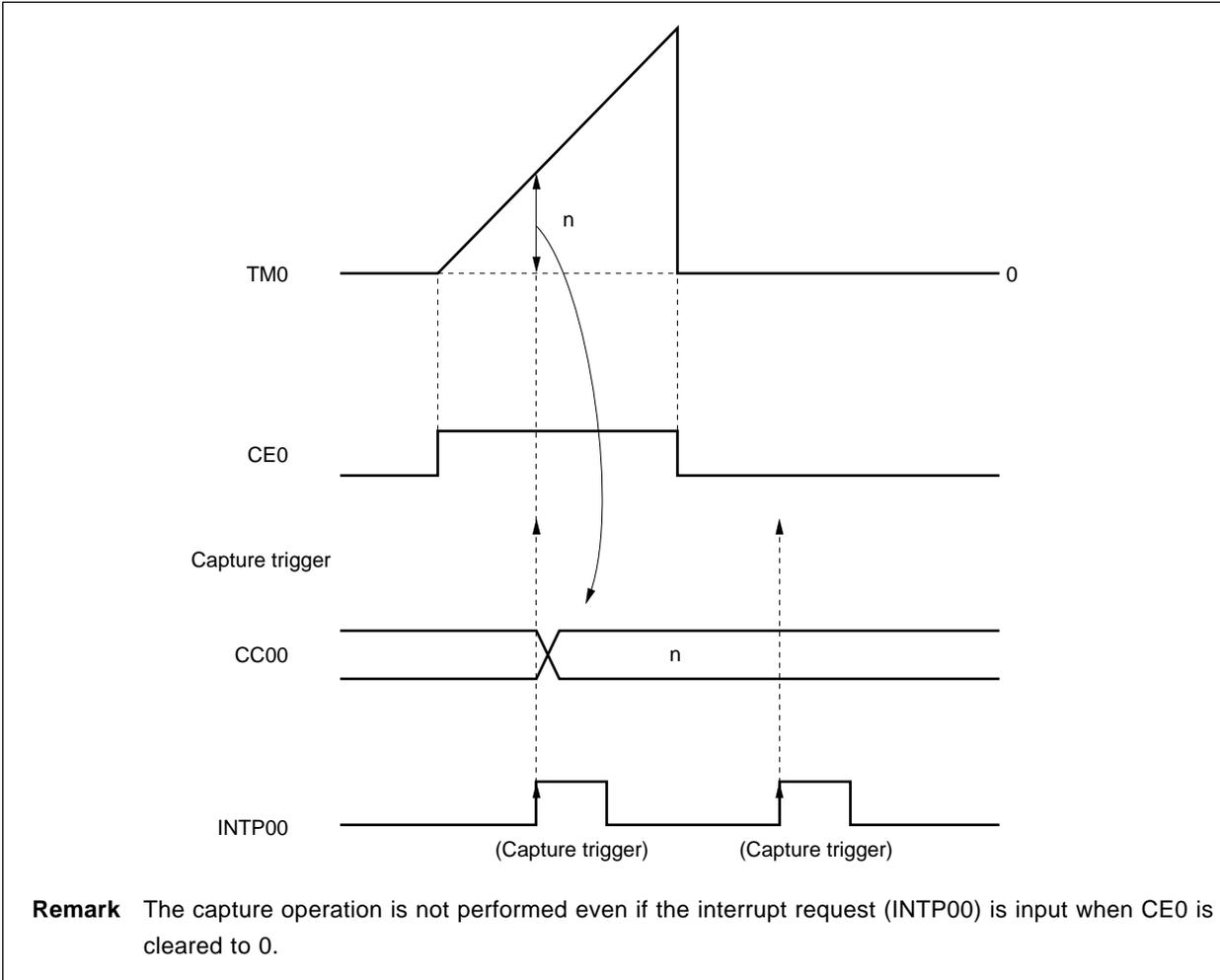
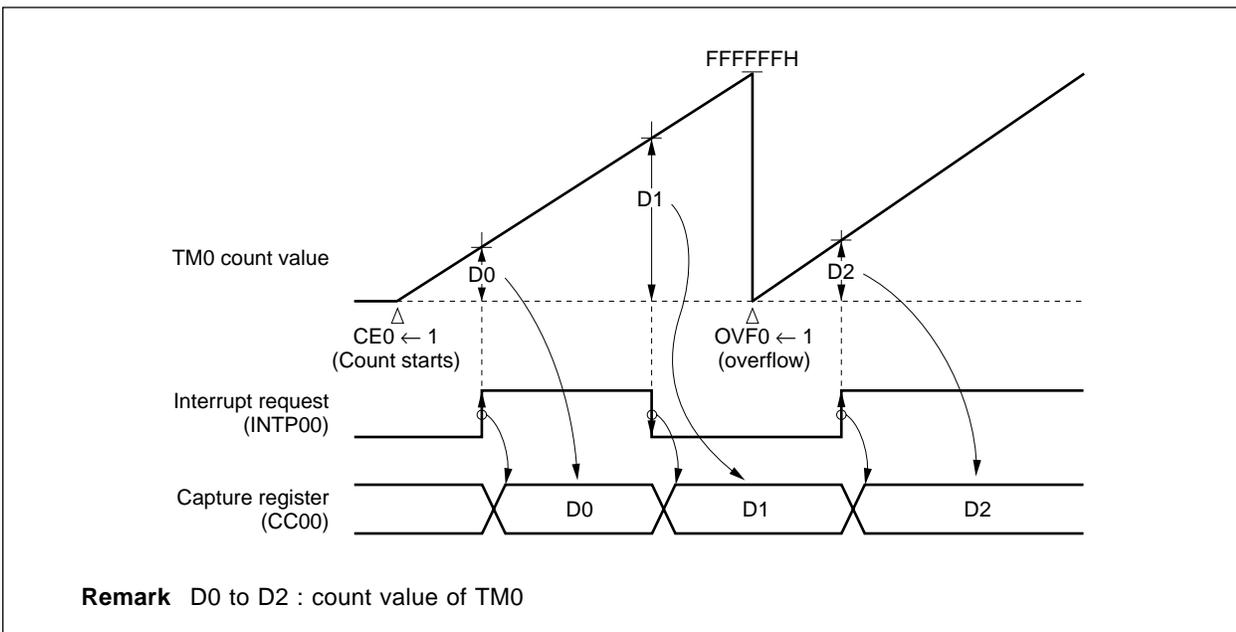


Figure 7-8. Example of TM0 Capture Operation (when both edges are specified)



**7.4.6 Compare operation**

When the TMC01 register is set as a compare register, the capture/compare registers (CC00 to CC03) perform a comparison between the value of the compare register with the count values of TM0.

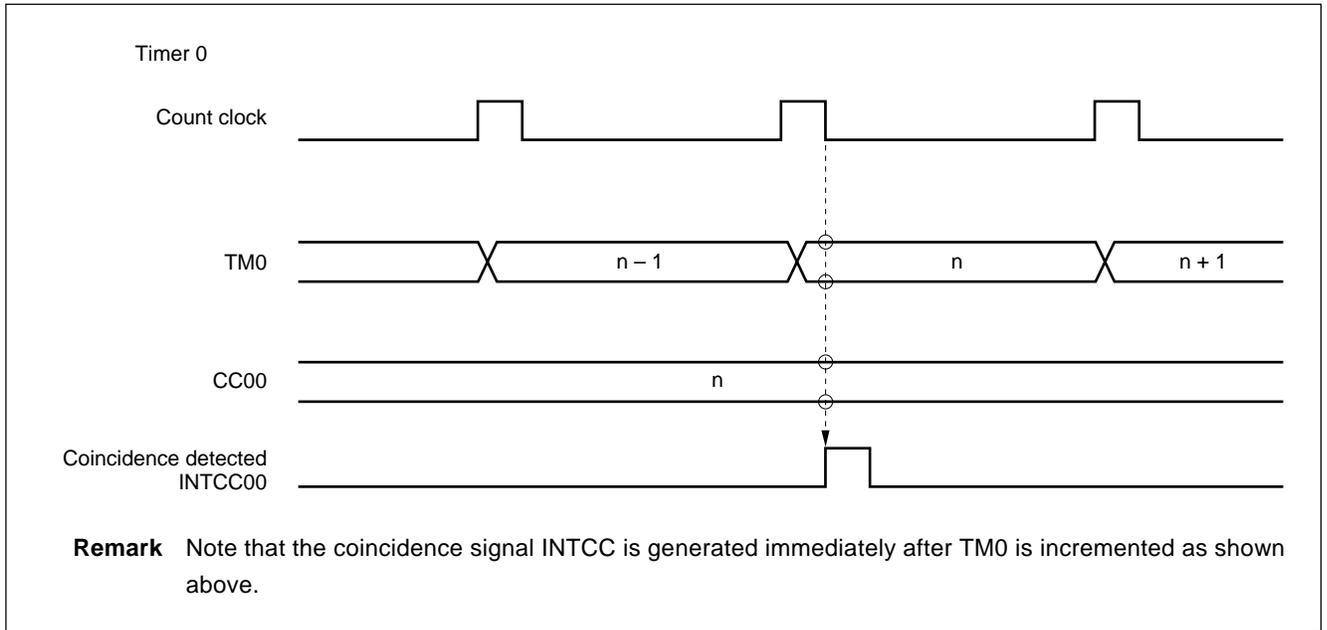
When the count values of TM0 coincide with the value of the compare register programmed in advance, a coincidence signal is sent to the output control circuit (refer to **Figure 7-9**). The levels of the timer output pins (TO) can be changed by the coincidence signal, and an interrupt request signal INTCC can be generated at the same time (n = 00, 01).

**Table 7-3. Interrupt Request Signal from 24-Bit Compare Register**

Compare Register	Interrupt Request	Compare Match Trigger
CC00/CC00L	INTCC00	TO00 (S)
CC01/CC01L	INTCC01	TO00 (R)
CC02/CC02L	INTCC02	TO01 (S)
CC03/CC03L	INTCC03	TO01 (R), A/D converter

**Remark** S/R: set/reset

**Figure 7-9. Example of Compare Operation**



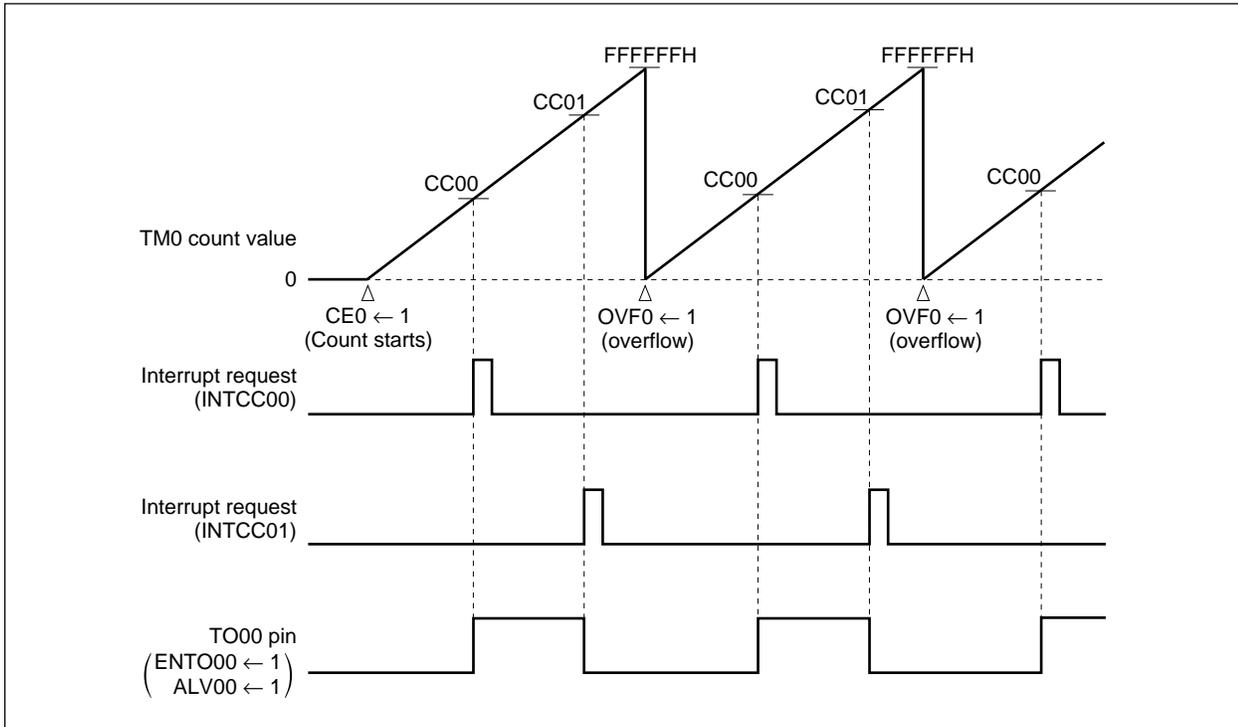
Timer 0 has two timer output pins: TO00 and TO01.

The count values of TM0 are compared with the values of CC02. When the two values coincide, the output level of the TO01 pin is set. The count values of TM0 are also compared with the values of CC03. When the two values coincide, the output levels of the TO01 pin are reset.

Similarly, the count values of TM0 are compared with the values of CC00. When the two values coincide, the output levels of the TO00 pin are set. The count values of TM0 are also compared with the values of CC01. When the two values coincide, the output levels of the TO00 pin are reset.

The output levels of the TOn pins can be specified by the TOC0 register (n = 00, 01).

**Figure 7-10. Example of TM0 Compare Operation (set/reset output mode)**



## 7.5 Timer 1 Operation

### 7.5.1 Count operation

Timer 1 functions as a 24-bit free-running timer or event counter, as specified by timer control register 1 (TMC1).

Timer 1 performs counting up by count clock. Start/stop of counting is controlled by the CE1 bit of timer control register 1 (TMC1).

#### (1) Start counting

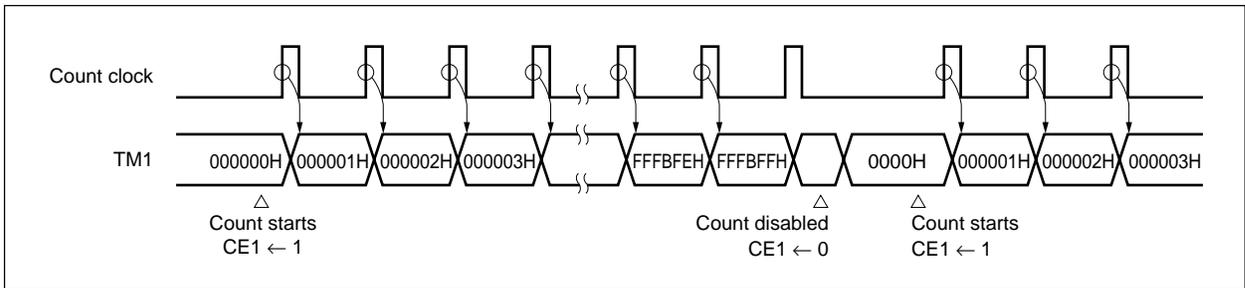
Timer 1 starts counting by setting the CE1 bit to 1.

Writing 1 to TM1 during counting operations (CE1 = 1) does not clear the TM1 register, and timer 1 continues counting.

#### (2) Stop counting

Timer 1 stops counting by setting the CE1 bit to 0. If the OST bit of the TMC1 register is set to 1, timer 1 stops operation after occurrence of overflow. However, the value of the timer register can immediately be cleared by setting CE1 = 0.

Figure 7-11. Basic Operation of Timer 1



### 7.5.2 Count clock selection

An internal or external count clock frequency can be input to timer 1. Which count clock frequency is used is selected by the PRM10 to PRM13 bits of the TMC1 register.

**Caution** Do not change the count clock frequency while the timer operates.

#### (1) Internal count clock

An internal count clock frequency is selected by the PRM bits of the TMC1 register, from  $\phi$ ,  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ , and  $\phi/256$ .

PRM13	PRM12	PRM11	PRM10	Internal Count Clock Frequency
0	0	0	0	$\phi$
0	0	0	1	$\phi/2$
0	0	1	0	$\phi/4$
0	0	1	1	$\phi/8$
0	1	0	0	$\phi/16$
0	1	0	1	$\phi/32$
0	1	1	0	$\phi/64$
0	1	1	1	$\phi/128$
1	0	0	0	$\phi/256$

**(2) External count clock**

The signal input to the TI1 pins is counted. At this time, timer 1 operates as an event counter. To set an external count clock, see the table below.

PRM13	PRM12	PRM11	PRM10	External Count Clock
1	1	1	1	TI1 input

The valid edge of TI1 is specified by the ES bits of the INTM3 register.

ES141	ES140	Valid Edge
0	0	Falling edge
0	1	Rising edge
1	0	RFU (reserved)
1	1	Both rising and falling edges

**7.5.3 Overflow**

If overflow occurs as a result of counting the TM1 register count clock frequency to FFFFFFFH, a flag is set to the OVF1 bits of the TOVS register, and an overflow interrupt (INTOV1) is generated.

The value of the OVF1 flag is retained until it is changed by user application.

The operation of the TM1 register after occurrence of overflow is determined by the OST1 bit.

**(1) Operation after occurrence of overflow when OST1 = 0**

The TM1 register continues counting.

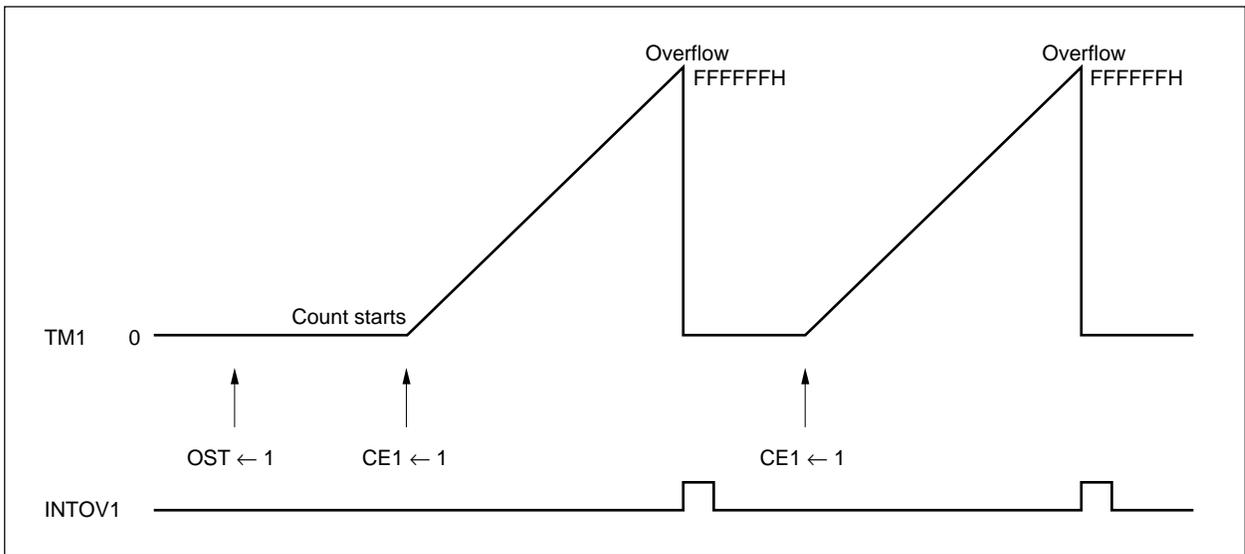
**(2) Operation after occurrence of overflow when OST1 = 1**

TM1 = 000000H is retained, and the TM1 register stops counting. At this time TM1 stops with CE1 = 1. Perform the following to resume counting.

- Write 1 to CE1 bit
- Write 1 to CS1 bit

The operation is not affected even if the CE1 bit is set to 1 during count operation.

Figure 7-12. Operation after Occurrence of Overflow (OST1 = 1)



**7.5.4 Clearing/starting timer**

There are two methods of clearing/starting timer 1: by overflow and by software.

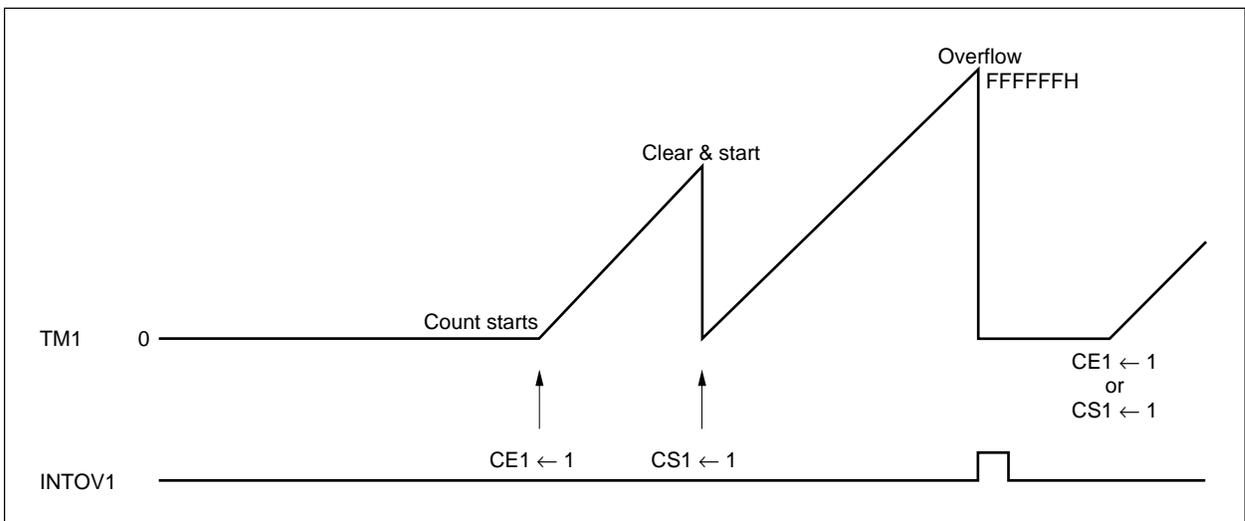
**(1) Clearing/starting by overflow**

For the details of the operation, refer to 7.5.3 Overflow.

**(2) Clearing/starting by software**

When the CS1 bit is set to 1 by software, the TM1 register clears its value and starts counting from 0. However, this setting of the bit is valid only when the value of the CE1 bit is 1.

Figure 7-13. Clearing/Starting Timer by Software (when OST1 = 1)



7.5.5 Capture operation

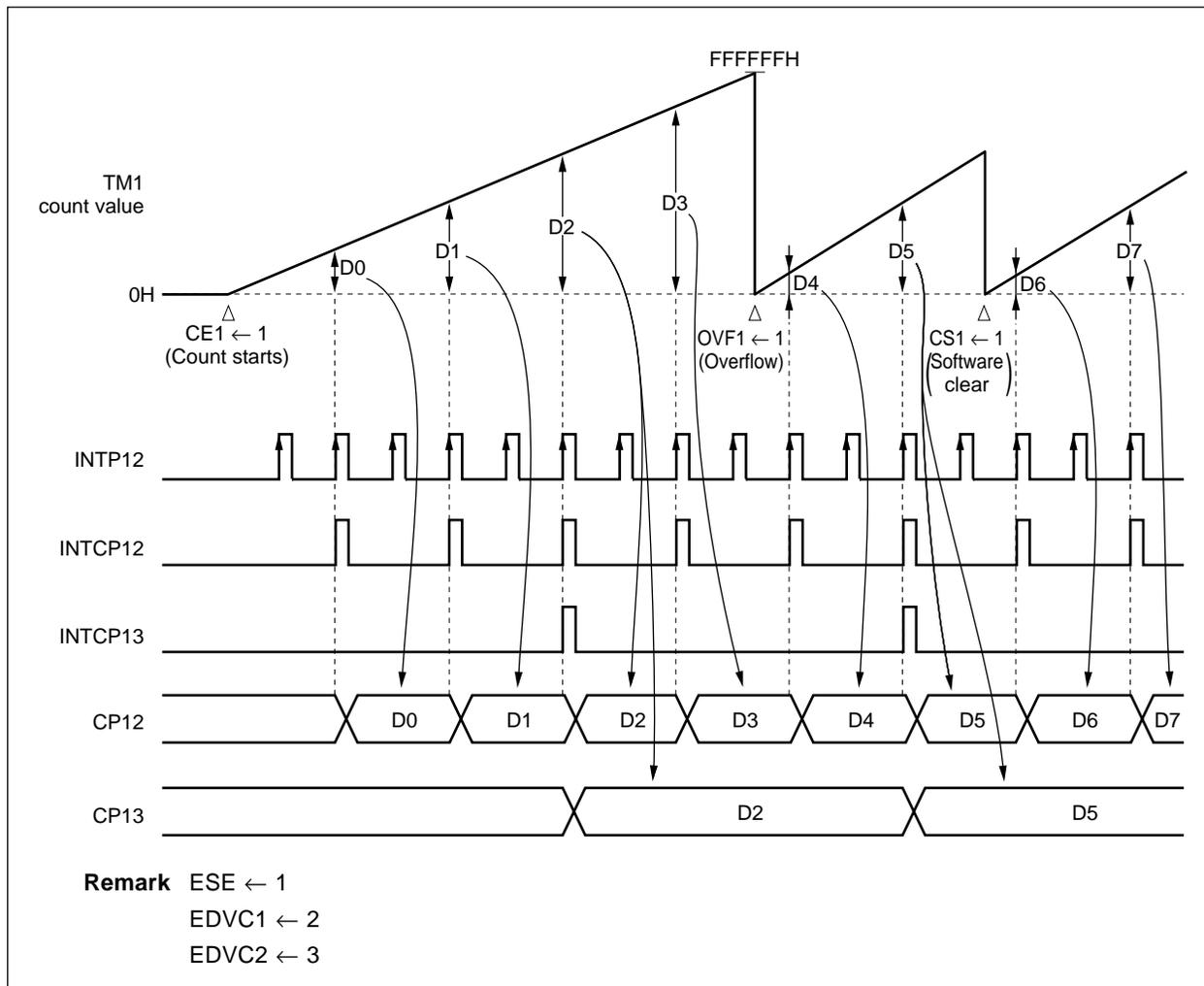
A capture operation that captures and holds the count values of TM1 and loads them to a capture register in asynchronization with an external trigger can be performed. The trigger divided by the valid edge from the external interrupt request input pin INTP1n is used as the capture trigger. In synchronization with this capture trigger signal, the count values of TM1n during counting, are captured and loaded to the capture register and the interrupt request INTCP1n is simultaneously issued. The value of the capture register is retained until the next capture trigger is generated.

Table 7-4. Capture Trigger Signal to 24-Bit Capture Register

Capture Trigger Signal	Capture Register	Interrupt Request
INTP10	CP10/CP10L	INTCP10
Divide of INTP11	CP11/CP11L	INTCP11
Divide of INTP12	CP12/CP12L	INTCP12
Divide of INTP12/INTP13	CP13/CP13L	INTCP13

The valid edge of the INTP1n input is set by the external interrupt mode registers (INTM2, INTM3). The frequency dividing ratio of the INTCP11 to INTCP13 triggers is set by the EDVCn register and the EVS register. For the details, refer to 5.3.9 Frequency divider.

Figure 7-14. Example of TM1 Capture Operation



**7.5.6 Compare operation**

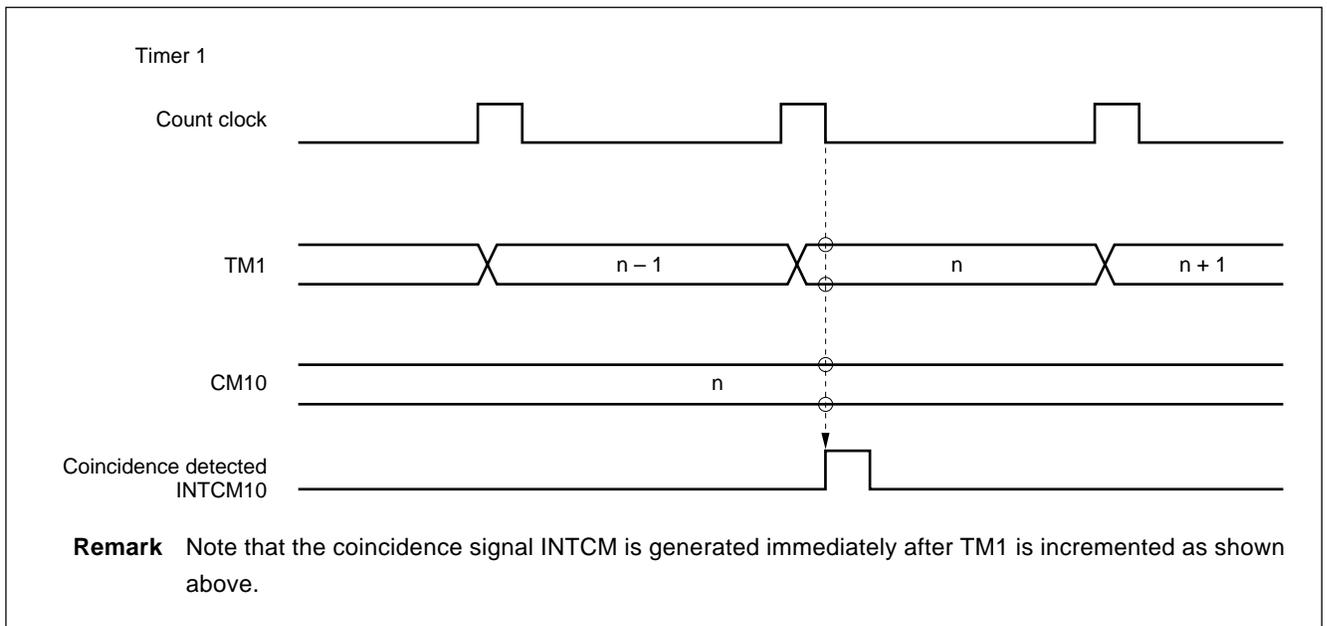
A comparison between the value in a compare register with the count values of TM1 can be performed.

When the count values of TM1 coincide with the value of the compare register programmed in advance, INTCM10 coinciding with CM10 is generated as a trigger of the real time output port. An interrupt request signal INTCM can be generated at the same time.

**Table 7-5. Interrupt Request Signal from 24-Bit Compare Register**

Compare Register	Interrupt Request	Compare Match Trigger
CM10/CM10L	INTCM10	Real time output port
CM11/CM11L	INTCM11	—

**Figure 7-15. Example of Compare Operation**



## 7.6 Timer 2 Operation

### 7.6.1 Count operation

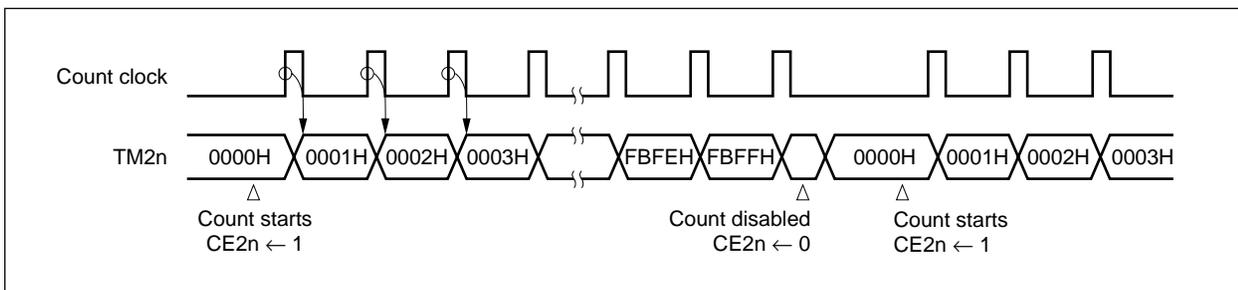
Timer 2 functions as a 16-bit interval timer. The operation is specified by the timer control registers 20 to 24 (TMC20 to TMC24).

The operation of timer 2 counts the internal count clocks ( $\phi/2$  to  $\phi/256$  or the external count clock (TI2n)) specified by the PRM2n0 to PRM2n3 bits of the TMC2n register.

If the count value of TM2n coincides with the value of CM2n, the value TM2n is cleared while simultaneously a coincidence interrupt (INTCM2n) is generated and the TO2n signal is output in toggle.

**Remark** n = 0 to 4

**Figure 7-16. Basic Operation of Timer 2**



### 7.6.2 Count clock selection

An internal or external count clock frequency can be input to timer 2. Which count clock frequency is used is selected by the PRM2n0 to PRM2n3 bits of the TMC2n register (n = 0 to 4).

**Caution** Do not change the count clock frequency while the timer operates.

#### (1) Internal count clock

An internal count clock frequency is selected by the PRM bits of the TMC2n register, from  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ , and  $\phi/256$ .

PRM2n3	PRM2n2	PRM2n1	PRM2n0	Internal Count Clock Frequency
0	0	0	1	$\phi/2$
0	0	1	0	$\phi/4$
0	0	1	1	$\phi/8$
0	1	0	0	$\phi/16$
0	1	0	1	$\phi/32$
0	1	1	0	$\phi/64$
0	1	1	1	$\phi/128$
1	0	0	0	$\phi/256$

**(2) External count clock**

The signal input to the TI2n pins are counted. At this time, timer 2 operates as an event counter. To set an external count clock see the table below.

PRM2n3	PRM2n2	PRM2n1	PRM2n0	External Count Clock
1	1	1	1	TI2n input

The valid edge of TI2n is specified by the ES bits of the INTM3 and the INTM4 registers.

ES2n1	ES2n0	Valid Edge
0	0	Falling edge
0	1	Rising edge
1	0	RFU (reserved)
1	1	Both rising and falling edges

**Remark** n = 0 to 4

**7.6.3 Overflow**

If TM2n overflows as a result of counting the internal count clock, a flag is set to the OVF2n bit of the TOVS register (n = 0 to 4).

**7.6.4 Clearing/starting timer**

There are two methods of clearing/starting timer 2: by coincidence with a compare register and by software.

**(1) Clearing/starting by coincidence signal of a compare register**

When the set value of the compare register (CM2n) and the value of TM2n coincide, TM2n clears its value at the next count clock and starts count operation (n = 0 to 4). At the same time, it generates an interrupt request signal (INTCM20 to INTCM24) and a timer output trigger.

The interval time set to a compare register can be calculated by the following expression:

$$(\text{Set value} + 1) \times \text{Count cycle}$$

For the details, refer to **7.6.5 Compare operation**.

**(2) Clearing/starting by software**

When the value of the CS2n bit of the TMC2n register is set to 1, TM2n clears its value at the next count clock and starts count operation (n = 0 to 4). However, the setting of this bit is valid only when the value of the CE2n bit is 1 (n = 0 to 4).

**7.6.5 Compare operation**

A comparison can be performed with the counter values of TM20 to TM24 and the compare registers (CM20 to CM24).

When the count value of TM2n coincides with the value of the compare register, a coincidence interrupt (INTCM2n) is generated. As a result, TM2n is cleared to 0 at the next count timing (refer to **Figure 7-17**). This function allows timer 2 to be used as an interval timer.

CM2n can be also set to 0. In this case, a coincidence is detected when TM2n overflows and is cleared to 0, and INTCM2n is generated. The value of TM2n is cleared to 0 at the next count timing, but INTCM2n is not generated when a coincidence occurs at this time (refer to **Figure 7-18**).

**Remark** n = 0 to 4

Figure 7-17. Operation with CM2n at 1 to FFFFH

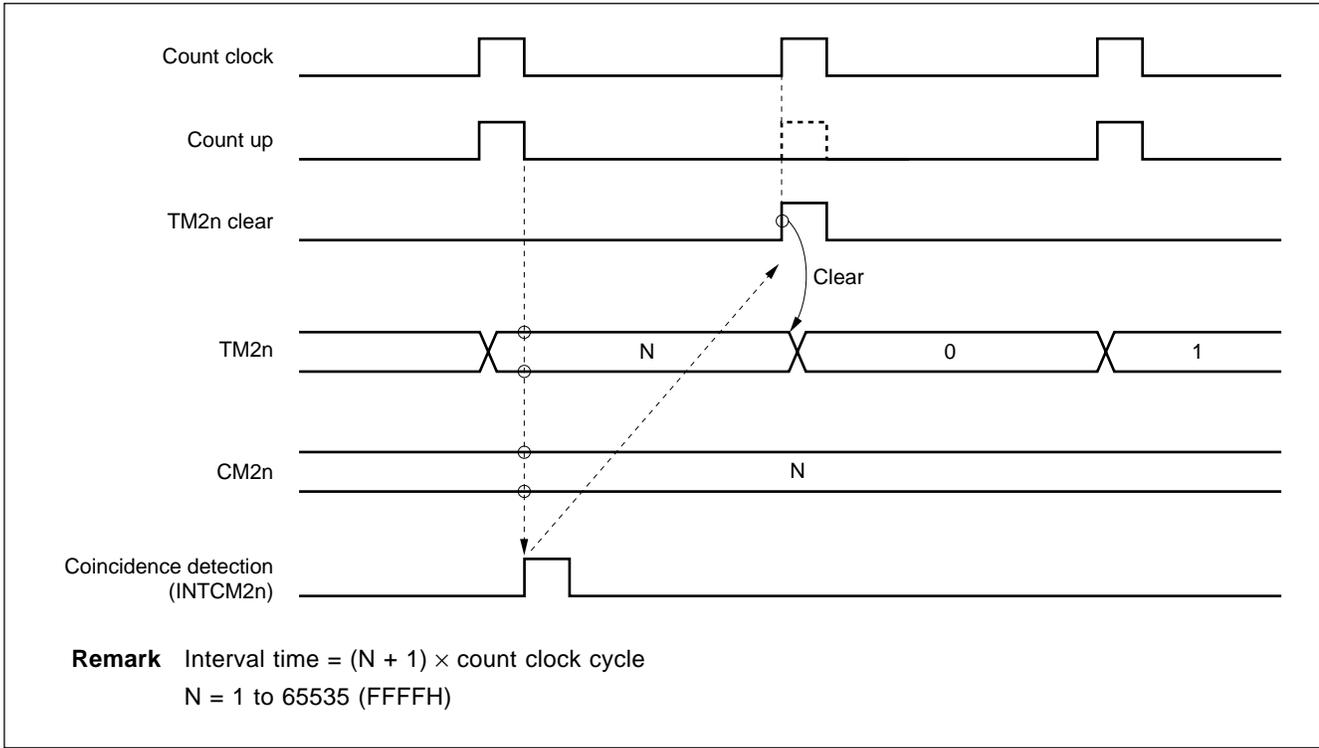
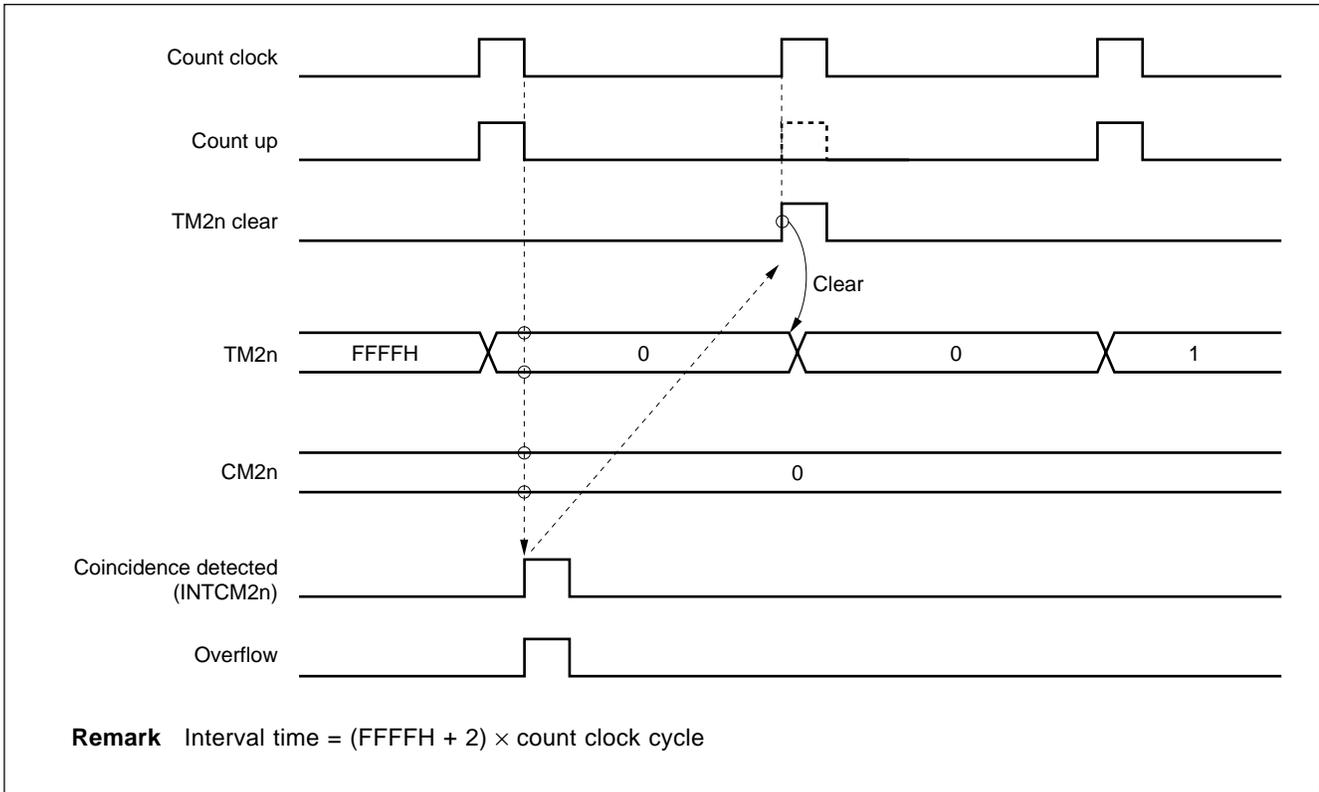


Figure 7-18. When CM2n is Set to 0

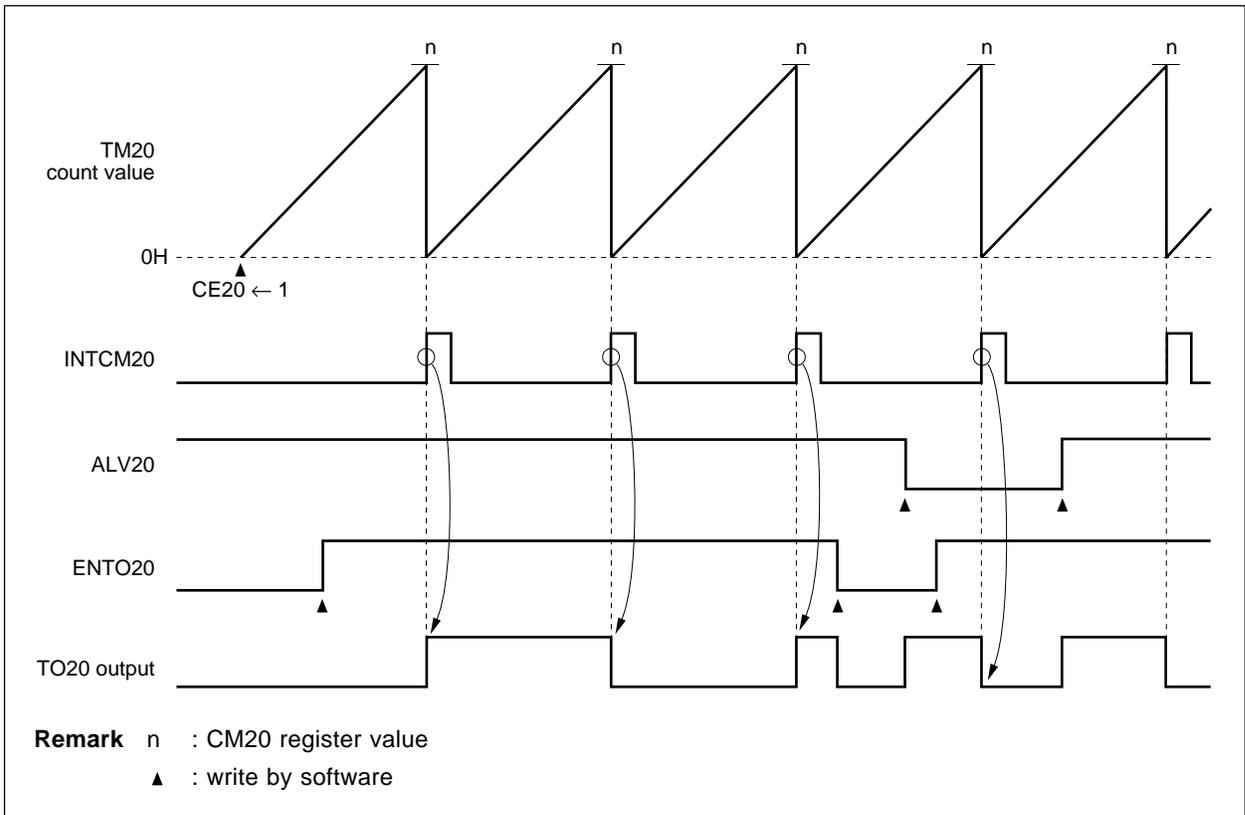


### 7.6.6 Toggle output

Toggle output is an operation mode to invert output levels each time the value of the compare register (CM20 to CM24) coincides with that of CM2n (n = 0 to 4). The relations between timers to be compared and compare register to timer outputs are shown below.

- TM20, CM20 → TO20
- TM21, CM21 → TO21
- TM22, CM22 → TO22
- TM23, CM23 → TO23
- TM24, CM24 → TO24

Figure 7-19. Example of Toggle Output Operation



## 7.7 Timer 3 Operation

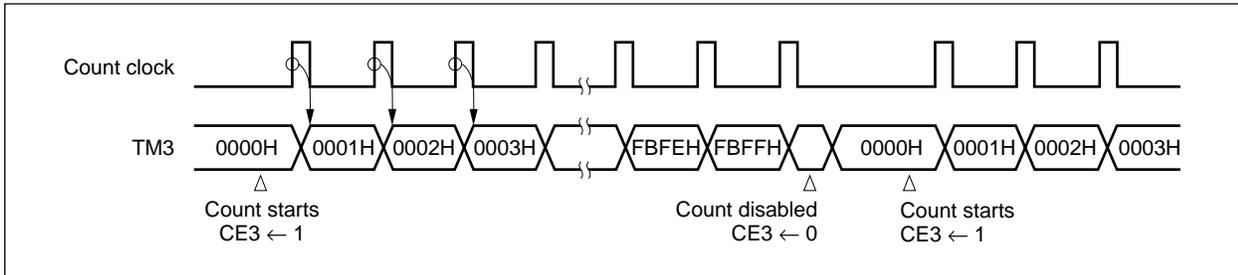
### 7.7.1 Count operation

Timer 3 functions as a 16-bit interval timer. The operation is specified by the timer control registers 3 (TMC3).

The operation of timer 3 counts the internal count clocks ( $\phi/2$  to  $\phi/256$ ) specified by the PRM30 to PRM33 bits of the TMC3 register.

If the count value of TM3 coincides with the value of CC3, the value TM3 is cleared while simultaneously a coincidence interrupt (INTCC3) is generated.

Figure 7-20. Basic Operation of Timer 3



### 7.7.2 Count clock selection

An internal count clock frequency is selected by the PRM30 to PRM33 bits of the TMC3 register, from  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ , and  $\phi/256$ .

**Caution** Do not change the count clock frequency while the timer operates.

PRM33	PRM32	PRM31	PRM30	Internal Count Clock Frequency
0	0	0	1	$\phi/2$
0	0	1	0	$\phi/4$
0	0	1	1	$\phi/8$
0	1	0	0	$\phi/16$
0	1	0	1	$\phi/32$
0	1	1	0	$\phi/64$
0	1	1	1	$\phi/128$
1	0	0	0	$\phi/256$

### 7.7.3 Overflow

If TM3 overflows as a result of counting the internal count clock, a flag is set to the OVF3 bit of the TOVS register.

### 7.7.4 Clearing/starting timer

There are three methods of clearing/starting timer 3: by coincidence of compare register, by capture trigger of CC3, and by software.

#### (1) Clearing/starting by compare coincidence of CC3

If CC3 is specified as a compare register by the CMS 3 bit, TM3 clears its value at the next count clock and starts count operation when the set value of the compare register (CC3) and the value of TM3 coincide. At the same time, an interrupt request (INTCC30) is generated.

The interval timer set to a compare register can be calculated by the following expression:

$$(\text{Set value} + 1) \times \text{Count cycle}$$

For the details, refer to **7.7.6 Compare operation**.

#### (2) Clearing/starting by capture trigger of CC3

If CC3 is specified as a capture register by the CMS3 bit, when a capture trigger of CC3 is generated, the count value of TM3 is captured in CC3, TM3 is cleared, and count operation is started. At the same time, an interrupt request (INTCC3) is generated.

#### (3) Clearing/starting by software

When the CS3 bit of the TMC3 register is set to 1, the TM1 register clears its value and start counting from 0.

However, this setting of the bit is valid only when the value of the CE3 bit is 1.

### 7.7.5 Capture operation

When the TMC3 register is set as a capture register, the capture/compare register (CC3) performs a capture operation that captures and holds the count value of TM3 and loads it to a capture register in synchronization with an external trigger and in asynchronization with the count clock. The valid edge from the external interrupt request input pin INTP30 is used as the capture trigger. In synchronization with this capture trigger signal the count values of TM3 during counting are captured and loaded to the capture register. The value of the capture register is retained until the next capture trigger is generated.

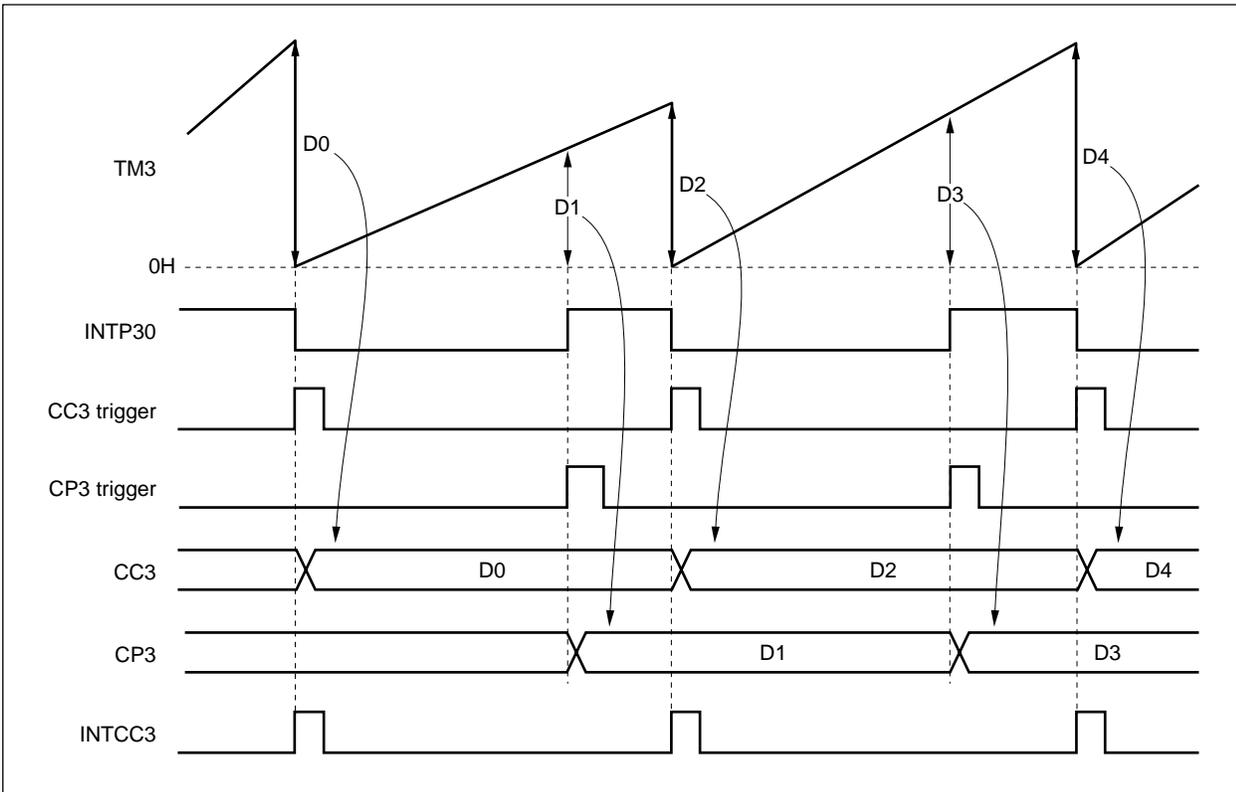
When the capture timing of the capture register and the write operation to the register are in contention, the latter is given priority and the capture operation is ignored.

**Table 7-6. Capture Trigger Signal to 16-Bit Capture Register**

Capture Source	Selection of Valid Edge				Capture Trigger	Interrupt Request
	↓	↑	—	↑↓		
INTP30	↓	↑	—	↑↓	CC3	INTCC3
	↑	↓	↑↓	—	CP3	—

The valid edge of the capture trigger is set by the external interrupt mode register (INTM7). When the CC3 trigger is set to the falling edge and the CP3 trigger is set to the rising edge, input pulse width and input pulse cycle from external can be measured with one interrupt (INTCC3).

Figure 7-21. Example of TM3 Capture Operation (when ES301 = 0, ES300 = 0, CMS3 = 0, CE3 = 1)



### 7.7.6 Compare operation

When the TM3 register is set as a compare register, the capture/compare register (CC3) performs a comparison between the value in a compare register and the count value of TM3.

When the count value of TM3 coincides with the value of the compare register programmed in advance, clear and start of TM3 is performed, and interrupt request signal INTCC3 is generated at the same time.

7.8 Application Examples

(1) Operation as interval timer (timer 0, timer 2, and timer 3)

The following shows that timer 2 used as an interval timer that repeatedly generates an interrupt request at time intervals specified by the count value set in advance to compare register CM20. Figure 7-22 shows the timing. Figure 7-23 illustrates the setting procedure.

Figure 7-22. Example of Timing of Interval Timer Operation (timer 2)

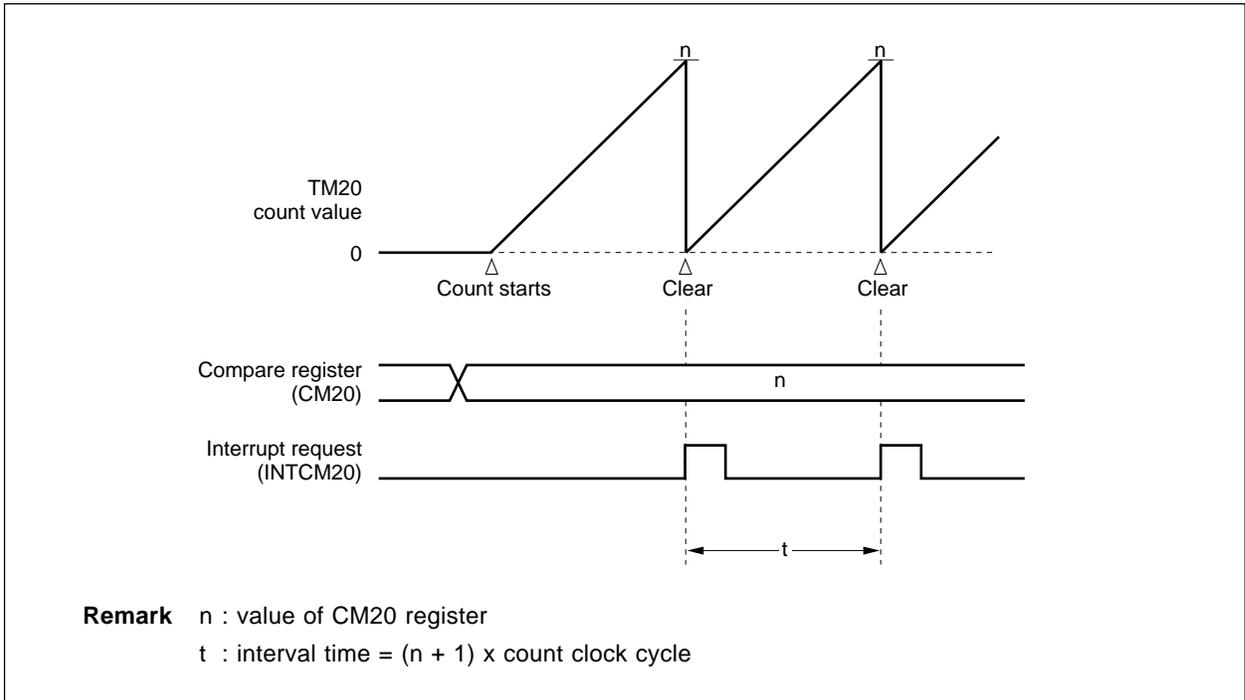
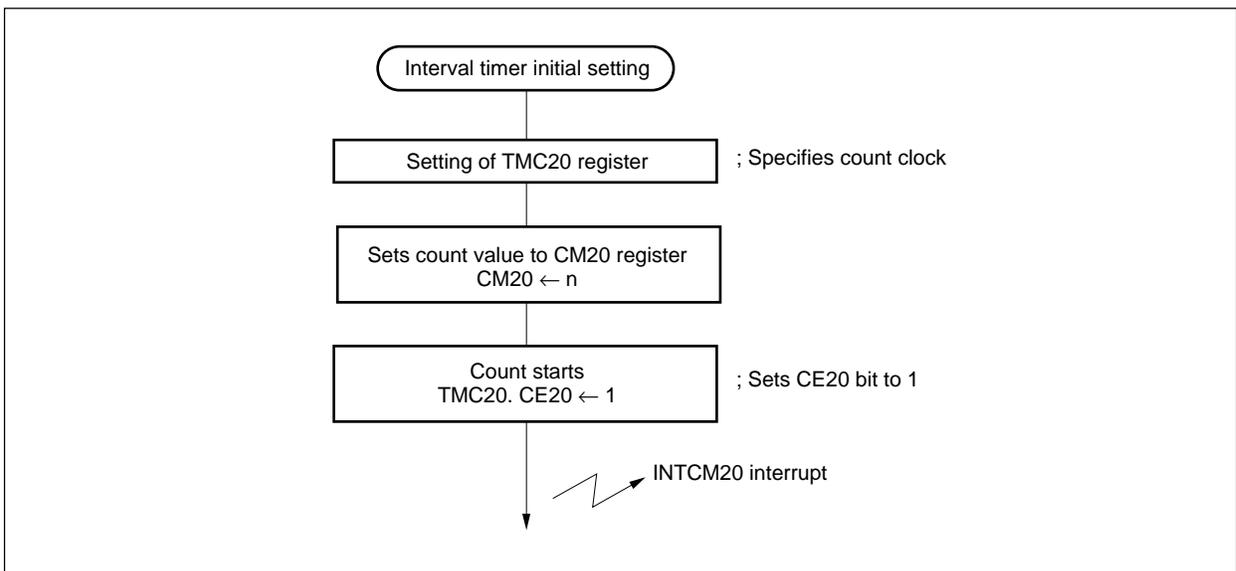


Figure 7-23. Setting Procedure of Interval Timer Operation (timer 2)



**(2) Pulse width measurement (timer 0, timer 1, and timer 3)**

An example of pulse width measurement is shown below.

In this example, the width of the high or low level of an external pulse input to the INTP00 pin is measured. The value of timer 1 (TM0) is captured to a capture/compare register (CC00) in synchronization with the valid edge of the INTP00 pin (both the rising and falling edges) and is pended, as shown in Figure 7-24.

To calculate the pulse width, the difference between the count value of TM0 captured to the CC00 register on detection of the nth valid edge (Dn), and the count value on detection of the (n – 1)th valid edge (Dn – 1) is calculated. This difference is multiplied by the count clock.

**Figure 7-24. Pulse Width Measurement Timing (timer 0)**

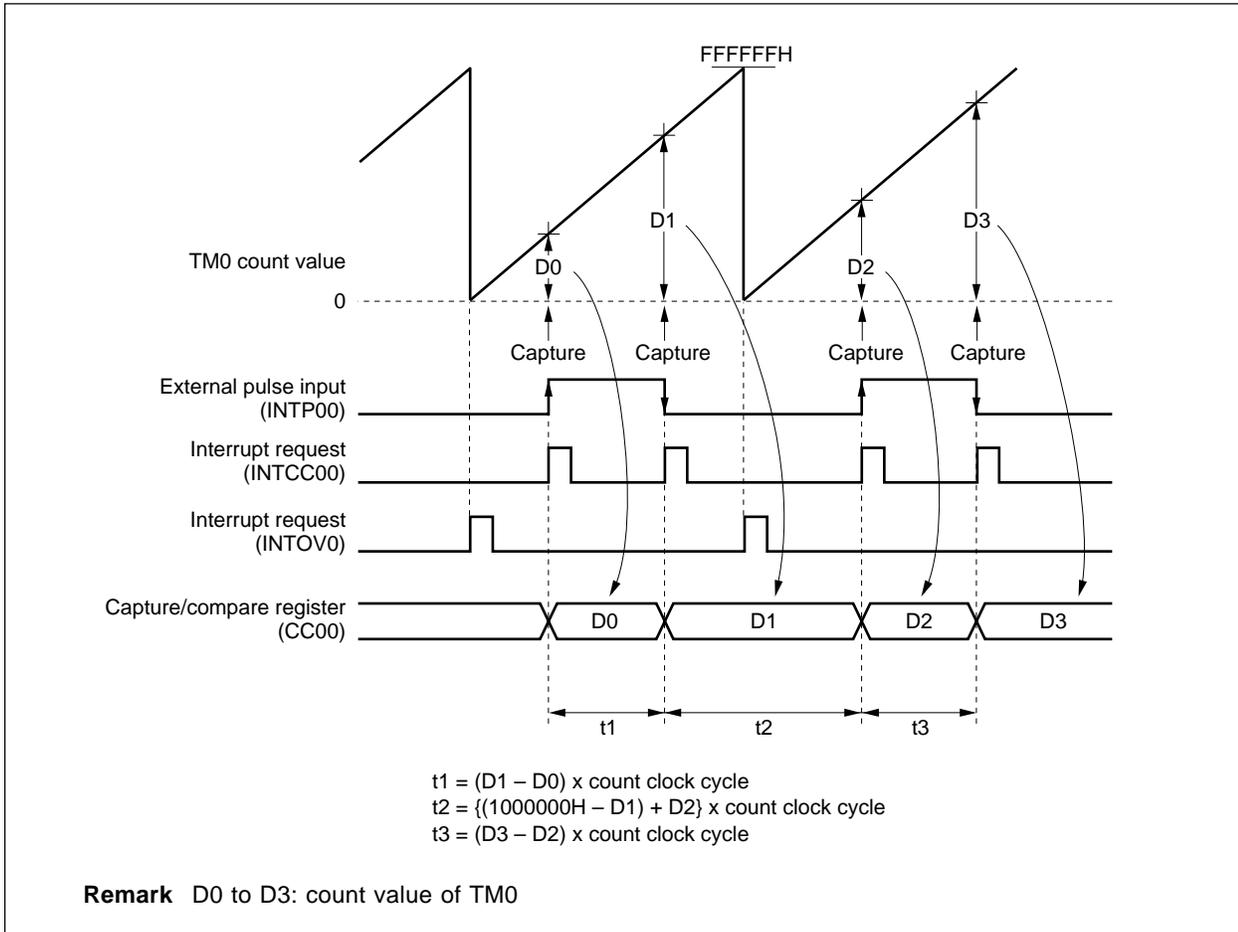


Figure 7-25. Setting Procedure for Pulse Width Measurement (timer 0)

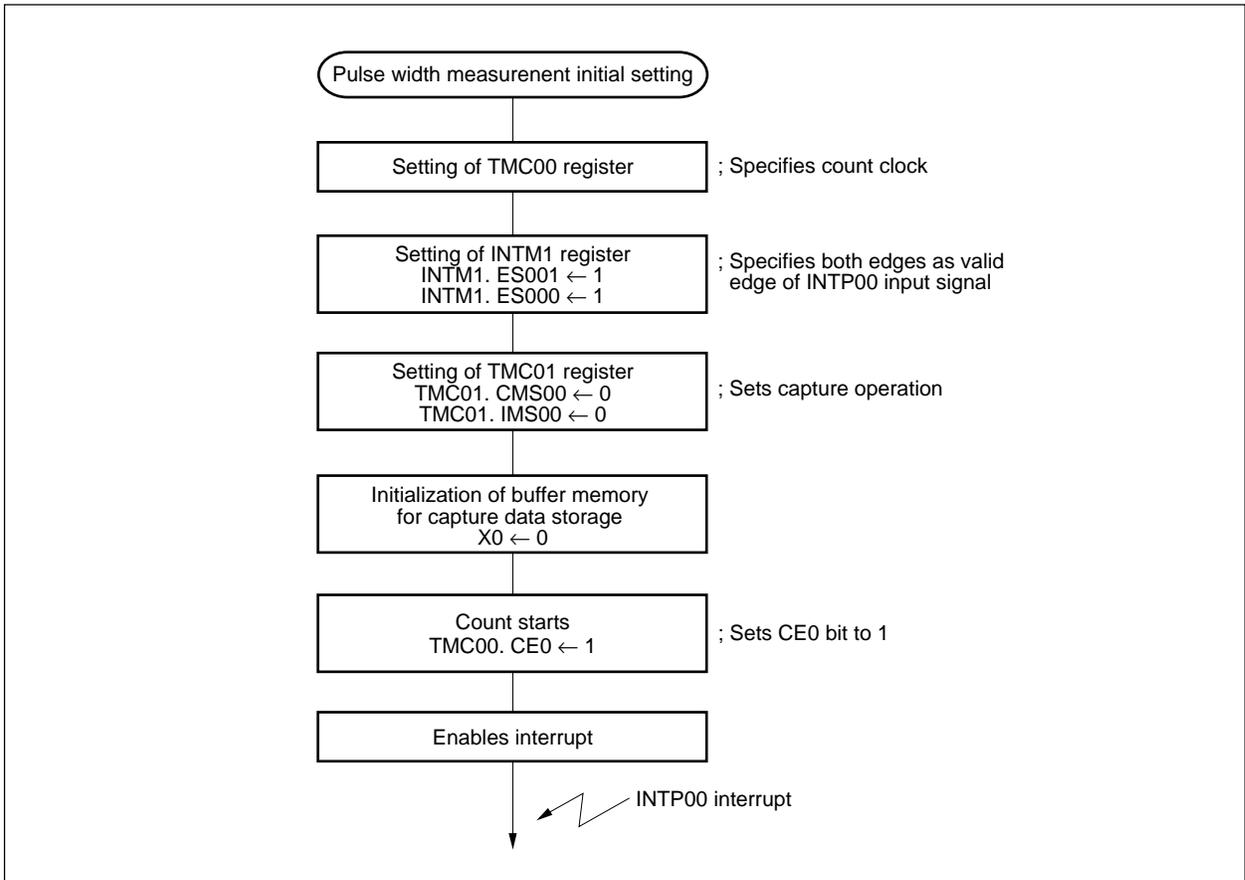
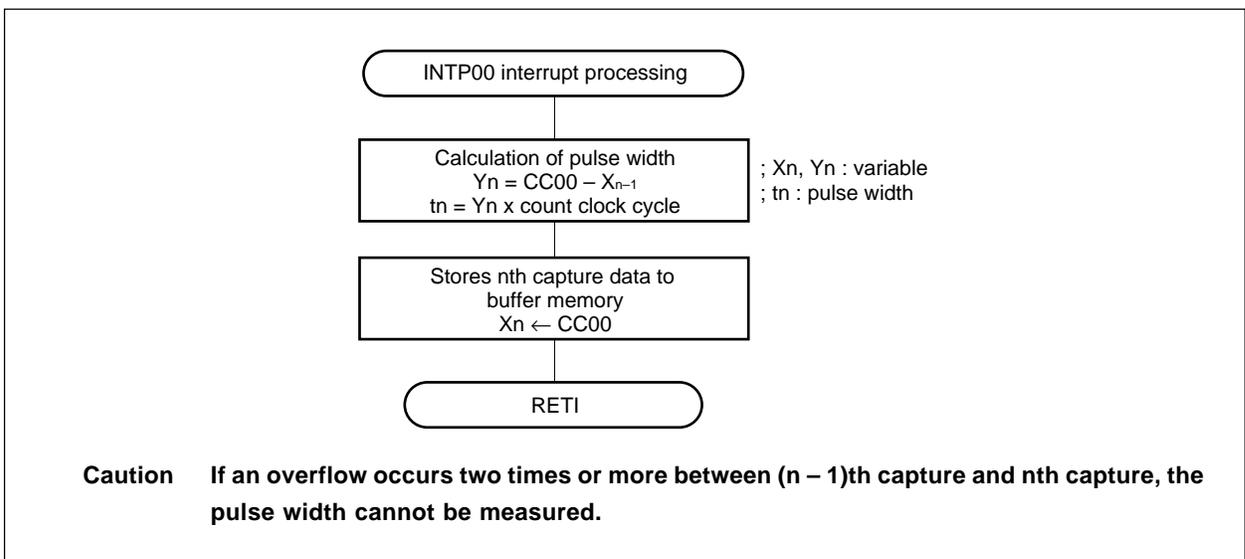


Figure 7-26. Interrupt Request Processing Routine Calculating Pulse Width (timer 0)



**(3) PWM output (timer 0)**

Any square wave can be output to timer output pin (TO<sub>n</sub>) by combining the use of timer 0 and the timer output function and can be used as a PWM output.

Shown below is an example of PWM output using two capture/compare registers, CC00 and CC01. In this case, a PWM signal with an accuracy of 24 bits can be output from the TO00 pin. Figure 7-27 shows the timing. When timer 0 is used as a 24-bit timer, the rising timing of the PWM output is determined by the value set to capture/compare register CC00, and the falling timing is determined by the value set to capture/compare register CC01.

The interval frequency of timer output can be changed freely by using compare coincidence of CC03 and by clearing and starting TM0.

**Figure 7-27. Example of PWM Output Timing**

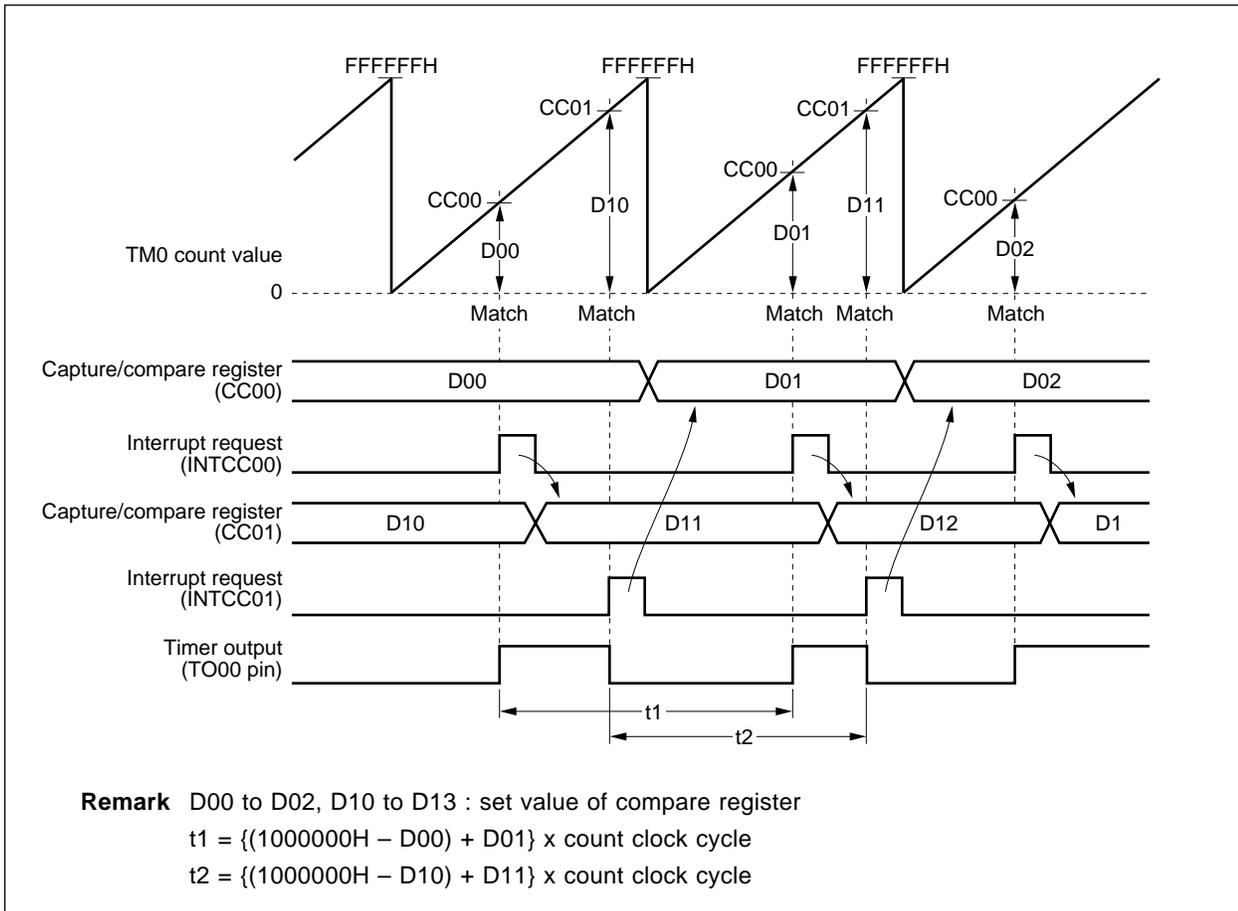


Figure 7-28. Example of PWM Output Programming Procedure

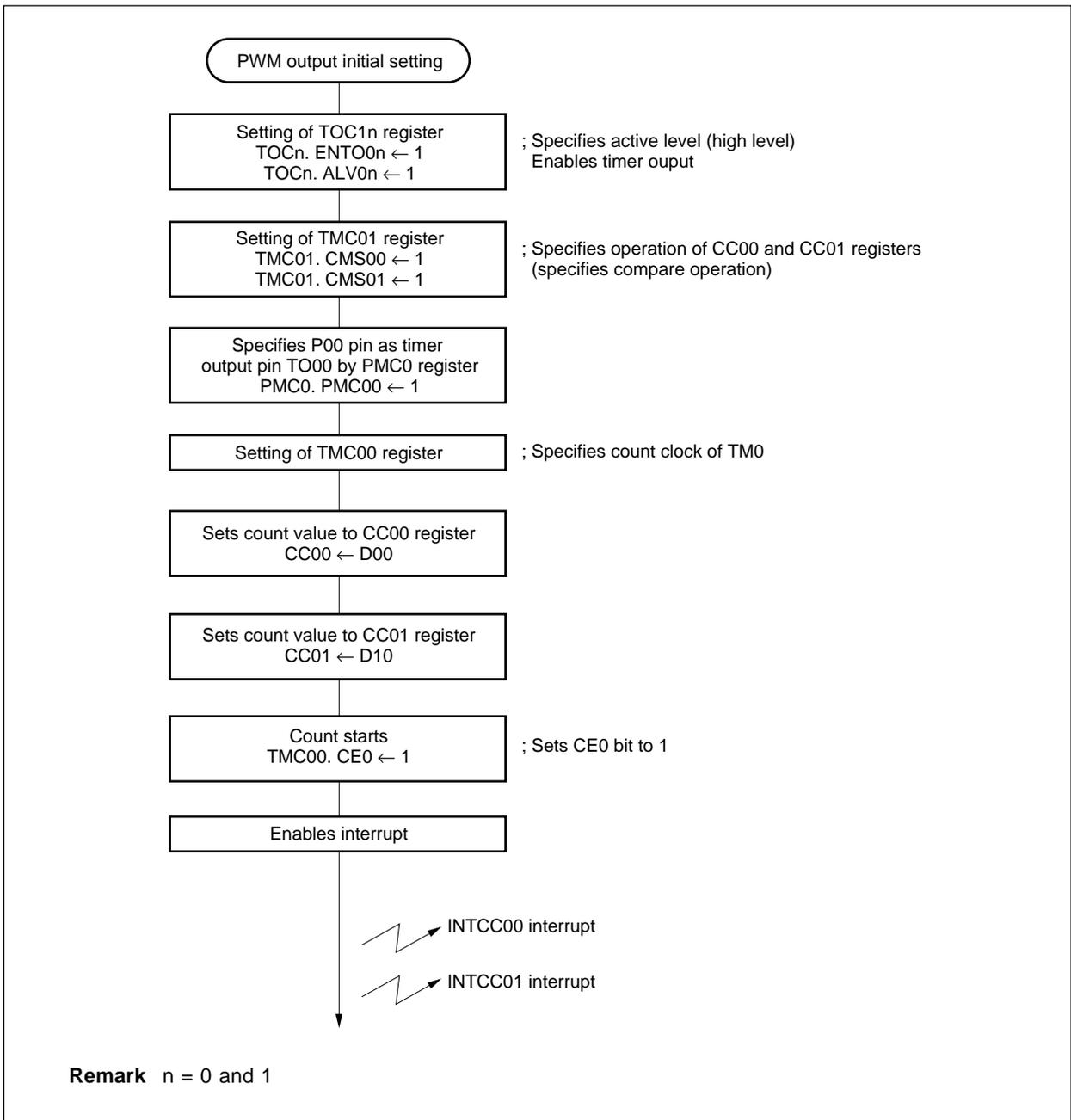
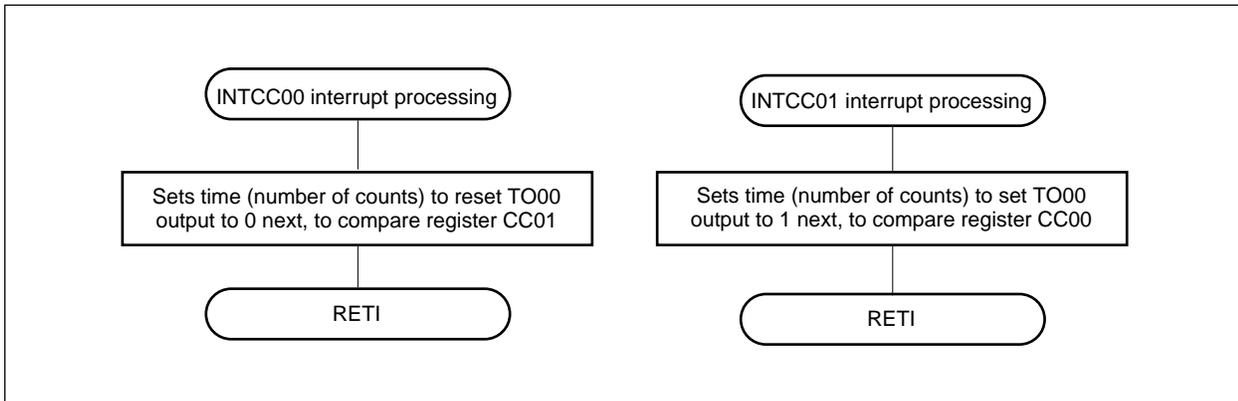


Figure 7-29. Example of Interrupt Request Processing Routine, Modifying Compare Value



**(4) Frequency measurement (timer 0, timer 1, and timer 3)**

Timer 0, timer 1, and timer 3 can be used to measure the cycle or frequency of an external pulse input to the INTP pin.

Shown below is an example where the frequency of the external pulse input to the INTP00 pin is measured with an accuracy of 24 bits, by combining the use of timer 0 and the capture/compare register CC00.

The valid edge of the INTP00 input signal is specified by the INTM1 register to be the rising edge.

To calculate the frequency, the difference between the count value of TM0 captured to the CC00 register at the  $n$ th rising edge ( $D_n$ ), and the count value captured at the  $(n - 1)$ th rising edge ( $D_{n - 1}$ ), is calculated, and the value multiplied by the count clock frequency.

The frequency measurement exceeding the maximum count value of TM0 is performed by counting the number of overflow with the INTOV0 overflow interrupt request.

**Figure 7-30. Example of Frequency Measurement Timing**

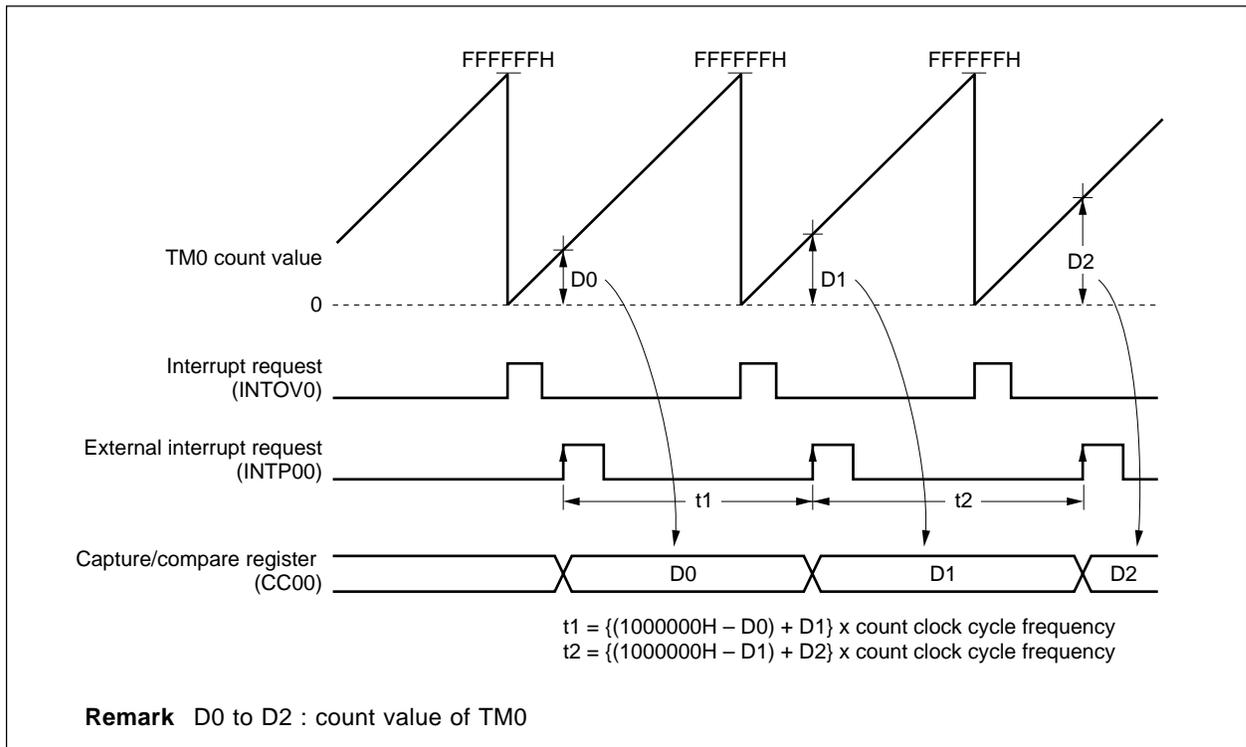


Figure 7-31. Example of Set-up Procedure for Frequency Measurement

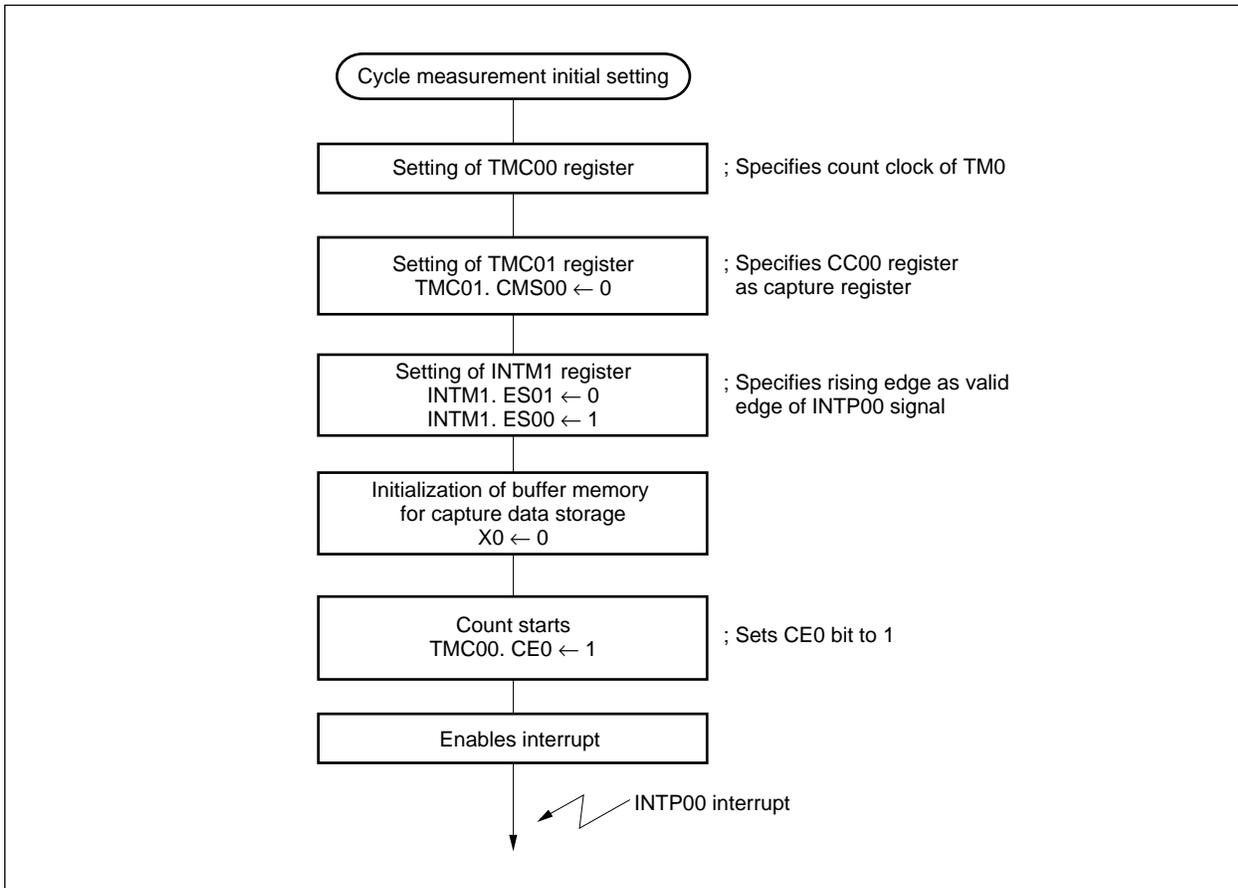
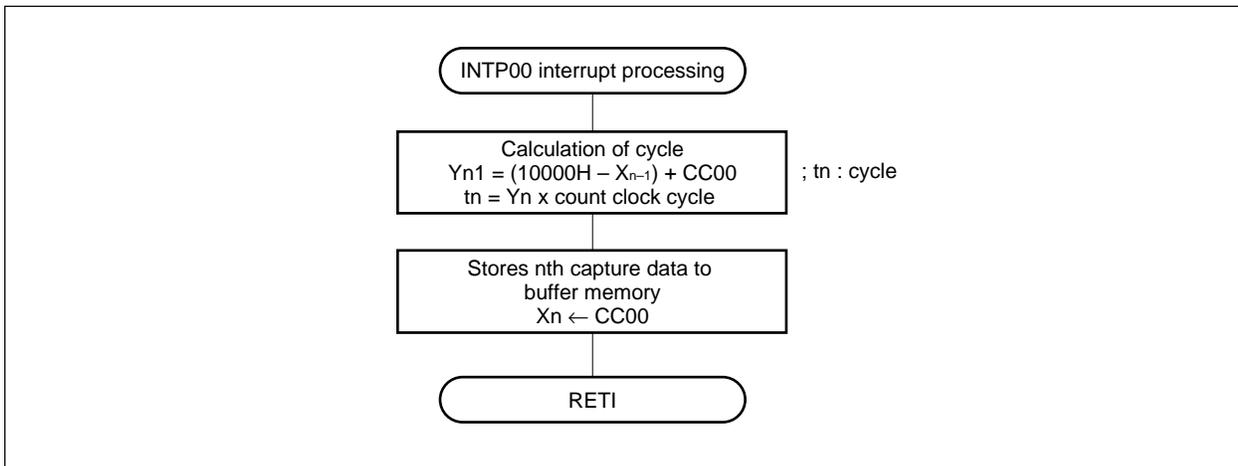


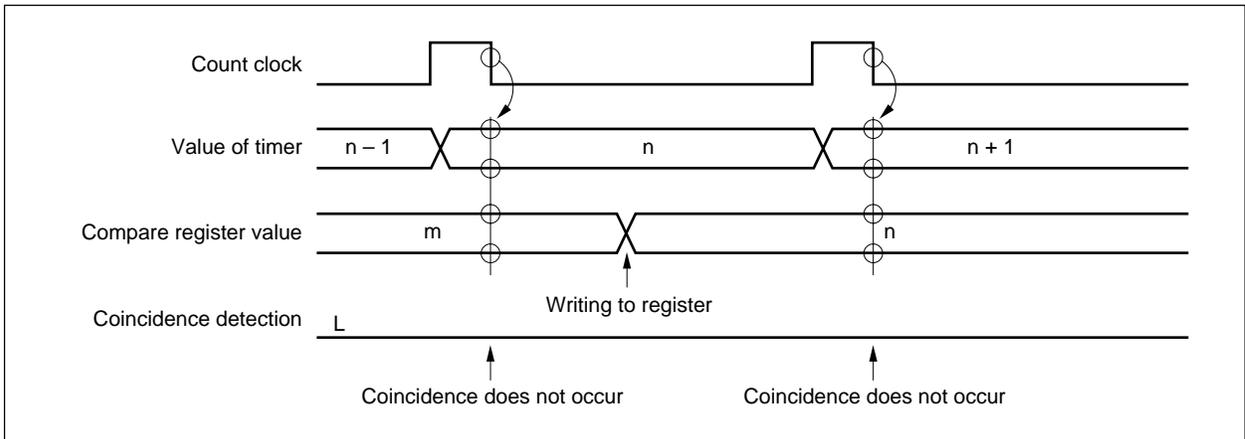
Figure 7-32. Example of Interrupt Request Processing Routine Calculating Cycle



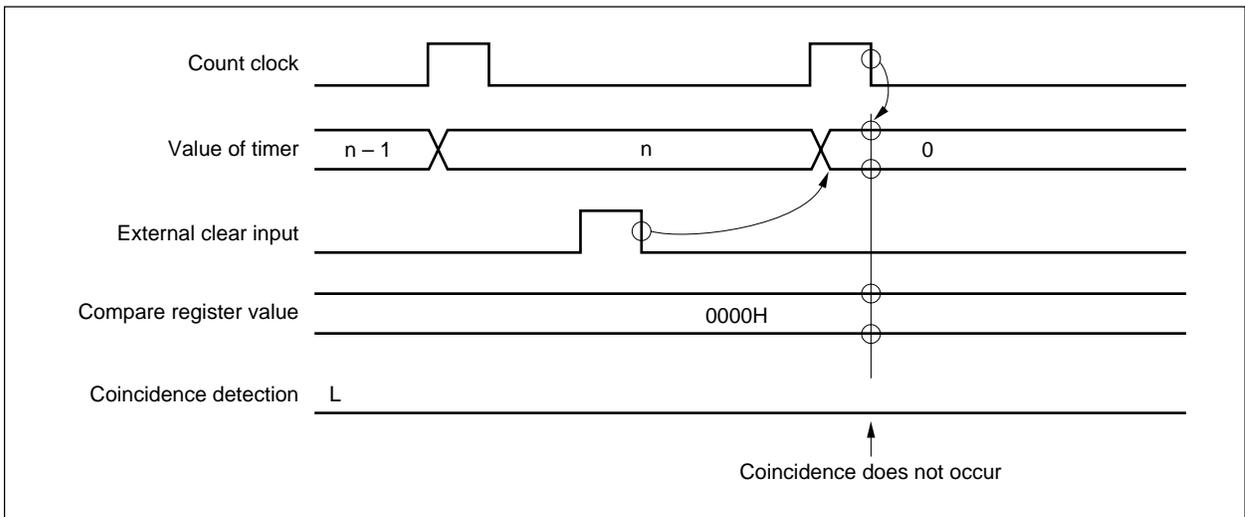
**7.9 Note**

Coincidence is detected by the compare register immediately after the timer value matches the compare register value, and does not take place in the following cases:

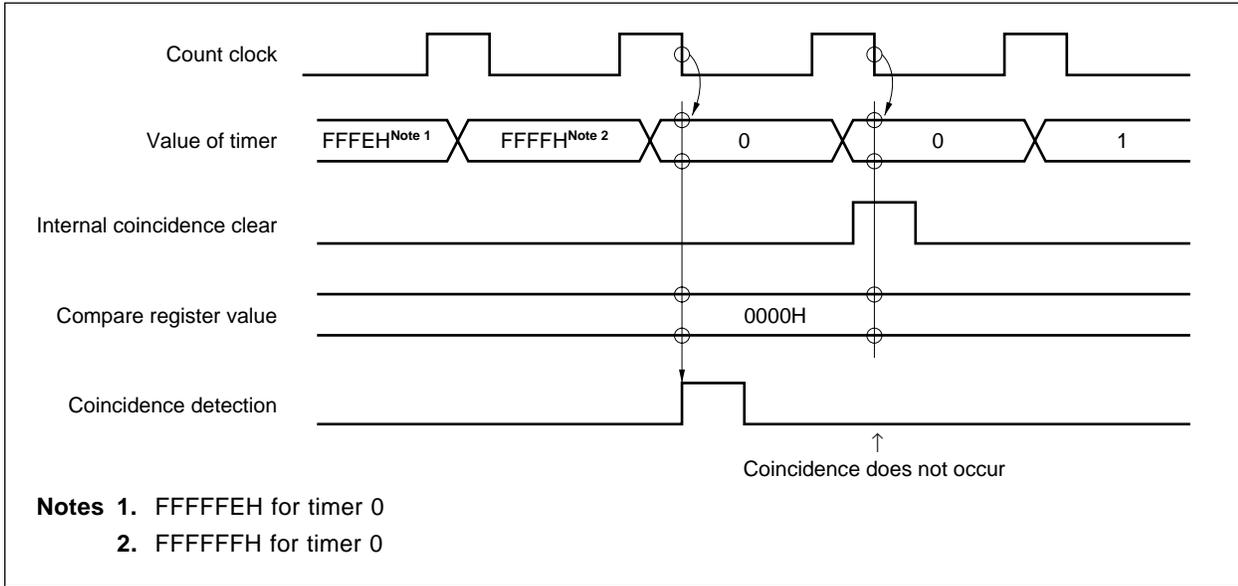
**(1) When compare register is rewritten (timer 0 to timer 3)**



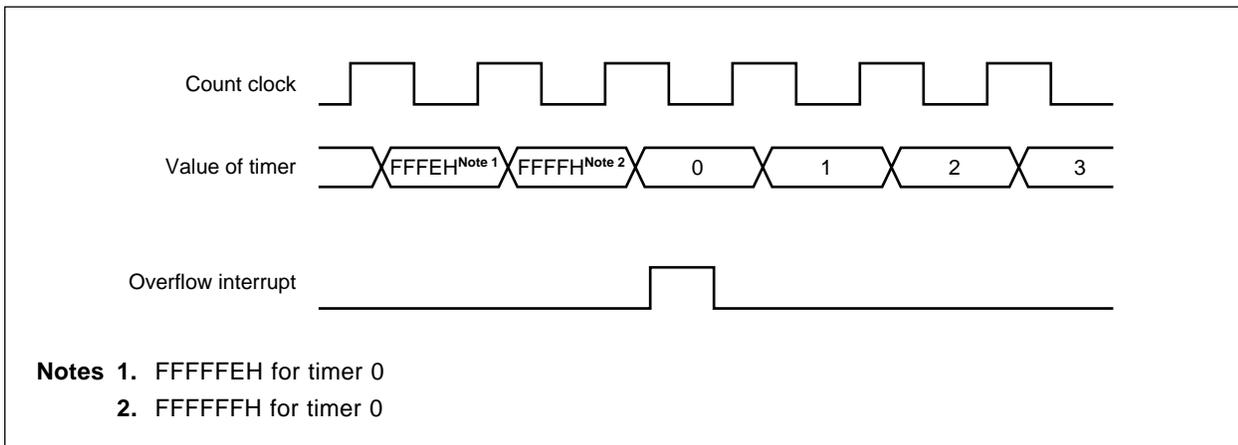
**(2) When timer is cleared by external input (timer 0)**



(3) When timer is cleared (timer 0, timer 2, and timer 3)



**Remark** When timer 0 or timer 1 is operated as a free running timer, the timer value is cleared to 0 when the timer overflows.



## CHAPTER 8 SERIAL INTERFACE FUNCTION

### 8.1 Features

The V854 is provided with three types of serial interfaces which operate as 6-channel transmission/reception channels. Four channels can be used simultaneously.

There are the following three types of interfaces.

- (1) Asynchronous serial interface (UART) : 1 channel
- (2) Clocked serial interface (CSI) : 4 channels
- (3) I<sup>2</sup>C bus interface (I<sup>2</sup>C) : 1 channel ( $\mu$ PD703008Y and 70F3008Y only)

The UART transmits/receives 1-byte serial data following a start bit and can perform full-duplex communication.

The CSI uses three signal lines to high-speed synchronous data transfers data (3-wire serial I/O): serial clock ( $\overline{\text{SCKn}}$ ), serial input (SIn), and serial output (SOn) lines.

The I<sup>2</sup>C uses two signal lines, which are serial clock (SCL) and serial data bus (SDA), to transfer data ( $\mu$ PD703008Y and 70F3008Y only).

**Caution** UART and CSI0, and CSI1 and I<sup>2</sup>C share the same pin respectively. Either one of these is selected according to ASIM0 and ASIM1.

## 8.2 Asynchronous Serial Interface (UART)

### 8.2.1 Features

- Transfer rate: 150 bps to 153600 bps (Baud rate generator used, @  $\phi$  = 33-MHz operation)  
110 bps to 614400 bps (Baud rate generator used, @  $\phi$  = 19.660-MHz operation)
- ★ 110 bps to 38400 bps (Baud rate generator used, @  $\phi$  = 16-MHz operation)  
Max. 1031 kbps ( $\phi/2$  use, @  $\phi$  = 33-MHz operation)
- Full-duplex communication: internal receive buffer (RXB)
- Two-pin configuration: TXD: transmit data output pin  
RXD: receive data input pin
- Receive error detection function
  - Parity error
  - Framing error
  - Overrun error
- Interrupt sources: 3
  - Receive error interrupt (INTSER)
  - Reception completion interrupt (INTSR)
  - Transmission completion interrupt (INTST)
- Character length of transmit/receive data is specified by ASIMn registers.
- Character length: 7, 8 bits  
9 bits (when extended)
- Parity function: odd, even, 0, none
- Transmit stop bit: 1, 2 bits
- Dedicated internal baud rate generator

**Remark**  $\phi$  : Internal system clock

### 8.2.2 Configuration of asynchronous serial interface

The asynchronous serial interface is controlled by the asynchronous serial interface mode register (ASIMn) and the asynchronous serial interface status register (ASIS). The receive data is stored in the receive buffer (RXB), and the transmit data is written to the transmit shift register (TXS).

Figure 8-1 shows the configuration of the asynchronous serial interface.

#### (1) Asynchronous serial interface mode registers (ASIM0, ASIM1)

ASIMn are 8-bit registers that specify the operation of the asynchronous serial interface.

#### (2) Asynchronous serial interface status registers (ASIS)

ASIS are registers containing flags that indicate receive errors, if any, and a transmit status flag. Each receive error flag is set to 1 when a receive error occurs, and is reset to 0 when data is read from the receive buffer (RXB), or when new data is received (if the next data contains an error, the corresponding error flag is set). The transmit status flag is set to 1 when transmission is started, and reset to 0 when transmission ends.

#### (3) Reception control parity check

The reception operation is controlled according to the contents programmed in the ASIMn registers. During the receive operation, errors such as parity error are also checked. If an error is found, the appropriate value is set to the ASIS registers.

#### (4) Receive shift register

This shift register converts the serial data received on the RXD pin into parallel data. When it receives 1 byte of data, it transfers the receive data to the receive buffer.

This register cannot be directly operated.

#### (5) Receive buffers (RXB, RXBL)

RXB are 9-bit buffer registers that hold receive data. If data of 7 or 8 bits/character is received, 0 is stored to the most significant bit position of these registers.

If these registers are accessed in 16-bit units, RXB are specified. To access in lower 8-bit units, RXBL are specified.

While reception is enabled, the receive data is transferred from the receive shift register to the receive buffer in synchronization with shift-in processing of 1 frame.

When the data is transferred to the receive buffer, a reception completion interrupt request (INTSR) occurs.

#### (6) Transmit shift registers (TXS, TXSL)

TXS are 9-bit shift registers used for transmit operation. When data is written to these registers, the transmission operation is started.

A transmission complete interrupt request (INTST) is generated after each complete data frame is transmitted. When these registers are accessed in 16-bit units, TXS are specified. To access in lower 8-bit units, TXSL are specified.

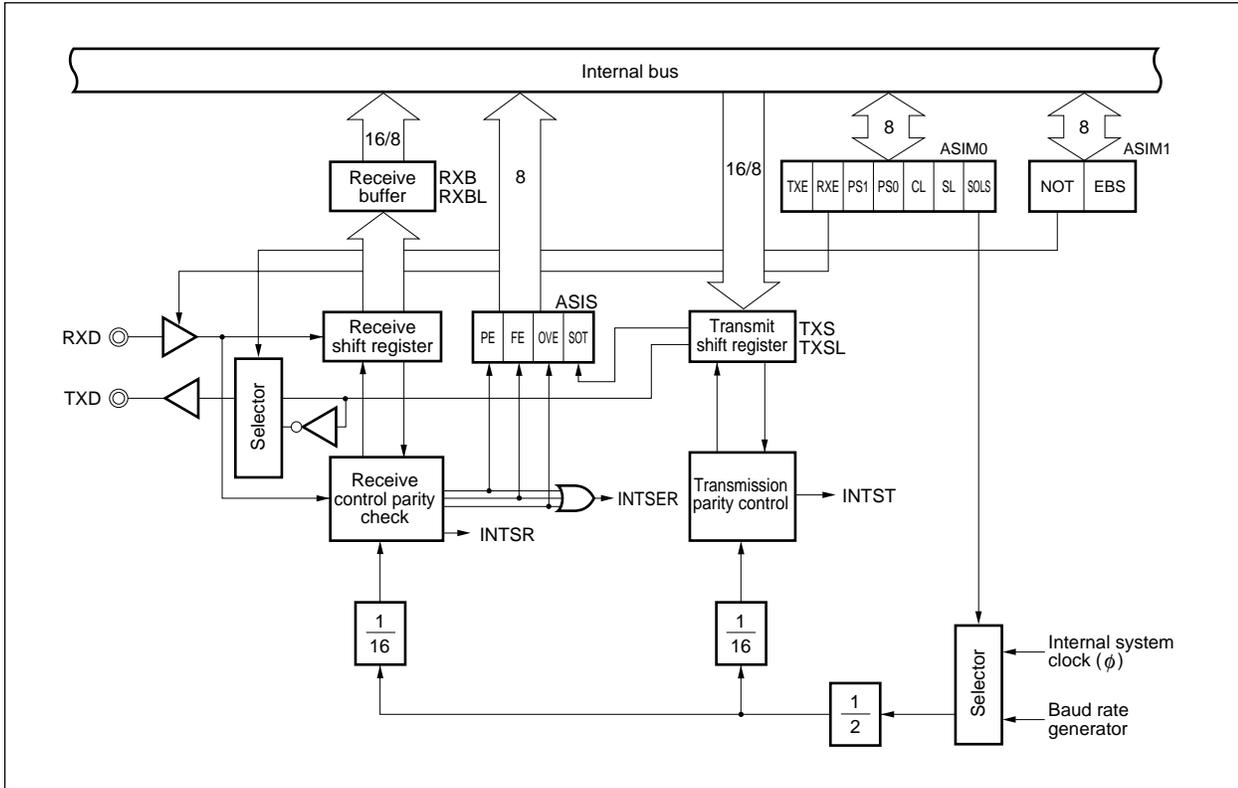
#### (7) Transmission parity control

A start bit, parity bit, and stop bit are appended to the data written to the TXS registers, according to the contents programmed in the ASIMn registers, to control the transmission operation.

(8) Selector

Selects the source of the serial clock.

Figure 8-1. Block Diagram of Asynchronous Serial Interface



8.2.3 Control registers

(1) Asynchronous serial interface mode registers 0, 1 (ASIM0, ASIM1)

These registers specify the transfer mode of the UART.

They can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
ASIM0	TXE	RXE	PS1	PS0	CL	SL	0	SCLS	Address	After reset
									FFFFF0C0H	80H

Bit Position	Bit Name	Function
7	TXE	Transmit Enable Enable/disable transmission. 0 : Disable transmission 1 : Enable transmission
6	RXE	Receive Enable Enable/disable reception. 0 : Disable reception 1 : Enable reception  When reception is disabled, the receive shift register does not detect the start bit. Data is not shifted into the receive shift register and neither is any transfer to the receive buffer performed. Therefore, the previous contents of the receive buffer are retained. When reception is enabled, the data is shifted into the receive shift register and transferred to the receive buffer when one complete frame has been received. A reception completion interrupt (INTSRn) is generated in synchronization with the transfer to the receive buffer.  If this bit is set to receive disabled status during receive operation, the data being received is relinquished and the data before the receive operation is started is read out.

Bit Position	Bit Name	Function															
5, 4	PS1, PS0	<p>Parity Select Specifies parity bit.</p> <table border="1"> <thead> <tr> <th>PS1</th> <th>PS0</th> <th>Operation</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>No parity. Extended bit operation</td> </tr> <tr> <td>0</td> <td>1</td> <td>0 parity Transmission side → Transmits with parity bit 0 Reception side → Does not generate parity error on reception</td> </tr> <tr> <td>1</td> <td>0</td> <td>Odd parity</td> </tr> <tr> <td>1</td> <td>1</td> <td>Even parity</td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>• <b>Even parity</b> Parity bit is set to “1” when number of bits equal to one in received data is odd. If number of bits that are one is even, parity bit is cleared to 0. In this way, number of bits that are “1” in transmit data and parity bit is controlled to become even. During reception, number of bits that are “1” in receive data and parity bit are counted. If it is odd, parity error occurs.</li> <li>• <b>Odd parity</b> In contrast to even parity, number of bits included in transmit data and parity bit that are “1” is controlled to become odd. Since no parity bit check is performed during reception, no parity error occurs.</li> <li>• <b>0 parity</b> Parity bit is cleared to “0” during transmission, regardless of transmit data. Since no parity bit check is performed during reception, no parity error occurs.</li> <li>• <b>No parity</b> No parity bit is appended to the transmit data. Reception is performed on assumption that there is no parity bit. Because no parity bit is used, parity error does not occur. Extended bit operation can be specified by EBS bit of ASIM1 register.</li> </ul>	PS1	PS0	Operation	0	0	No parity. Extended bit operation	0	1	0 parity Transmission side → Transmits with parity bit 0 Reception side → Does not generate parity error on reception	1	0	Odd parity	1	1	Even parity
PS1	PS0	Operation															
0	0	No parity. Extended bit operation															
0	1	0 parity Transmission side → Transmits with parity bit 0 Reception side → Does not generate parity error on reception															
1	0	Odd parity															
1	1	Even parity															
3	CL	<p>Character Length Specifies character length of one frame. 0 : 7 bits 1 : 8 bits</p>															
2	SL	<p>Stop Bit Length Specifies stop bit length. 0 : 1 bit 1 : 2 bits</p>															

Bit Position	Bit Name	Function																		
0	SCLS	<p>Serial Clock Source Specifies serial clock. 0 : Specified by BRGC0 and BPRM0 1 : <math>\phi/2</math></p> <ul style="list-style-type: none"> <li>When SCLS = 1 <math>\phi/2</math> is selected as serial clock source. In asynchronous mode, baud rate is expressed as follows because sampling rate of x16 is used: <math display="block">\text{Baud rate} = \frac{\phi/2}{16} \text{ bps}</math> Value of baud rate when typical clock is used based on above expression is as follows:</li> </ul> <table border="1"> <thead> <tr> <th><math>\phi</math></th> <th>33 MHz</th> <th>25 MHz</th> <th>20 MHz</th> <th>16 MHz</th> <th>12.5 MHz</th> <th>10 MHz</th> <th>8 MHz</th> <th>5 MHz</th> </tr> </thead> <tbody> <tr> <td>Baud rate</td> <td>1031 K</td> <td>781 K</td> <td>625 K</td> <td>500 K</td> <td>390 K</td> <td>312 K</td> <td>250 K</td> <td>156 K</td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>When SCLS = 0 Baud rate generator output is selected as serial clock source. For details of baud rate generator, refer to <b>8.5 Baud Rate Generator 0 to 3 (BRG0 to BRG3)</b>.</li> </ul>	$\phi$	33 MHz	25 MHz	20 MHz	16 MHz	12.5 MHz	10 MHz	8 MHz	5 MHz	Baud rate	1031 K	781 K	625 K	500 K	390 K	312 K	250 K	156 K
$\phi$	33 MHz	25 MHz	20 MHz	16 MHz	12.5 MHz	10 MHz	8 MHz	5 MHz												
Baud rate	1031 K	781 K	625 K	500 K	390 K	312 K	250 K	156 K												

**Caution** The operation of UART is not guaranteed if these registers are changed while UART is transmitting/receiving data.

**Remark**  $\phi$ : Internal system clock

	7	6	5	4	3	2	1	0	Address	After reset
ASIM1	0	0	0	0	0	0	NOT	EBS	FFFFFF0C2H	00H

Bit Position	Bit Name	Function
1	NOT	<p>Not Inverts the output level from TXD pin. 0 : Does not invert the output level 1 : Inverts output level Use this function to connect an external circuit having inverted level to TXD pin.</p>
0	EBS	<p>Extended Bit Select Specifies extended bit operation of transmit/receive data when no parity is specified (PS0 = 0, PS1 = 0). 0 : Disables extended bit operation 1 : Enables extended bit operation When extended bit operation is enabled, 1 data bit is appended as most significant bit to 8-bit transmit/receive data, and therefore 9-bit data is communicated. Extended bit operation is valid only when no parity is specified by ASIM0 register. If zero, even, or odd parity is specified, specification by EBS bit is invalid, and extended bit is not appended.</p>

**(2) Asynchronous serial interface status register (ASIS)**

This register contains three error flags that indicate the receive error status for each character received and the status of the transmit shift register.

The error flags always indicate the status of an error that has occurred most recently. If two or more errors occur before the current received data, only the status of the error that has occurred last is retained.

If a receive error occurs, read the data of the receive buffer RXB/RXBL after reading the ASIS register, and then clear the error flag.

This register can only be read in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
ASIS	SOT	0	0	0	0	PE	FE	OVE	Address FFFFF0C4H	After reset 00H

Bit Position	Bit Name	Function
7	SOT	Status of Transmission Status flag that indicates transmission operation status. Set (1) : Beginning of transmission of a data frame (writing to TXS register) Clear (0) : End of transmission of a data frame (occurrence of INTST) When serial data transfer begins, this flag will indicate if the transmit shift register is ready to be written or not.
2	PE	Parity Error Status flag that indicates parity error. Set (1) : Transmit parity and receive parity do not match Clear (0) : No error; this flag is automatically cleared to 0 when the data is read from the receive buffer.
1	FE	Framing Error Status flag that indicates framing error. Set (1) : Stop bit is not detected Clear (0) : No error; this flag is automatically cleared to 0 when the data is read from the receive buffer.
0	OVE	Overrun Error Status flag that indicates overrun error. Set (1) : Overrun error; Contents of the receive shift register are transferred to the receive buffer before the previous data has been read by the CPU. This will cause an overwriting of data and the previous information will be lost. Clear (0) : No error; this flag is automatically cleared to 0 when the data is read from the receive buffer. Because contents of receive shift register are transferred to receive buffer each time one frame of data has been received, if overrun error occurs, next receive data is written over contents of receive buffer, and previous receive data is discarded.

**(3) Receive buffers (RXB, RXBL)**

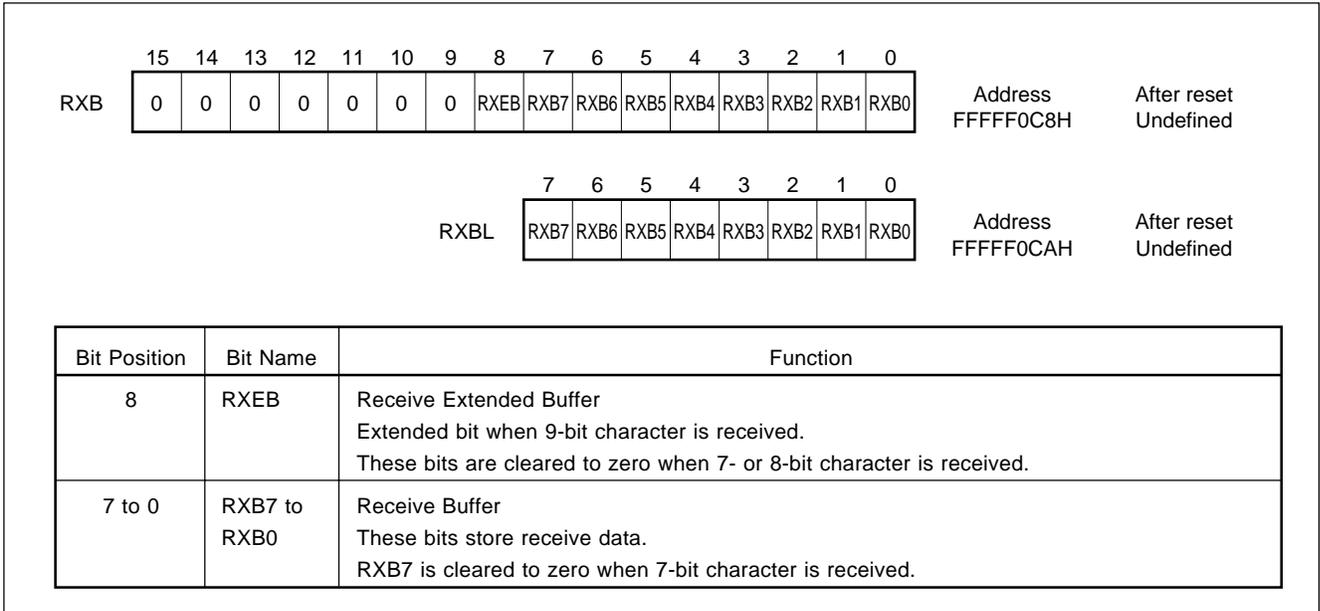
RXB are 9-bit buffer registers that hold the receive data. When a 7- or 8-bit character is received, the higher bit of these registers are 0.

When reading 16-bit data from the receive buffer, RXB is specified. When data is read from the lower 8bits, RXBL is specified.

During the state where reception is enabled the receive data is transferred from the receive shift register to the receive buffer synchronizing one complete frame of data has been shifted in.

When the receive data is transferred to the receive buffer, a reception completion interrupt request (INTSR) occurs.

During the state where reception is disabled the data is not transferred into the reception buffer even when one complete frame of data has been shifted in and the data of the reception buffer is retained. In addition, the reception completion interrupt request is not generated. RxB is only possible for reading in 16-bit units and RXBL in 8-/1-bit units.

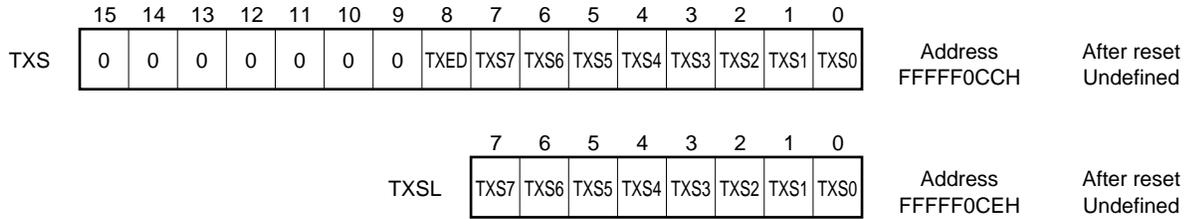


**(4) Transmit shift registers (TXS, TXSL)**

TXS are 9-bit shift registers for data transmission. The transmit operation is started when data is written to these registers during transmission enable status.

If data is written to the transmit shift register in the transmission disabled status, the values written are ignored. Transmission complete interrupt request (INTST) is generated after each complete data frame including TXS is transmitted.

In the case of access in 16-bit units, TXS is specified. To access the lower 8bits, TXSL is specified. TXS can only be written to TXS in 16-bit units, and TXSL in 8-bit units.



Bit Position	Bit Name	Function
8	TXED	Transmit Extended Data Extended bit on transmission of 9-bit/character.
7 to 0	TXS7 to TXS0	Transmit Shifter Writes transmit data.

**Caution** Since the UART does not have a transmit buffer, an interrupt request due to the end of transmission is not generated but an interrupt request (INTST) is generated in synchronization with the end of transmission of one frame of data.

### 8.2.4 Interrupt request

UART generates the following three types of interrupt requests:

- Receive error interrupt (INTSER)
- Reception completion interrupt (INTSR)
- Transmission completion interrupt (INTST)

Of these three, the receive error interrupt has the highest default priority, followed by the reception completion interrupt and transmission completion interrupt.

**Table 8-1. Default Priority of Interrupts**

Interrupt	Priority
Receive error	1
Reception completion	2
Transmission completion	3

#### (1) Receive error interrupt (INTSER)

A receive error interrupt occurs as a result of ORing the three types of receive errors described in description of the ASIS registers when reception is enabled.

This interrupt does not occur when reception is disabled.

#### (2) Reception completion interrupt (INTSR)

The reception completion interrupt occurs if data is received in the receive shift register and then transferred to the receive buffer when reception is enabled.

This interrupt also occurs when a receive error occurs, but the receive error interrupt has higher priority.

The reception completion interrupt does not occur when reception is disabled.

#### (3) Transmission completion interrupt (INTST)

Because the UART does not have a transmit buffer, a transmission completion interrupt occurs when one frame of transmit data including a 7-/8-/9-bit character is shifted out from the transmit shift register.

The transmission completion interrupt is output when the last bit of data has been transmitted.

8.2.5 Operation

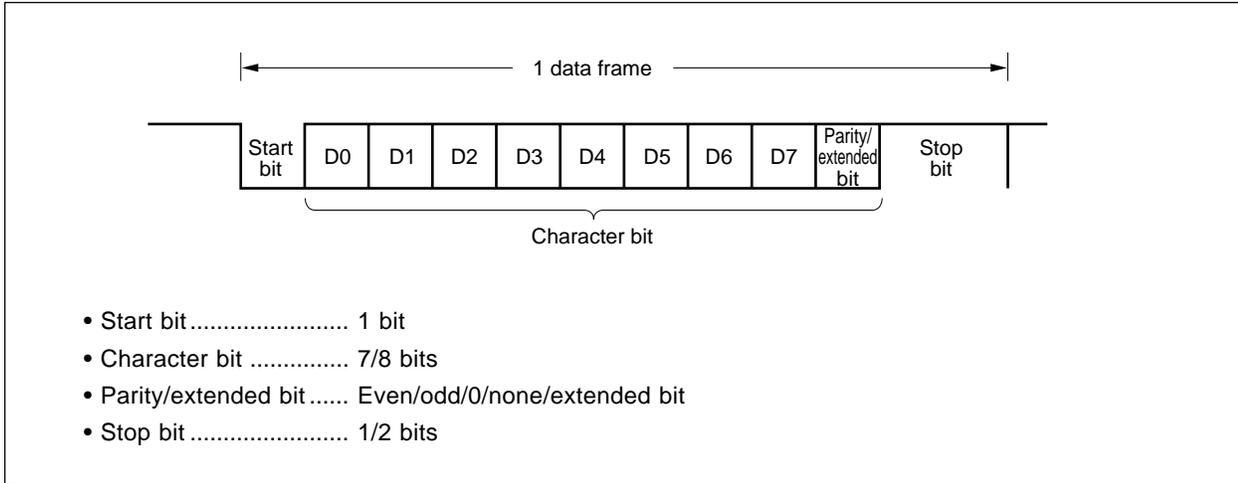
(1) Data format

Full-duplex serial data is transmitted/received.

One data frame of the transmit/receive data consists of a start bit, character bit, parity bit, and stop bit, as shown in Figure 8-2.

The length of the character bit, parity, and the length of the stop bit in one data frame are specified by the asynchronous serial interface mode registers (ASIMn).

Figure 8-2. Format of Transmit/Receive Data of Asynchronous Serial Interface



(2) Transmission

Transmission is started when data is written to the transmit shift registers (TXS or TXSL). The next data is written to the TXS or TXSL registers by the service routine of the transmission completion interrupt processing routine (INTST).

(a) Transmission enabled status

Set using the TXE bit of the ASIM0 register.

- TXE = 1 : Transmission enabled status
- TXE = 0 : Transmission disabled status

However, to set the transmission enabled status, set both the CTXE0 and CRXE0 bits of the clocked serial interface mode register (CSIM0) to "0".

Because the UART does not have a CTS (transmission enabled signal) input pin, use a general input port when checking whether the other is in the reception enabled status.

(b) Starting transmission

In the transmission enabled status, transmission starts when data is written to the transmission shift register (TXS or TXSL). The transmit data is transferred starting from the start bit with the LSB first. The start bit, parity bit, and stop bit are automatically appended.

Data cannot be written to the transmission shift register in the transmission disabled status, and values written are ignored.

**(c) Transmission interrupt request**

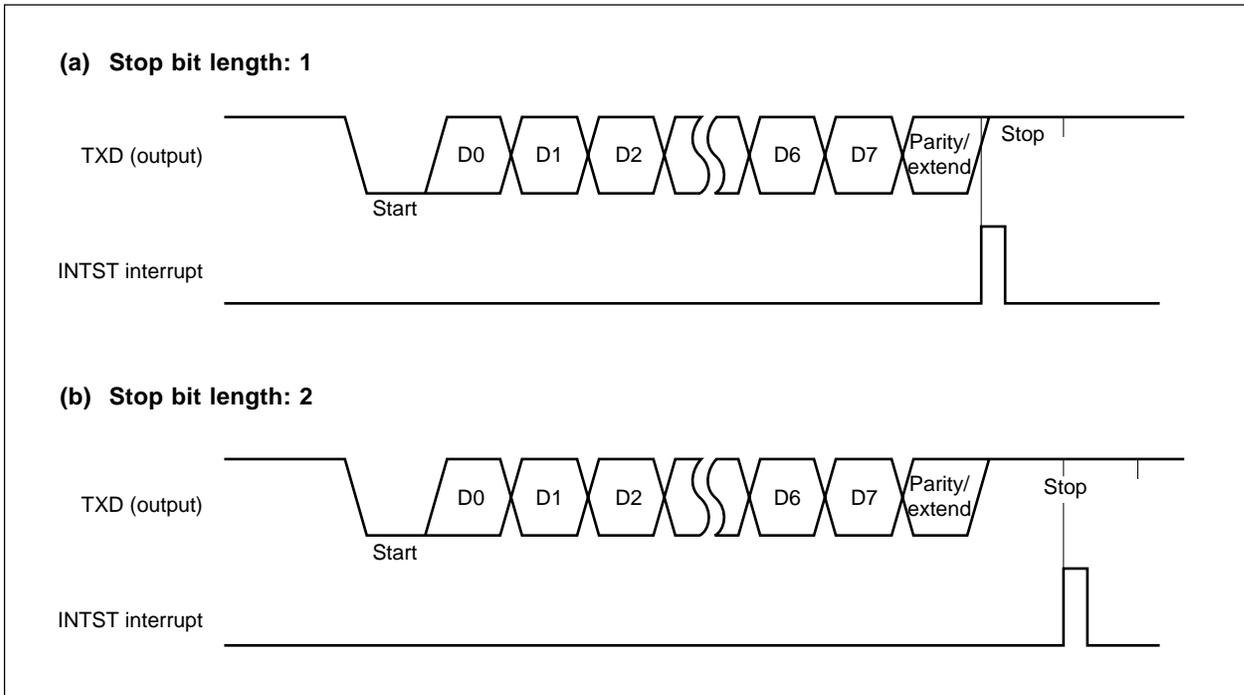
When one frame of data or character has been completely transferred, a transmission completion interrupt request (INTST) occurs.

Unless the data to be transmitted next is written to the TXS or TXSL registers, the transmission is aborted. The communication rate drops unless the next transmit data is written to the TXS or TXSL registers immediately after transmission has been completed.

**Cautions** 1. Generally, the transmission completion interrupt (INTST) is generated when the transmit shift register (TXS or TXSL) is empty. However, by  $\overline{\text{RESET}}$  input, the transmission completion interrupt (INTST) is not generated when the transmit shift register (TXS or TXSL) is empty.

2. During the transmit operation, writing data into the TXS or TXSL register is ignored (the data is discarded) until INTST is generated.

**Figure 8-3. Asynchronous Serial Interface Transmission Completion Interrupt Timing**



**(3) Reception**

When reception is enabled, sampling of the RXD pin is started, and reception of data begins when the start bit is detected. Each time one frame of data or character has been received, the reception completion interrupt (INTSR) occurs. Usually, the receive data is transferred from the receive buffer (RXB or RXBL) to memory by this interrupt processing.

**(a) Reception enabled status**

Reception is enabled when the RXE bits of the ASIM registers are set to 1.

RXE = 1: Reception is enabled

RXE = 0: Reception is disabled

However, to set the reception enabled status, set both the CTXE and CRXE bits of the clocked serial interface mode register (CSIM) to "0".

When reception is disabled, the receive hardware stands by in the initial status.

At this time, the reception completion interrupt/receive error interrupt does not occur, and the contents of the receive buffer are retained.

**(b) Starting reception**

Reception is started when the start bit is detected.

The RXD pin is sampled with the serial clock from baud rate generator. The RXD pin is sampled again eight clocks after the falling edge of the RXD pin has been detected. If the RXD pin is low at this time, it is recognized as the start bit, and reception is started. After that, the RXD pin is sampled in 16 clock ticks.

If the RXD pin is high eight clocks after the falling edge of the RXD pin has been detected, this falling edge is not recognized as the start bit. The serial clock counter is reinitialized, and the UART waits for the input of the next falling edge or valid start bit.

**(c) Reception completion interrupt request**

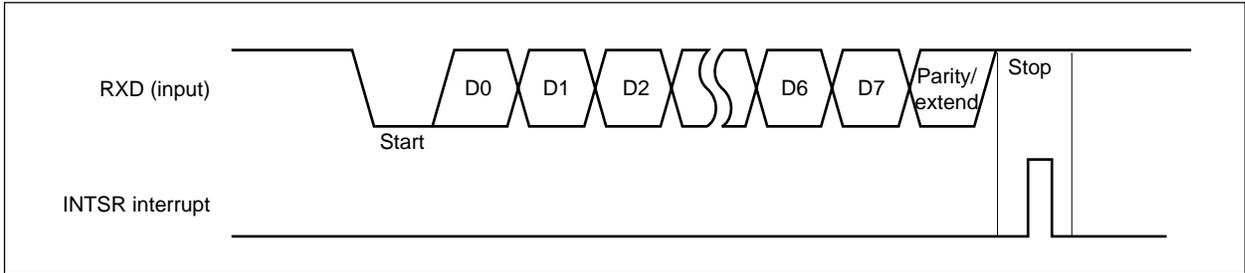
When one frame of data has been received with RXE = 1, the receive data in the shift register is transferred to RXB, and a reception completion interrupt request (INTSR) is generated.

If an error occurs, the receive data that contains an error is transferred to the receive buffer (RXB or RXBL), and the transmission completion interrupt (INTSR) and receive error interrupt (INTSER) occur simultaneously.

If the RXE bit is reset (0) during receive operation, the receive operation stops immediately. In this case, the contents of the receive buffer (RXB or RXBL) and the asynchronous serial interface status register (ASIS) do not change, and neither reception completion interrupt (INTSR) nor reception error interrupt (INTSER) is generated.

When RXE = 0 (reception disabled), no reception completion interrupt occurs.

Figure 8-4. Asynchronous Serial Interface Reception Completion Interrupt Timing



**(d) Reception error flag**

Three error flags, parity error, framing error, and overrun error flags, are related with the reception operation.

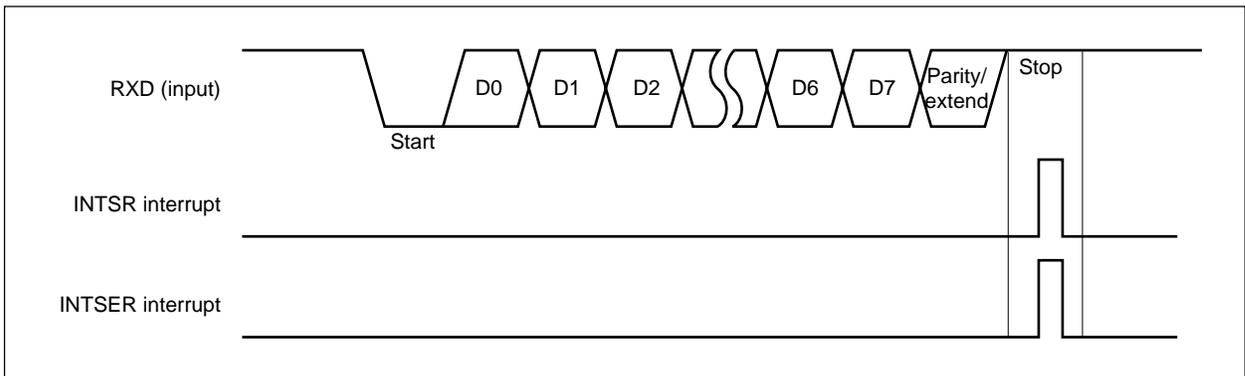
The receive error interrupt request occurs as a result of ORing these three error flags.

By reading the contents of the ASIS registers, the error which caused the receive error interrupt (INTSER) can be identified.

The contents of the ASIS registers are reset to 0 when the receive buffer (RXB or RXBL) is read or the next data frame is received (if the next data contains an error, the corresponding error flag is set).

Receive	Error Cause
Parity error	Parity specified during transmission does not coincide with parity of receive data
Framing error	Stop bit is not detected
Overrun error	Next data is completely received before data is read from receive buffer

Figure 8-5. Receive Error Timing



### 8.3 Clocked Serial Interface 0 to 3 (CSI0 to CSI3)

#### 8.3.1 Features

- Number of channels: 4 channels (CSIn)
- ★ ○ High transfer speed CSI0, CSI2, CSI3: 8.25 Mbps max. ( $\phi/4$  use,  $\phi = 33$  MHz)  
CSI1: 2.00 Mbps max.
- Half duplex communication
- Character length: 8 bits
- MSB first/LSB first selectable
- External serial clock input/internal serial clock output selectable
- 3 wires: SOn : serial data output  
SIn : serial data input  
SCKn : serial clock I/O
- Interrupt source: 4
  - Transmission/reception completion interrupt (INTCSIn)

**Remark** n = 0 to 3

$\phi$ : Internal system clock

#### 8.3.2 Configuration

CSIn is controlled by the clocked serial interface mode register (CSIMn). The transmit/receive data is read/written from/to the serial I/O shift register (SIO<sub>n</sub>) (n = 0 to 3).

##### (1) Clocked serial interface mode registers (CSIM0 to CSIM3)

CSIMn are 8-bit registers that specify the operation of CSIn.

##### (2) Serial I/O shift registers (SIO0 to SIO3)

SIO<sub>n</sub> registers are 8-bit registers that convert serial data into parallel data, and vice versa. SIO<sub>n</sub> are used for both transmission and reception.

Data is shifted in (received) or shifted out (transmitted) from the MSB or LSB side.

The actual transmitting and receiving of data is actually performed by writing data to and reading data from the SIO<sub>n</sub> registers.

##### (3) Selector

Selects the serial clock to be used.

##### (4) Serial clock control circuit

Controls supply of the serial clock to the shift register. When the internal clock is used, it also controls the clock output to the SCKn pin.

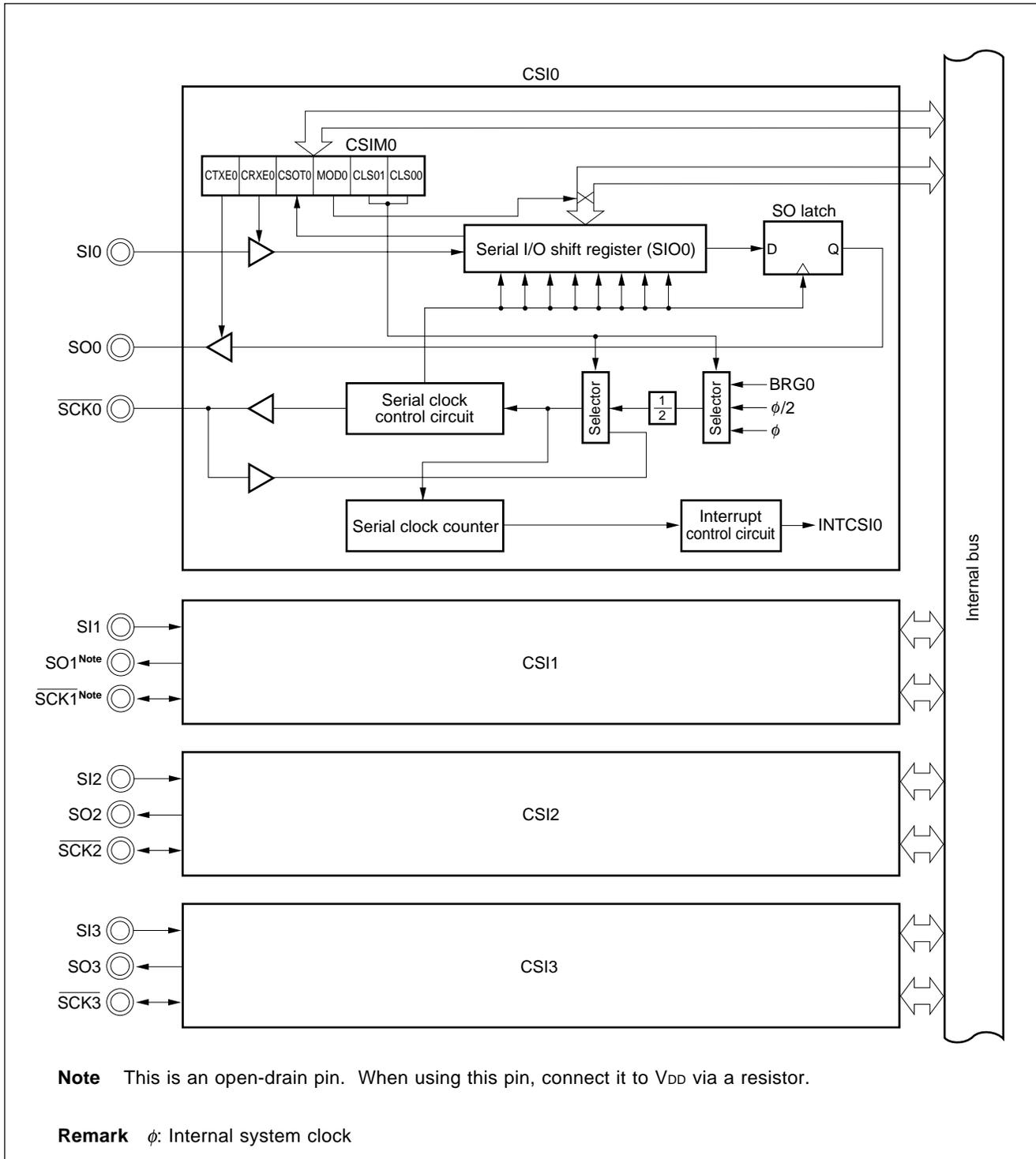
##### (5) Serial clock counter

Counts the serial clocks being output and the serial clocks received during transmission/reception to check whether 8-bit data has been transmitted or received.

##### (6) Interrupt control circuit

Controls whether an interrupt request is generated when the serial clock counter has counted eight serial clocks.

Figure 8-6. Block Diagram of Clocked Serial Interface



8.3.3 Control registers

(1) Clocked serial interface mode register 0 to 3 (CSIM0 to CSIM3)

These registers specify the basic operation mode of CSIn.

They can be read/written in 8- or 1-bit units (note, however, that bit 5 can only be read).

CSIM0	7	6	5	4	3	2	1	0	Address FFFFF088H	After reset 00H
	CTXE0	CRXE0	CSOT0	0	0	MOD0	CLS01	CLS00		
CSIM1	7	6	5	4	3	2	1	0	Address FFFFF098H	After reset 00H
	CTXE1	CRXE1	CSOT1	0	0	MOD1	CLS11	CLS10		
CSIM2	7	6	5	4	3	2	1	0	Address FFFFF0A8H	After reset 00H
	CTXE2	CRXE2	CSOT2	0	0	MOD2	CLS21	CLS20		
CSIM3	7	6	5	4	3	2	1	0	Address FFFFF0B8H	After reset 00H
	CTXE3	CRXE3	CSOT3	0	0	MOD3	CLS31	CLS30		

Bit Position	Bit Name	Function
7	CTXEn	CSI Transmit Enable Enables or disables transmission. 0 : Disables transmission 1 : Enables transmission When CTXEn = "0", both SOn and SIn pins go into high-impedance state.
6	CRXEn	CSI Receive Enable Disables or enables reception. 0 : Disables reception 1 : Enables reception If serial clock is received when both transmission is enabled (CTXEn = 1) and reception is disabled, "0" is input to shift register. If this bit is set to reception disabled status (CRXEn = 0) during receive operation, the contents of the SIO register become undefined.
5	CSOTn	CSI Status of Transmission Indicates that transfer operation is in progress. Set (1) : Transmission enable and start timing (writing to SIO0 register) Clear (0) : Transmission cleared and end timing (INTCSI occurs) This bit is used to check whether writing to serial I/O shift register (SIO) is permitted or not. Serial data transfer is started by enabling transmission (CTXEn = 1).
2	MODn	Mode Specifies operation mode. 0 : MSB first 1 : LSB first

**Remark** n = 0 to 3

Bit Position	Bit Name	Function																		
1, 0	CLS <sub>n</sub> 1, CLS <sub>n</sub> 0	<p>Clock Source Specifies serial clock.</p> <table border="1"> <thead> <tr> <th>CLS<sub>n</sub>1</th> <th>CLS<sub>n</sub>0</th> <th>Specifies Serial Clock</th> <th><math>\overline{\text{SCK}}_n</math> pin</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>External clock</td> <td>Input</td> </tr> <tr> <td>0</td> <td>1</td> <td rowspan="3">Internal clock</td> <td>Specified by BPRM<sub>n</sub> register<sup>Note 1</sup></td> </tr> <tr> <td>1</td> <td>0</td> <td><math>\phi/4</math><sup>Note 2</sup></td> </tr> <tr> <td>1</td> <td>1</td> <td><math>\phi/2</math><sup>Note 2</sup></td> </tr> </tbody> </table> <p><b>Notes</b> 1. For setting of BPRM<sub>n</sub> register, refer to section 8.5 Baud Rate Generator 0 to 3 (BRG0 to BRG3). 2. <math>\phi/4</math> and <math>\phi/2</math> indicate dividing signals (<math>\phi</math> : internal system clock).</p>	CLS <sub>n</sub> 1	CLS <sub>n</sub> 0	Specifies Serial Clock	$\overline{\text{SCK}}_n$ pin	0	0	External clock	Input	0	1	Internal clock	Specified by BPRM <sub>n</sub> register <sup>Note 1</sup>	1	0	$\phi/4$ <sup>Note 2</sup>	1	1	$\phi/2$ <sup>Note 2</sup>
CLS <sub>n</sub> 1	CLS <sub>n</sub> 0	Specifies Serial Clock	$\overline{\text{SCK}}_n$ pin																	
0	0	External clock	Input																	
0	1	Internal clock	Specified by BPRM <sub>n</sub> register <sup>Note 1</sup>																	
1	0		$\phi/4$ <sup>Note 2</sup>																	
1	1		$\phi/2$ <sup>Note 2</sup>																	

**Remark** n = 0 to 3

**Caution** Set the CLS<sub>n</sub>1 and CLS<sub>n</sub>0 bits, in the transmission/reception disable state (CTXE<sub>n</sub> bit = CRXE<sub>n</sub> bit = 0). If these bits are set in a state other than transmission/reception disabled, normal operation is not guaranteed.

**(2) Serial I/O shift register 0 to 3 (SIO0 to SIO3)**

These registers convert 8-bit serial data into parallel data, and vice versa. The actual transmitting and receiving of data is performed by writing data to and reading data from the SIO<sub>n</sub> registers.

A shift operation is performed when CTXE = "1" or CRXE = "1".

These registers can be read/written in 8- or 1-bit units.

SIO0	<table border="1"> <tr> <td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td>SIO07</td><td>SIO06</td><td>SIO05</td><td>SIO04</td><td>SIO03</td><td>SIO02</td><td>SIO01</td><td>SIO00</td> </tr> </table>	7	6	5	4	3	2	1	0	SIO07	SIO06	SIO05	SIO04	SIO03	SIO02	SIO01	SIO00	Address FFFFF08AH	After reset Undefined
7	6	5	4	3	2	1	0												
SIO07	SIO06	SIO05	SIO04	SIO03	SIO02	SIO01	SIO00												
SIO1	<table border="1"> <tr> <td>SIO17</td><td>SIO16</td><td>SIO15</td><td>SIO14</td><td>SIO13</td><td>SIO12</td><td>SIO11</td><td>SIO10</td> </tr> </table>	SIO17	SIO16	SIO15	SIO14	SIO13	SIO12	SIO11	SIO10	Address FFFFF09AH	After reset Undefined								
SIO17	SIO16	SIO15	SIO14	SIO13	SIO12	SIO11	SIO10												
SIO2	<table border="1"> <tr> <td>SIO27</td><td>SIO26</td><td>SIO25</td><td>SIO24</td><td>SIO23</td><td>SIO22</td><td>SIO21</td><td>SIO20</td> </tr> </table>	SIO27	SIO26	SIO25	SIO24	SIO23	SIO22	SIO21	SIO20	Address FFFFF0AAH	After reset Undefined								
SIO27	SIO26	SIO25	SIO24	SIO23	SIO22	SIO21	SIO20												
SIO3	<table border="1"> <tr> <td>SIO37</td><td>SIO36</td><td>SIO35</td><td>SIO34</td><td>SIO33</td><td>SIO32</td><td>SIO31</td><td>SIO30</td> </tr> </table>	SIO37	SIO36	SIO35	SIO34	SIO33	SIO32	SIO31	SIO30	Address FFFFF0BAH	After reset Undefined								
SIO37	SIO36	SIO35	SIO34	SIO33	SIO32	SIO31	SIO30												

Bit Position	Bit Name	Function
7 to 0	SIO <sub>n</sub> 7 to SIO <sub>n</sub> 0 (n = 0 to 3)	Serial I/O Data is shifted in (received) or out (transmitted) from MSB or LSB side.

8.3.4 Basic operation

(1) Transfer format

The CSI performs interfacing by using three lines: one clock line and two data lines.

Serial transfer is started by executing an instruction that writes transfer data to the SIO<sub>n</sub> register.

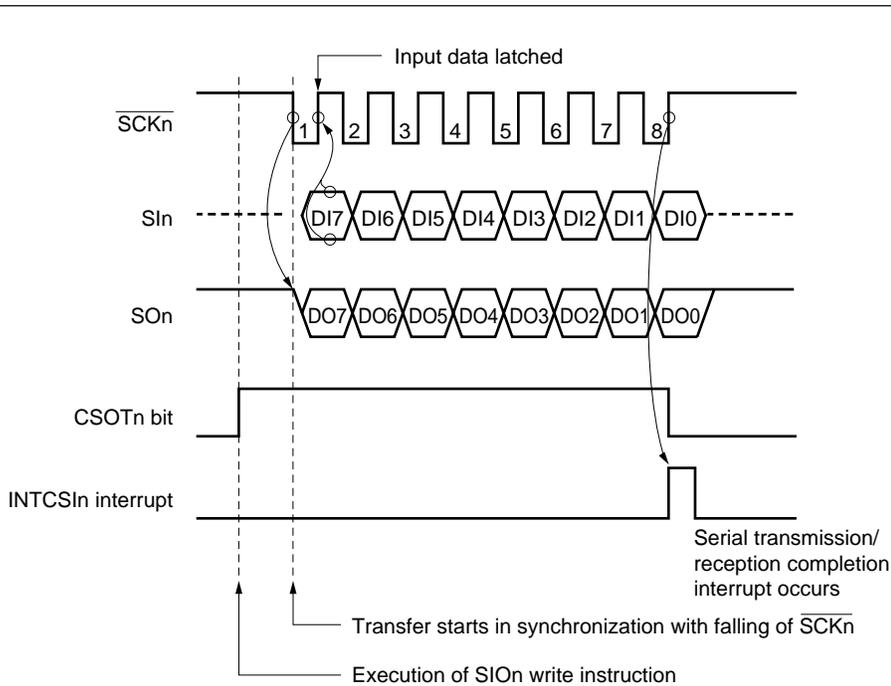
During transmission, the data is output from the SO<sub>n</sub> pin in synchronization with the falling edge of  $\overline{\text{SCKn}}$ .

During reception, the data input to the SI<sub>n</sub> pin is latched in synchronization with the rising edge of  $\overline{\text{SCKn}}$ .

$\overline{\text{SCKn}}$  stops when the serial clock counter overflows (at the rising edge of the 8th count), and  $\overline{\text{SCKn}}$  remains high until the next data transmission or reception is started. At the same time, a transmission/reception completion interrupt (INTCSI) is generated.

**Caution** If CTXE is changed from 0 to 1 after the transmit data is sent to the SIO<sub>n</sub> registers, serial transfer will not begin.

**Remark** n = 0 to 3



**Remark** n = 0 to 3

**(2) Enabling transmission/reception**

Each CSIn has only one 8-bit shift register and does not have a buffer. Transmission and reception are therefore performed simultaneously.

**(a) Transmission/reception enabling condition**

The transmission/reception enabling condition of CSIn is specified by the CTXEn and CRXEn bits of the CSIMn register.

CTXEn	CRXEn	Transmission/Reception Operation
0	0	Transmission/reception disabled
0	1	Reception enabled
1	0	Transmission enabled
1	1	Transmission/reception enabled

**Remark** n = 0 to 3

- Remarks**
1. When CTXEn bit is 0, SOn pin output of CSIn becomes high impedance. When CTXEn bit is 1, the data of the SIO n register of CSIn is output.
  2. When CRXEn bit = 0, the shift register input is "0". When CRXEn bit = 1, the serial input data is input to the shift register.
  3. To receive the transmit data and to check whether bus contention occurs, set CTXEn bit and CRXEn bit to 1.

**(b) Starting transmission/reception**

Transmission/reception is started by reading/writing the SIO n registers. Transmission/reception is controlled by setting the transmission enable bit (CTXEn) and reception enable bit (CRXEn) as follows:

CTXEn	CRXEn	Start Condition
0	0	Does not start
0	1	Reads SIO n registers
1	0	Writes SIO n registers
1	1	Writes SIO n registers
0	0 → 1	Rewrites CRXEn bits

**Remark** n = 0 to 3

In the above table, note that these bits should be set in advance of data transfer. For example, if the CTXEn bit is not changed from 0 to 1 before reading data from or writing data to the SIO n registers, transfer will not begin. The bottom of the table means that, if the CRXEn bit is changed from 0 to 1 when the CTXEn bit is "0", the serial clock will be generated to initiate receive operation.

8.3.5 Transmission in CSI0 to CSI3

Transmission is started when data is written to the SIO<sub>n</sub> register after transmission has been enabled by the clocked serial interface mode register (CSIM<sub>n</sub>) (n = 0 to 3).

(1) Starting transmission

Transmission is started by writing the transmit data to the shift register (SIO<sub>n</sub>) after the CTXE bit of the clocked serial interface mode registers (CSIM<sub>n</sub>) has been set (the CRXE bit is cleared to "0").

If the CTXE bit is reset to 0, the SIO<sub>n</sub> pin goes into a high-impedance state.

(2) Transmitting data in synchronization with serial clock

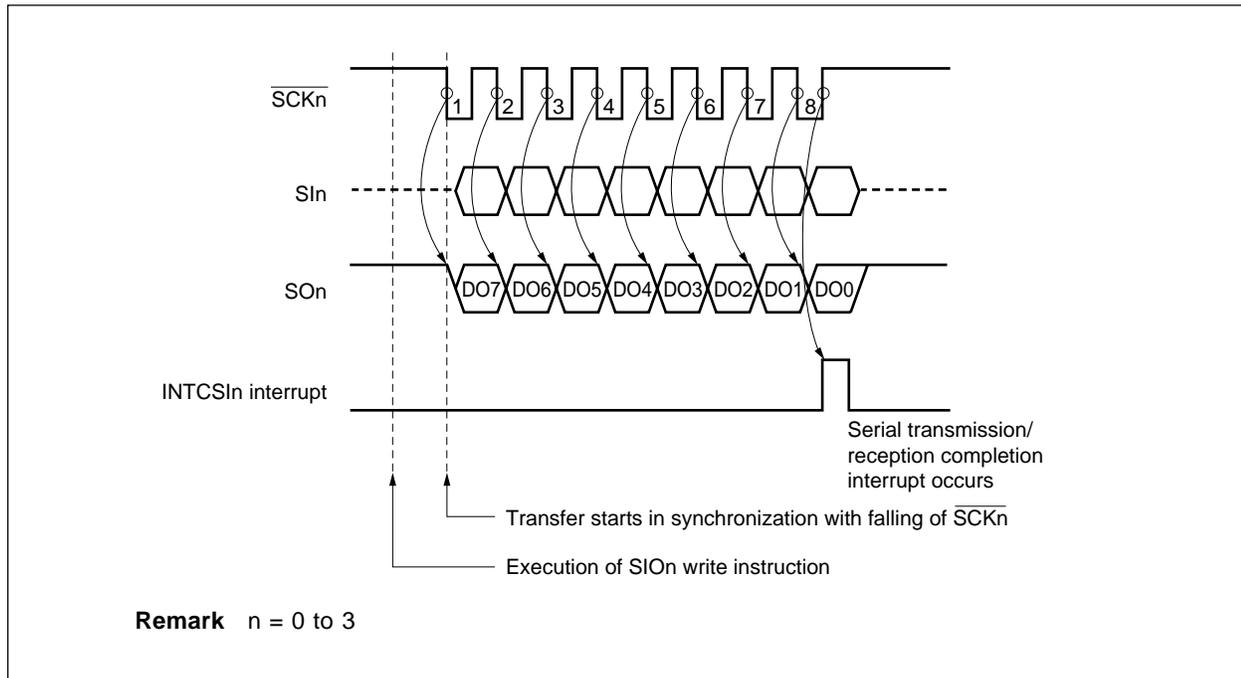
(a) When internal clock is selected as serial clock

When transmission is started, the serial clock is output from the  $\overline{SCKn}$  pin, and at the same time, data is sequentially output to the SIO<sub>n</sub> pin from SIO<sub>n</sub> in synchronization with the falling edge of the serial clock.

(b) When external clock is selected as serial clock

When transmission is started, the data is sequentially output from SIO<sub>n</sub> to the SIO<sub>n</sub> pin in synchronization with the falling of the serial clock input to the  $\overline{SCKn}$  pin immediately after transmission has been started. The shift operation is not performed even if the serial clock is input to the  $\overline{SCKn}$  pin if transmission is not enabled, and the output level of the SIO<sub>n</sub> pin will not change.

Figure 8-7. Timing of 3-Wire Serial I/O Mode (transmission)



**8.3.6 Reception in CSI0 to CSI3**

Reception is started if the status is changed from reception disabled to reception enabled status by the clocked serial interface mode registers (CSIMn) or if the SIO<sub>n</sub> registers are read by the CPU with reception enabled (n = 0 to 3).

**(1) Starting reception**

Reception can be started in the following two ways:

- <1> Changing the status of the CRXEn bits of the CSIMn registers from “0” (reception disabled) to “1” (reception enabled)
- <2> Reading the receive data from the shift registers (SIO<sub>n</sub>) when the CRXEn bits of the CSIMn registers are “1” (reception enabled)

If CRXEn bit of the CSIMn register has already been set to “1”, writing “1” to these bits do not initiate receive operation. When bit CRXEn = 0, the shift register inputs are “0”.

**(2) Receiving data in synchronization with serial clock**

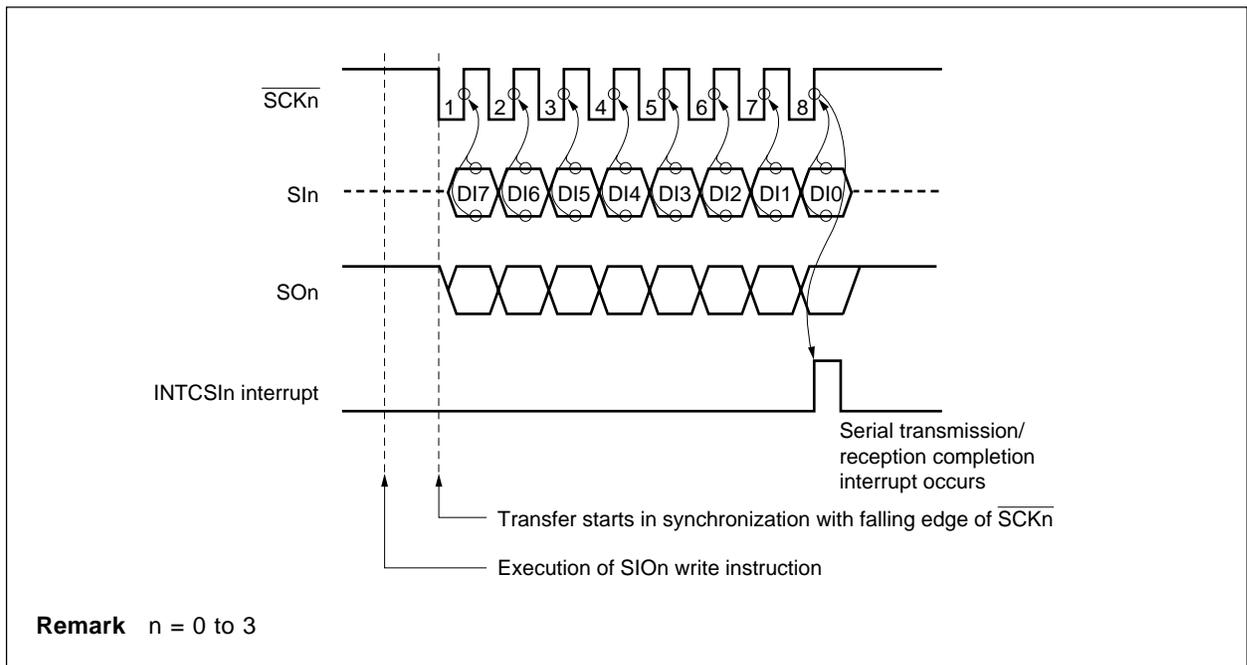
**(a) When internal clock is selected as serial clock**

When reception is started, the serial clock is output from the  $\overline{\text{SCKn}}$  pin, and at the same time, data is sequentially loaded from the SIn pin to SIO<sub>n</sub> in synchronization with the rising edge of the serial clock.

**(b) When external clock is selected as serial clock**

When reception is started, the data is sequentially loaded from the SIn pin to SIO<sub>n</sub> in synchronization with the rising of the serial clock input to the  $\overline{\text{SCKn}}$  pin immediately after reception has been started. The shift operation is not performed even if the serial clock is input to the  $\overline{\text{SCKn}}$  pin when reception is not enabled.

**Figure 8-8. Timing of 3-Wire Serial I/O Mode (reception)**



### 8.3.7 Transmission/reception in CSI0 to CSI3

Transmission and reception can be executed simultaneously if both transmission and reception are enabled by the clocked serial interface mode registers (CSIMn) (n = 0 to 3).

#### (1) Starting transmission/reception

Transmission and reception can be performed simultaneously (transmission/reception operation) when both the CTXEn and CRXEn bits of the clocked serial interface mode registers (CSIMn) are set to 1.

Transmission/reception can be started by writing the transmit data to the shift register n (SIO<sub>n</sub>) when both the CTXEn and CRXEn bits of the CSIMn register are “1” (transmission/reception enabled state)

If CRXEn bit of the CSIMn register has already been set to “1”, writing “1” to this bit does not initiate transmit/receive operation.

#### (2) Transmitting data in synchronization with serial clock

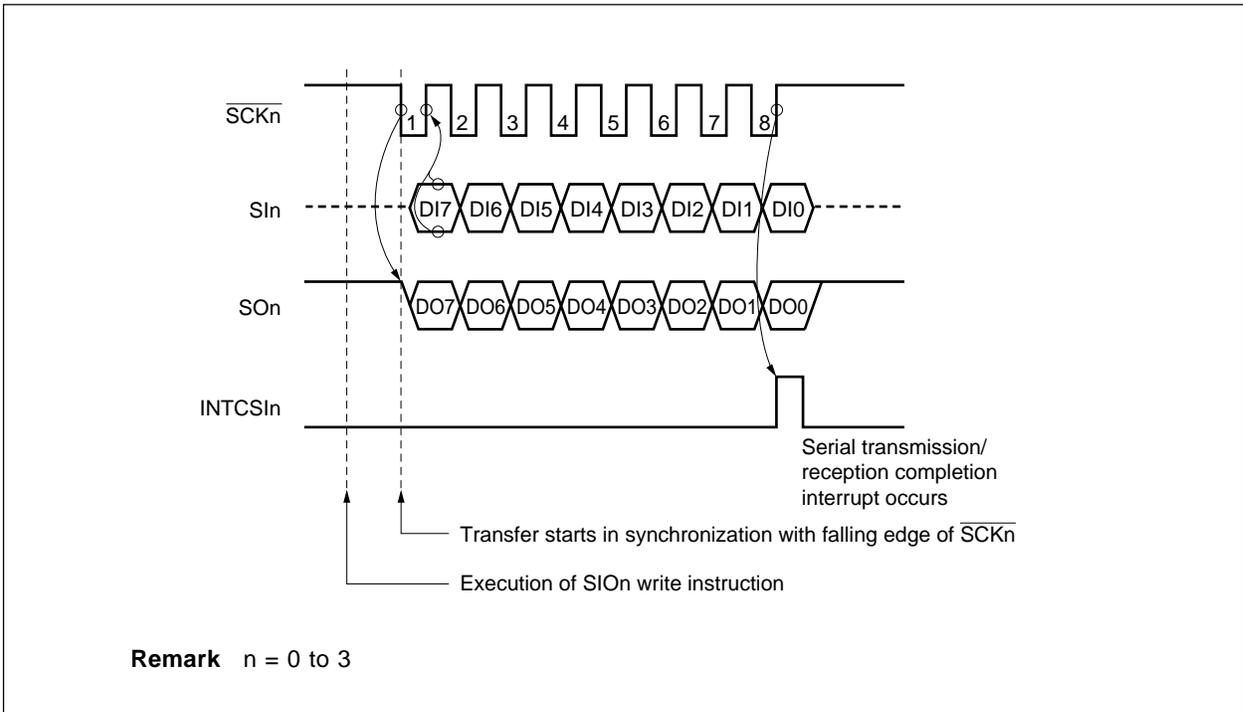
##### (a) When internal clock is selected as serial clock

When transmission/reception is started, the serial clock is output from the  $\overline{\text{SCKn}}$  pin, and at the same time, data is sequentially set to the SOn pin from SIO<sub>n</sub> in synchronization with the falling edge of the serial clock. Simultaneously, the data of the SIn pin is sequentially loaded to SIO<sub>n</sub> in synchronization with the rising edge of the serial clock.

##### (b) When external clock is selected as serial clock

When transmission/reception is started, the data is sequentially output from SIO<sub>n</sub> to the SOn pin in synchronization with the falling edge of the serial clock input to the  $\overline{\text{SCKn}}$  pin immediately after transmission/reception has been started. The data of the SIn pin is sequentially loaded to SIO<sub>n</sub> in synchronization with the rising edge of the serial clock. The shift operation is not performed even if the serial clock is input to the  $\overline{\text{SCKn}}$  pin when transmission/reception is not enabled, and the output level of the SOn pin does not change.

Figure 8-9. Timing of 3-Wire Serial I/O Mode (transmission/reception)



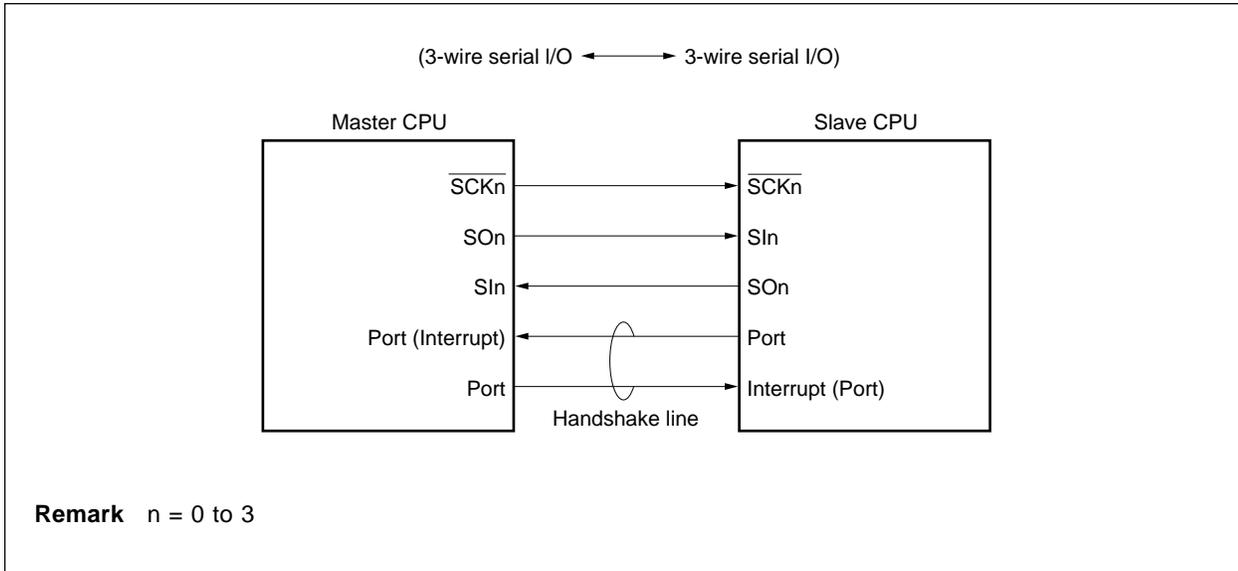
**8.3.8 System configuration example**

Data 8 bits long is transferred by using three types of signal lines: serial clock ( $\overline{\text{SCKn}}$ ), serial input (SIn), and serial output (SOn). This feature is effective for connecting peripheral I/Os and display controllers that have a conventional clocked serial interface.

To connect two or more devices, a handshake line is necessary.

Various devices can be connected, because it can be specified whether the data is transmitted starting from the MSB or LSB.

**Figure 8-10. Example of CSI System Configuration**



## 8.4 I<sup>2</sup>C Bus ( $\mu$ PD703008Y and 70F3008Y only)

### 8.4.1 Features

- I<sup>2</sup>C bus format
- Multi-master serial bus
- Serial data automatic discrimination function
  - Transfer speed: standard mode : 100 Kbps max.
  - high-speed mode : 400 Kbps max.
- Chip select by address
- Wake-up function
- Acknowledge signal ( $\overline{\text{ACK}}$ ) control function
- Wait signal ( $\overline{\text{WAIT}}$ ) control function
- Arbitration control function
- Conflict detection function

**8.4.2 Functions**

The following two modes are available for I<sup>2</sup>C bus.

- Operation stop mode
- I<sup>2</sup>C (Inter IC) bus mode (supports multi-master)

**(1) Operation stop mode**

Used when serial transfer is not carried out, reduces power consumption.

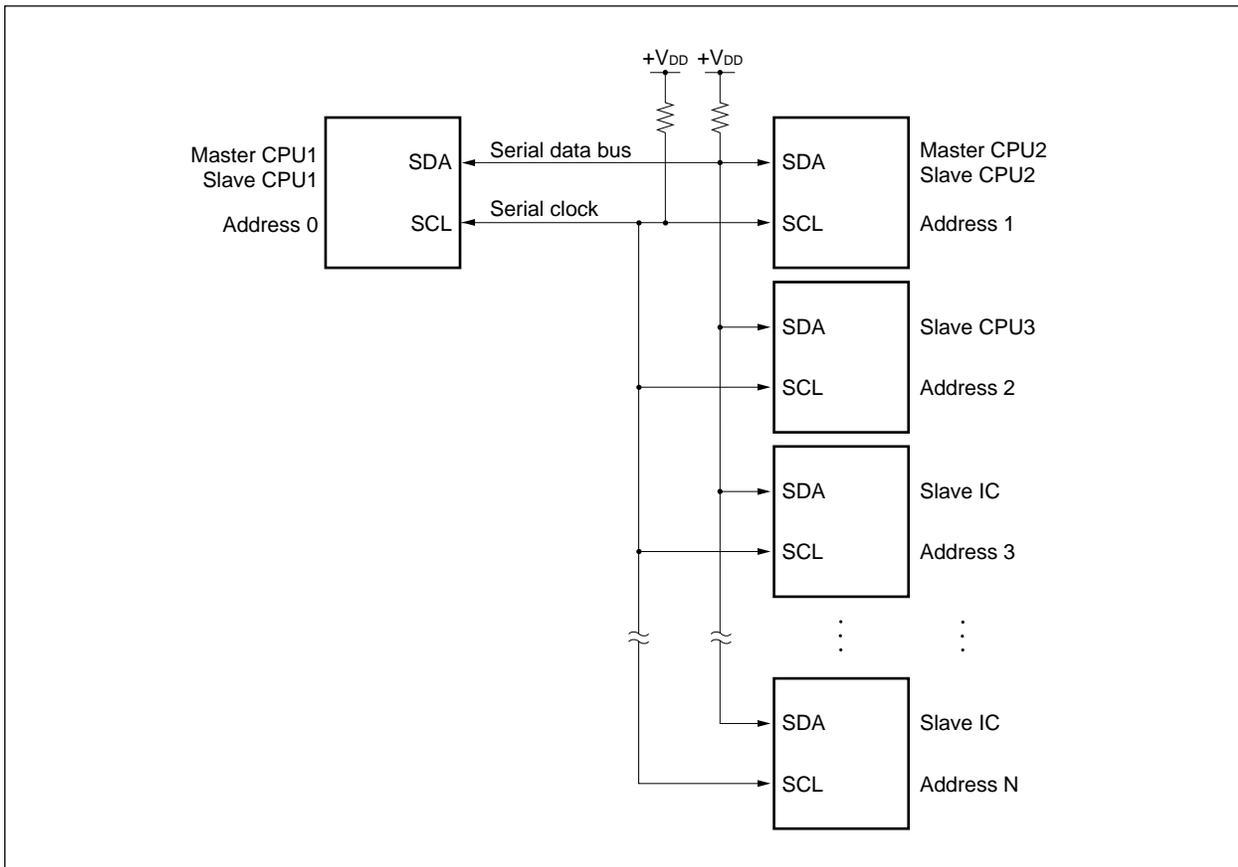
**(2) I<sup>2</sup>C bus mode (supports multi-master)**

Performs 8-bit data transfer with more than one device, using two lines, each for the serial clock (SCL) and the serial data bus (SDA).

Conforming to I<sup>2</sup>C bus format, either “start condition”, “data”, or “stop condition” can be output onto serial data bus in transmission. These data can be automatically detected by hardware in reception.

Because SCL and SDA are open drain outputs, pull-up resistors are required for the serial clock line and the serial data bus line.

**Figure 8-11. Example of Serial Bus Configuration Using I<sup>2</sup>C Bus**





**8.4.3 Configuration**

The I<sup>2</sup>C bus is configured with the following hardware.

**Table 8-2. I<sup>2</sup>C Bus Configuration**

Item	Configuration
Register	IIC shift register (IIC) Slave address register (SVA)
Control register	IIC clock selection register (IICCL) IIC status register (IICS) IIC control register (IICC)

**(1) IIC shift register (IIC)**

IIC is a register that converts 8-bit serial data into parallel data, and vice versa. IIC is used both for transmission and reception.

The actual transmission and reception of data is performed by writing data to and reading data from the IIC shift register.

This register is set by 8-bit memory manipulation instruction. It becomes 00H by  $\overline{\text{RESET}}$  input.

**(2) Slave address register (SVA)**

SVA is a register that sets the local address with 8-bit memory manipulation instruction when used as a slave. It becomes 00H by  $\overline{\text{RESET}}$  input.

**(3) SO latch**

Retains SDA pin output level.

**(4) Wake-up control circuit**

Generates interrupt requests when the address value set in the slave address register (SVA) and the reception address coincide or when an extension code is received.

**(5) Clock selector**

Selects the sampling clock to be used.

**(6) Serial clock counter**

Counts the serial clocks to be output or input in transmission/reception and checks whether 8-bit data has been transmitted/received.

**(7) Interrupt request signal generation circuit**

Controls generation of interrupt request signals.

I<sup>2</sup>C interrupt is generated by the following two triggers.

- The eighth or the ninth count of the serial clock (set by WTIM bit<sup>Note</sup>).
- The generation of an interrupt by the stop condition detection (set by SPIE bit<sup>Note</sup>).

**Note** WTIM bit : bit 3 of IIC control register (IICC)  
SPIE bit : bit 4 of IIC control register (IICC)

**(8) Serial clock control circuit**

Generates clocks to be output to SCL pin from the sampling clock in the master mode.

**(9) Serial clock wait control circuit**

Controls wait timings.

**(10) Acknowledge output circuit, stop condition detection circuit, start condition detection circuit, acknowledge detection circuit**

Performs output and detection of various control signals.

**(11) Data retention time correction circuit**

Generates data retention time for the fall of the serial clock.

#### 8.4.4 Serial interface control register

I<sup>2</sup>C bus is controlled by the following three registers.

- IIC control register (IICC)
- IIC status register (IICS)
- IIC clock selection register (IICCL)

The following registers are also used.

- IIC shift register (IIC)
- Slave address register (SVA)

**(1) IIC control register (IICC)**

This register performs enabling/disabling I<sup>2</sup>C operation, setting of wait timing, and other settings of I<sup>2</sup>C operation.

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

It becomes 00H by RESET input.

IICC	7	6	5	4	3	2	1	0	Address FFFFF0E0H	After reset 00H
	IICE	LREL	WREL	SPIE	WTIM	ACKE	STT	SPT		

Bit Position	Bit Name	Function
7	IICE	I <sup>2</sup> C Enable Enables or disables I <sup>2</sup> C operation. The data in IICC is not cleared. 0 : Disables I <sup>2</sup> C operation. Presets I <sup>2</sup> C status register (IICS) and disables internal operation. 1 : Enables I <sup>2</sup> C operation  Set condition : Instruction Clear condition : Instruction, reset input
6	LREL	Line Release Exits from the communication in progress, releases the bus, and waits for the start of the next communication. 0 : Normal operation 1 : Exits from communication and enters wait status This bit is used when an extension code not related to the local register is received. SCL and SDA lines go into high-impedance status. The following flags are cleared. EXC, ACKD, COI, MSTs, TRC, STD  Set condition : Instruction Clear condition : Reset input, or automatically cleared after execution.
5	WREL	Wait Release Selects whether releasing wait or not. 0 : Does not release wait 1 : Releases wait  Set condition : Instruction Clear condition : Reset input, or automatically cleared after execution.  <b>Caution</b> If wait is released when TRC = 1 and WREL are set to the ninth clock, TRC is cleared and SDA line goes into high impedance status.
4	SPIE	Stop Condition Interrupt Enable Disable or enables generation of INTIIC interrupt request by stop condition detection. 0 : Disables 1 : Enables  Set condition : Instruction Clear condition : Instruction, reset input

Bit Position	Bit Name	Function
3	WTIM	<p>Wait Timing Controls generation of wait and interrupt request.</p> <p>0 : Interrupt request generates at the fall of the eighth clock For master : Waits keeping clock output in low level after outputting eight clocks For slave : Waits for master setting clock output in low level after inputting eight clocks</p> <p>1 : Interrupt request generates at the fall of the ninth clock For master : Waits keeping clock output in low level after outputting nine clocks For slave : Waits for master setting clock output in low level after inputting nine clocks</p> <p>The setting of this bit becomes invalid while transferring address. When EXC = 1, wait is compulsorily inserted at the eighth clock, and when COI = 1, wait is compulsorily inserted at the ninth clock, and then the setting of this bit becomes valid. For master, wait is inserted at the ninth clock while transferring address. The slave which has received the local address enters wait status at the fall of the ninth clock after the generation of acknowledge.</p> <p>Set condition : Instruction Clear condition : Instruction, reset input</p>
2	ACKE	<p>Acknowledge Enable Controls acknowledge.</p> <p>0 : Disables acknowledge 1 : Enables automatic acknowledge output. SDA line becomes low level during the 9-clock period. However, it is invalid while transferring address and valid when EXC = 1.</p> <p>Set condition : Instruction Clear condition : Instruction, reset input</p>
1	STT	<p>Start Condition Trigger Selects whether generating start condition or not.</p> <p>0 : Does not generate start condition 1 : Generates start condition</p> <p>When bus is released (stop status): generates start condition by changing SDA line from high-level to low-level (starts up as master). And then, secures specification time and sets SCL to low level.</p> <p>When not joining bus (STT is set after STD =1 is set): this bit functions as a start condition reservation flag. When this bit is set, it automatically generates start condition after bus is released.</p> <p>Set condition : Instruction Clear condition : Instruction, reset input, when start condition is detected for master, (MSTS = 1 and STD = 1), or when defeated in arbitration.</p>
0	SPT	<p>Stop Condition Trigger Selects whether generating stop condition or not.</p> <p>If this bit is set in the early stage of communication, it outputs low-level to SDA line and generates stop condition.</p> <p>0 : Does not generate stop condition 1 : Generates stop condition</p> <p>Sets SCL line to high level after setting SDA line to low level, or waits SCL becomes high level. And then, it secures specification time , changes SDA line from low level to high level, and generates stop condition.</p> <p>Set condition : Instruction Clear condition : Instruction, reset input, when stop condition is detected, or when defeated in arbitration.</p>

**Caution** When performing transmission/reception of master between enabling operation (IICE = 1) and the first stop condition detection, generate a stop condition once by setting the SPT bit.

**(2) IIC status register (IICS)**

IIC status register indicates the I<sup>2</sup>C status.

This register is set by 1- or 8-bit memory manipulation instruction. It can only be read.

It becomes 00H by  $\overline{\text{RESET}}$  input.

	7	6	5	4	3	2	1	0		
IICS	MSTS	ALD	EXC	COI	TRC	ACKD	STD	SPD	Address FFFFFF0E2H	After reset 00H

Bit Position	Bit Name	Function
7	MSTS	Master Status Indicates master status. 0 : Slave status or communication wait status 1 : Master status  Set condition : Start condition generation Clear condition : Stop condition detection, ALD = 1, LREL = 1, IICE = 1 -> 0, or reset input
6	ALD	Arbitration Defeat Detects arbitration defeat. 0 : Arbitration has not occurred, or win in arbitration 1 : Defeat in arbitration. MSTS flag is cleared.  Set condition : Arbitration defeat Clear condition : IICE = 0, reset input, or after IICS read. A bit manipulation instruction is executed to another bit of the IICS register.
5	EXC	Extension Code Indicates reception of extension code. 0 : Extension code is not received 1 : Extension code is received  Set condition : The higher 4 bits of the received address are 0000 or 1111 (set at the rise of the eighth clock). Clear condition : Start condition detection, stop condition detection, LREL= 1, IICE = 0, or reset input
4	COI	Coincident Address Indicates coincidence of received address and local address. 0 : Addresses do not coincide 1 : Addresses coincide  Set condition : Coincidence of received address and local address (SVA) (set at the rise of the eighth clock) Clear condition : Start condition detection, LREL = 1, IICE = 0, or reset input

Bit Position	Bit Name	Function
3	TRC	<p>Transmit/Receive Condition Indicates current communication status.</p> <p>0 : Receiving status (status other than transmitting status). SDA line goes into high-impedance state.</p> <p>1 : Transmitting status. The value of SO latch can be output to SDA line (valid after the fall of the ninth clock of the first byte).</p> <p>Set condition : Start condition generation (STD = 1 and MSTs = 1) for master. 1 is input to the LSB (transfer direction specification bit) of the first byte for slave.</p> <p>Clear condition : LREL = 1, IICE = 0, ALD = 1, reset input, stop condition detection. 1 is output to the LSB (transfer direction specification bit) of the first byte for master.</p> <p>Start condition detection (STD = 1 and MSTs = 0) for slave when not joining communication.</p>
2	ACKD	<p>Acknowledge Detected Indicates detection of acknowledge signal.</p> <p>0 : Not detected 1 : Detected</p> <p>Set condition : SDA line becomes low-level at the rise of the ninth clock of SCL.</p> <p>Clear condition : LREL = 1, the rise of the first clock of the next byte, stop condition detection, IICE = 0, or reset input</p>
1	STD	<p>Start Condition Detected Indicates detection of start condition.</p> <p>0 : Start condition not detected 1 : Start condition detected. Indicates address transfer period.</p> <p>Set condition : Start condition detection</p> <p>Clear condition : Stop condition detection, LREL = 1, the rise of the first clock of the byte following the address transfer period, IICE = 0, or reset input</p>
0	SPD	<p>Stop Condition Detected Indicates detection of stop condition.</p> <p>0 : Stop condition not detected 1 : Stop condition detected. Communication in the master ends. Bus is released.</p> <p>Set condition : Stop condition detection</p> <p>Clear condition : The rise of the first clock of the address transfer byte after start condition detection and after setting this bit, IICE = 0, or reset input. Cleared with clock input even without start condition.</p>

**(3) IIC clock selection register (IICCL)**

IICCL is a register that sets the transfer clock of I<sup>2</sup>C.

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

It becomes 00H by  $\overline{\text{RESET}}$  input.

	7	6	5	4	3	2	1	0		
IICCL	0	0	CLD <sup>Note</sup>	DAD <sup>Note</sup>	SMC	DFC	CL1	CL0	Address FFFFF0E4H	After reset 00H

Bit Position	Bit Name	Function																																																
5	CLD	SCL Level Detected Indicates SCL line level detection result. Valid only when IICE = 1. 0 : SCL line is low level. 1 : SCL line is high level.																																																
4	DAD	SDA Level Detected Indicates the SDA line level detection result. Valid only when IICE = 1. 0 : SDA line is low level. 1 : SDA line is high level.																																																
3	SMC	Speed Mode Control Switches operation mode. 0 : Standard mode 1 : High-speed mode																																																
2	DFC	Digital Filter Condition Enables or disables digital filter operation. 0 : OFF 1 : ON Digital filter can be used in high-speed mode. The response becomes slow when digital filter is used.																																																
1, 0	CL1, CL0	Clock Selects internal clock according to the system clock. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th rowspan="3" style="width: 5%;">CL1</th> <th rowspan="3" style="width: 5%;">CL0</th> <th colspan="3" style="text-align: center;">Standard Mode</th> <th colspan="3" style="text-align: center;">High-speed Mode</th> </tr> <tr> <th colspan="2" style="text-align: center;">Transfer clock</th> <th colspan="2" style="text-align: center;">Transfer clock</th> </tr> <tr> <th style="text-align: center;">DFC=0</th> <th style="text-align: center;">DFC=1</th> <th style="text-align: center;">DFC=0</th> <th style="text-align: center;">DFC=1</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>\phi = 4.0</math> to 8.0 MHz</td> <td style="text-align: center;"><math>\phi/88</math></td> <td style="text-align: center;"><math>\phi/92</math></td> <td style="text-align: center;"><math>\phi = 8.0</math> to 16.0 MHz</td> <td style="text-align: center;"><math>\phi/48</math></td> <td style="text-align: center;"><math>\phi/48</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>\phi = 10.0</math> to 16.0 MHz</td> <td style="text-align: center;"><math>\phi/172</math></td> <td style="text-align: center;"><math>\phi/176</math></td> <td style="text-align: center;"><math>\phi = 8.0</math> to 16.0 MHz</td> <td style="text-align: center;"><math>\phi/48</math></td> <td style="text-align: center;"><math>\phi/48</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>\phi = 16.0</math> to 33.0 MHz</td> <td style="text-align: center;"><math>\phi/344</math></td> <td style="text-align: center;"><math>\phi/352</math></td> <td style="text-align: center;"><math>\phi = 16.0</math> to 33.0 MHz</td> <td style="text-align: center;"><math>\phi/96</math></td> <td style="text-align: center;"><math>\phi/96</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Setting prohibited</td> <td style="text-align: center;">—</td> <td style="text-align: center;">—</td> <td style="text-align: center;">Setting prohibited</td> <td style="text-align: center;">—</td> <td style="text-align: center;">—</td> </tr> </tbody> </table> <p><b>Remark</b>      <math>\phi</math>: Internal system clock</p>	CL1	CL0	Standard Mode			High-speed Mode			Transfer clock		Transfer clock		DFC=0	DFC=1	DFC=0	DFC=1	0	0	$\phi = 4.0$ to 8.0 MHz	$\phi/88$	$\phi/92$	$\phi = 8.0$ to 16.0 MHz	$\phi/48$	$\phi/48$	0	1	$\phi = 10.0$ to 16.0 MHz	$\phi/172$	$\phi/176$	$\phi = 8.0$ to 16.0 MHz	$\phi/48$	$\phi/48$	1	0	$\phi = 16.0$ to 33.0 MHz	$\phi/344$	$\phi/352$	$\phi = 16.0$ to 33.0 MHz	$\phi/96$	$\phi/96$	1	1	Setting prohibited	—	—	Setting prohibited	—	—
CL1	CL0	Standard Mode			High-speed Mode																																													
		Transfer clock			Transfer clock																																													
		DFC=0	DFC=1	DFC=0	DFC=1																																													
0	0	$\phi = 4.0$ to 8.0 MHz	$\phi/88$	$\phi/92$	$\phi = 8.0$ to 16.0 MHz	$\phi/48$	$\phi/48$																																											
0	1	$\phi = 10.0$ to 16.0 MHz	$\phi/172$	$\phi/176$	$\phi = 8.0$ to 16.0 MHz	$\phi/48$	$\phi/48$																																											
1	0	$\phi = 16.0$ to 33.0 MHz	$\phi/344$	$\phi/352$	$\phi = 16.0$ to 33.0 MHz	$\phi/96$	$\phi/96$																																											
1	1	Setting prohibited	—	—	Setting prohibited	—	—																																											

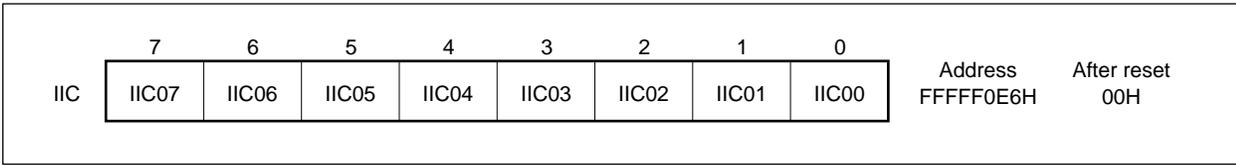
★ **Note** Bit 4 and bit 5 are Read Only bits.

**(4) IIC shift register (IIC)**

IIC shift register performs serial transmission/reception (shift operation) in synchronization with the serial clock.

This register can be read/written in 8-/1-bit units. However, do not write data to IIC register during data transfer.

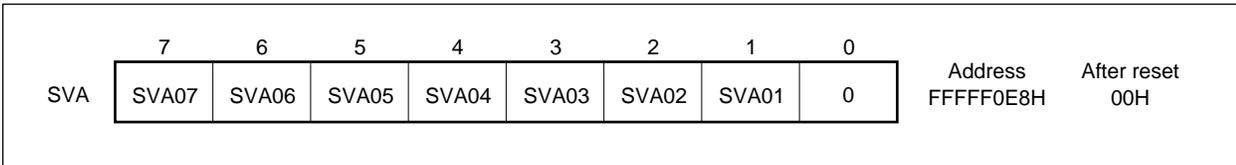
★ **Caution** In operations during restart, carry out writing of address data to the IIC shift register after the start conditions trigger is cleared and the start conditions are detected.



**(5) Slave address register (SVA)**

Slave address register stores slave address of I<sup>2</sup>C bus.

This register can be read/written in 8-/1-bit units. However, bit 0 is fixed to 0.



**8.4.5 I<sup>2</sup>C bus functions**

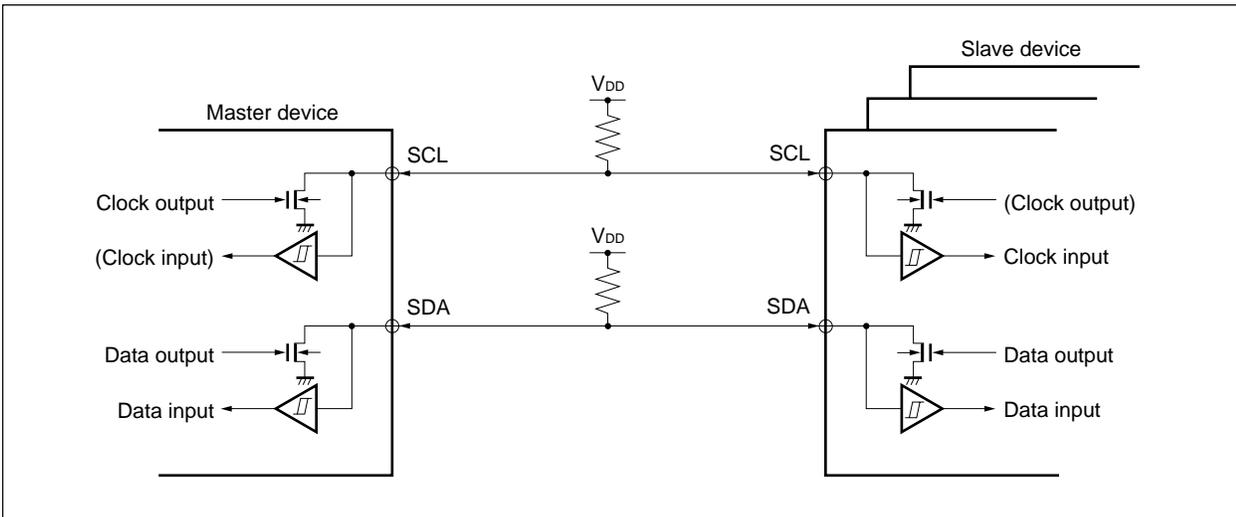
**(1) Pin configuration**

The configuration of the serial clock pin (SCL) and the serial data bus pin (SDA) is as follows.

- (a) SCL ..... Pin to input/output serial clock  
Output is N-ch open drain both for master and slave. Input is Schmitt input.
- (b) SDA ..... I/O multiplexed pin for serial data  
Output is N-ch open drain both for master and slave. Input is Schmitt input.

Because the outputs of serial clock line and the serial data bus line are N-ch open drain, external pull-up resistors are required.

**Figure 8-13. Pin Configuration**

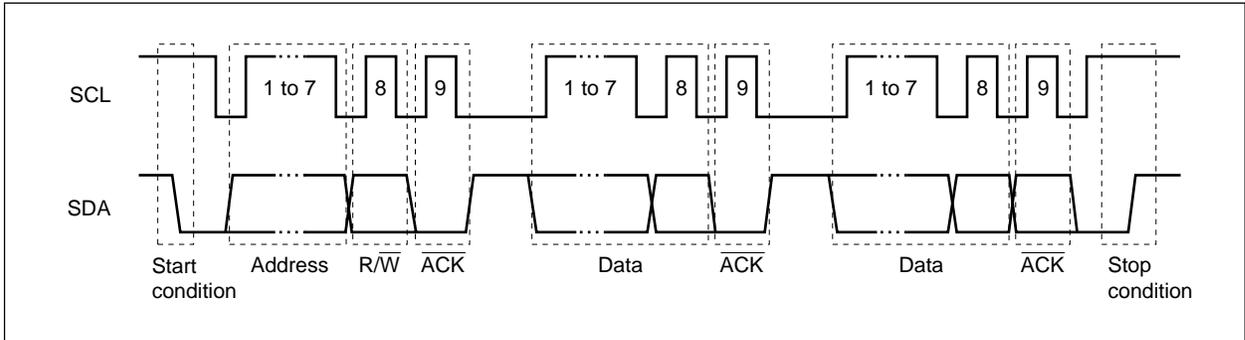


**8.4.6 Definition and controls of I<sup>2</sup>C bus**

The serial data communication format of I<sup>2</sup>C bus and the signals used are explained below.

Figure 8-14 shows each transfer timing of “start condition”, “data”, and “stop condition”, which are output onto the serial data bus of I<sup>2</sup>C bus.

**Figure 8-14. Serial Data Transfer Timing of I<sup>2</sup>C Bus**



Start condition, slave address, and stop condition are output from the master.

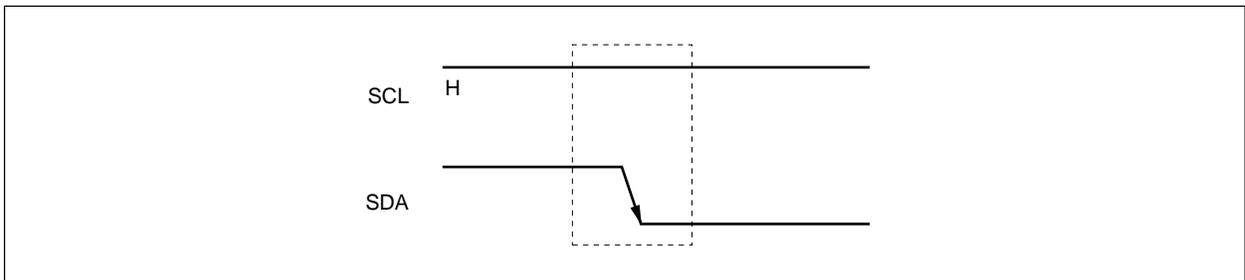
The acknowledge signal ( $\overline{\text{ACK}}$ ) is output either from the master or the slave (normally, it is output from the reception side of 8-bit data)

The serial clock (SCL) is kept being output from the master. However, the slave can extend the low level period of SCL and insert wait.

**(1) Start condition**

The start condition is generated when the SDA pin changes from high level to low level while the SCL pin is high level. The start condition of the SCL pin and the SDA pin is a signal which is the master outputs to the slave when a serial transfer starts. The slave is provided with the hardware to detect the start condition.

**Figure 8-15. Start Condition**



Start condition is output by setting (1) the STT bit of the IICC register in the stop condition detection status (the SPD bit of the IICS register = 1). When the start condition is detected, the STD bit of the IICS register is set (1).

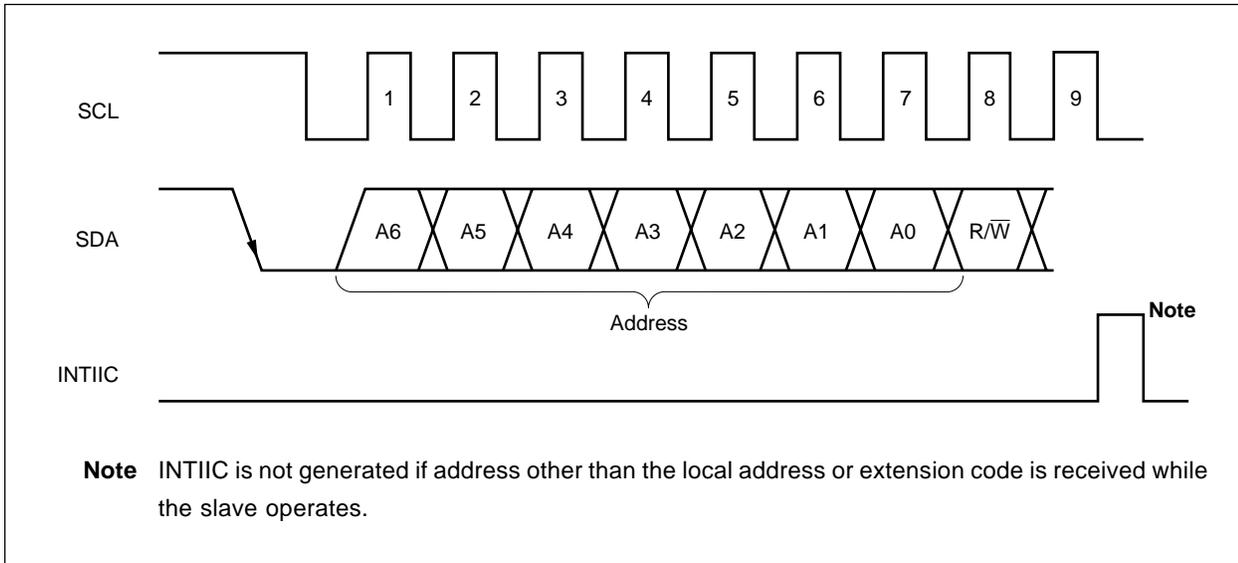
**(2) Address**

The 7-bit data following the start condition is defined as an address.

The address is an 7-bit data which the master outputs to select a specific slave from the two or more slaves connected to the bus line. Therefore, the slaves on the bus line must be assigned different addresses.

The slave detects this condition by hardware, and, in addition, checks if the 7-bit data coincides with the slave address register (SVA). When the 7-bit data and the value of SVA coincide, a slave is selected, and the slave performs communication with the master until the master transmits a start or stop condition.

**Figure 8-16. Address**



The address is output when the 8-bit address consisting of the slave address and the transfer direction explained in (3) Transfer direction specification are written to the shift register (IIC). The received address is written to IIC.

The slave address is assigned to the higher 7 bits of IIC.

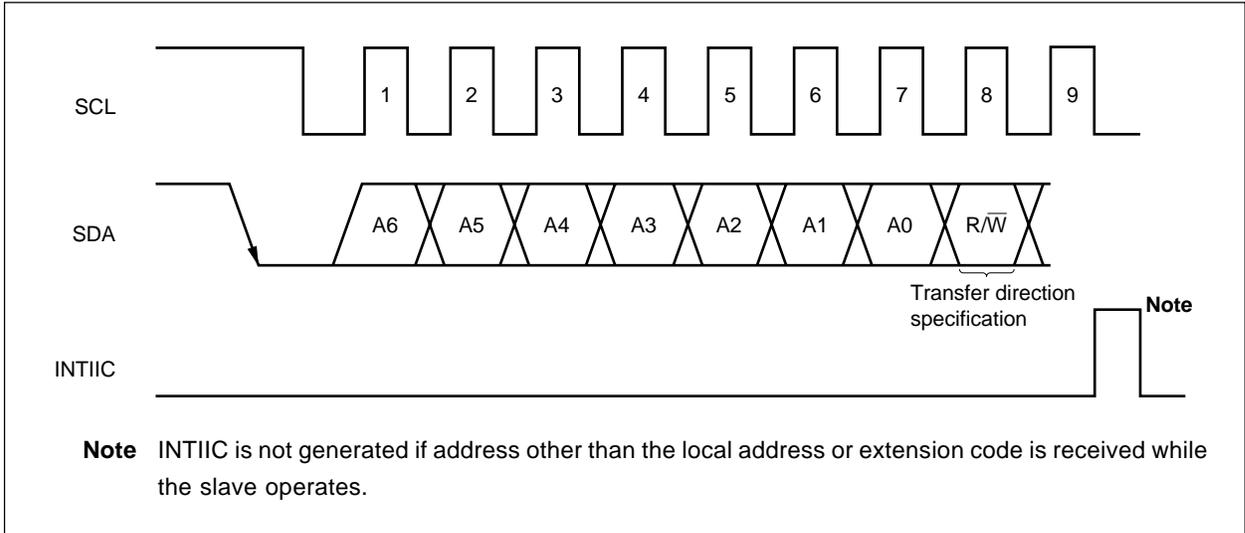
**(3) Transfer direction specification**

The master transmits 1-bit data because it specifies the transmit direction following the 7-bit address.

When the transmit direction specification bit is 0, the master transmits data to the slave.

When the transmit direction specification bit is 1, the master receives data from the slave.

**Figure 8-17. Transfer Direction Specification**



**(4) Acknowledge signal ( $\overline{\text{ACK}}$ )**

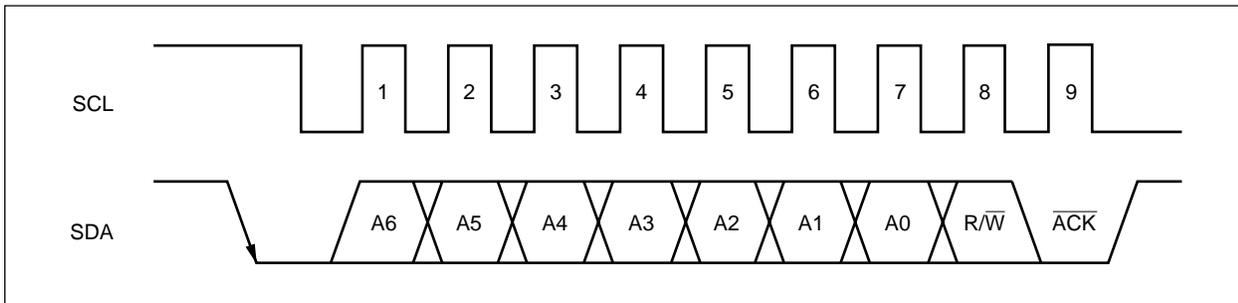
The acknowledge signal is a signal for checking serial data reception in the transmit and receive sides. The receive side sends back the acknowledge signal each time it receives an 8-bit data. The transmit side receives the acknowledge signal after transmitting the 8-bit data. However, in the case the master is the receive side, no acknowledge signal is output if the master receives the final data. The transmit side detects whether the receive side has sent the acknowledge signal back to the transmit side after transmitting 8 bits. If the acknowledge signal is sent back, the transmit side continues processing considering that reception has been performed correctly. If the slave does not send back the acknowledge signal, the reception has not been performed correctly. Therefore, the master outputs a stop condition or restart condition and aborts the transmission. If the acknowledge signal is not sent back, the following two factors can be considered.

- <1> The receptions has not been performed correctly.
- <2> The final data has been received.

When the receive side sets the SDA line to low level at the ninth clock, the acknowledge signal ( $\overline{\text{ACK}}$ ) becomes active (normal reception response).

When ACKE of the IICC register = 1, the acknowledge signal automatic generation enable status is set. The TRC bit of the IICS register is set according to the data of the eighth bit following the 7-bit address information. However, when the value of the TRC bit is 0, set ACKE = 1, because it is receive status. In the slave receive operation (TRC = 0), if the slave side has received two or more bytes and needs the next data, set ACKE = 0 so that the master side is disabled to start the next transfer. Similarly, in the master receive operation (TRC = 0), if the master side receives two or more bytes and needs the next data, set ACKE = 0 to output the restart condition or the stop condition so that the  $\overline{\text{ACK}}$  signal does not generate. This prevents the MSB data from being output to the SDA line in the slave transmit operation (transmission stops).

**Figure 8-18. Acknowledge Signal**



When receiving the local address, the acknowledge signal is automatically output in synchronization with the fall of the eighth clock of SCL regardless of the value of the ACKE. The acknowledge signal is not output when receiving an address other than the local address.

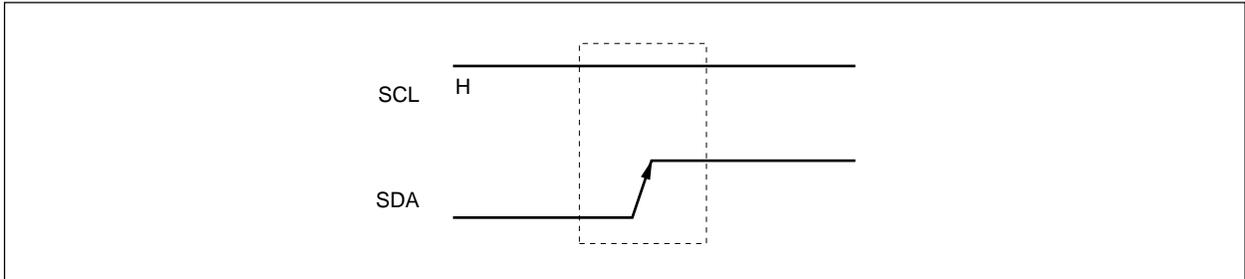
The output methods of the acknowledge signal are as follows according to the setting of the wait timing.

- When selecting 8-clock wait : acknowledge signal is output by setting ACKE = 1 before releasing wait.
- When selecting 9-clock wait : acknowledge signal is automatically output in synchronization with the fall of the eighth clock of SCL by setting ACKE = 1 beforehand.

**(5) Stop condition**

The stop condition is generated when the SDA pin changes from low level to high level while the SCL pin is high level.

The stop condition is a signal which the master outputs to the slave when a serial transfer has ended. The slave is provided with the hardware to detect the stop condition.

**Figure 8-19. Stop Condition**

The stop condition is generated by setting (to 1) bit 0 (SPT) of the IICC control register (IICC). When the stop condition is detected, bit 0 (SPD) of the IICC status register (IICS) is set (1). When bit 4 (SPIE) of IICC is set (1), INTIIC is generated.

**(6) Wait signal ( $\overline{\text{WAIT}}$ )**

The wait signal is a signal by which the master or the slave informs the other that it is preparing for transmitting/receiving data (wait status).

The master of the slave informs the wait status to the other by inputting the low level to the SCL pin. Both the master and slave can start the next transfer when the wait status is released.

**Figure 8-20. Wait Signal (1/2)**

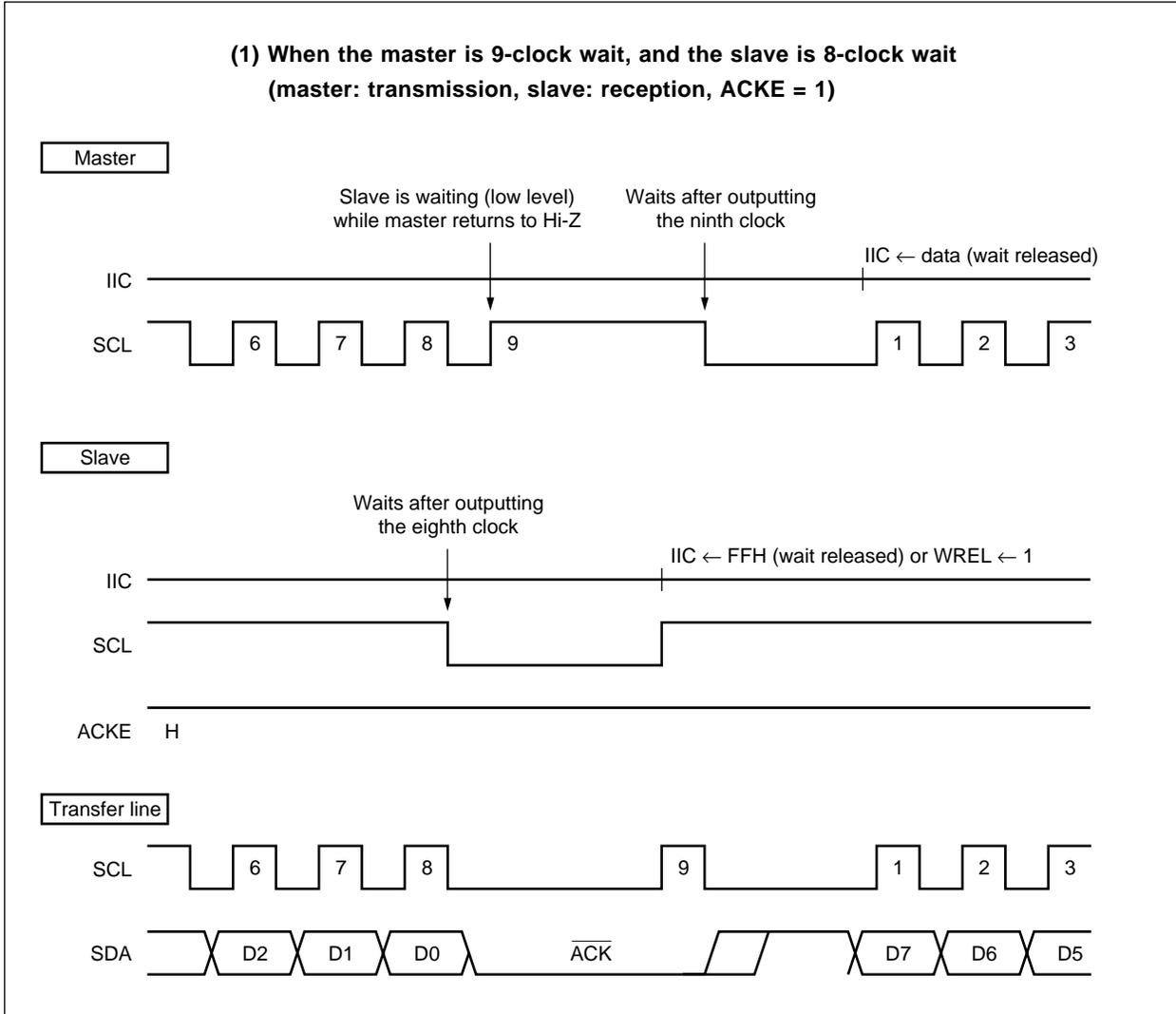
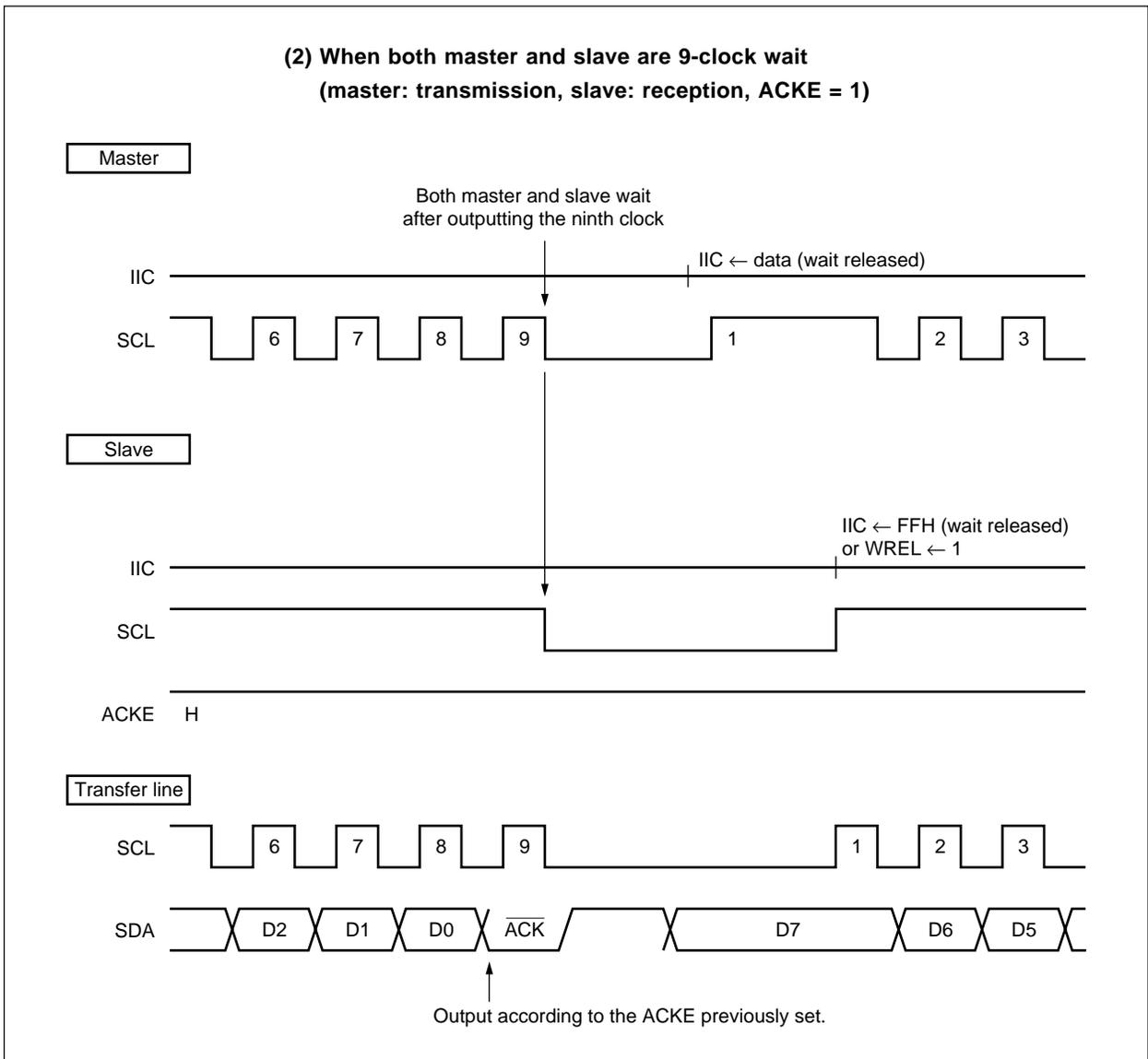


Figure 8-20. Wait Signal (2/2)



The wait of the IICC register automatically generates according to the setting of WTIM.

Normally, the receive side releases wait when WREL = 1 or when FFH is written to IIC, and the transmit side releases wait when data is written to IIC.

For the master, wait can be released also with the following method.

- STT = 1
- SPT = 1

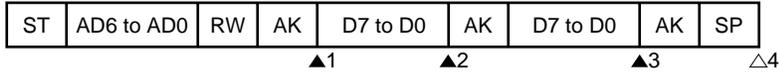
**(7) I<sup>2</sup>C interrupt (INTIIC)**

The following shows the INTIIC interrupt request generation timing and the value of the IIC status register (IICS) in INTIIC interrupt timing.

**(a) Master operation**

**(i) Start-Address-Data-Data-Stop (normal transmission/reception)**

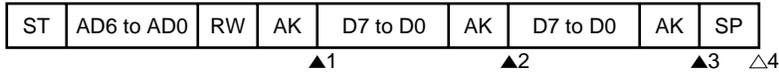
**<1> When WTIM = 0**



- ▲ 1: IICS = 10xxx110 B
- ▲ 2: IICS = 10xxx000 B
- ▲ 3: IICS = 10xxx000 B
- △ 4: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

**<2> When WTIM = 1**

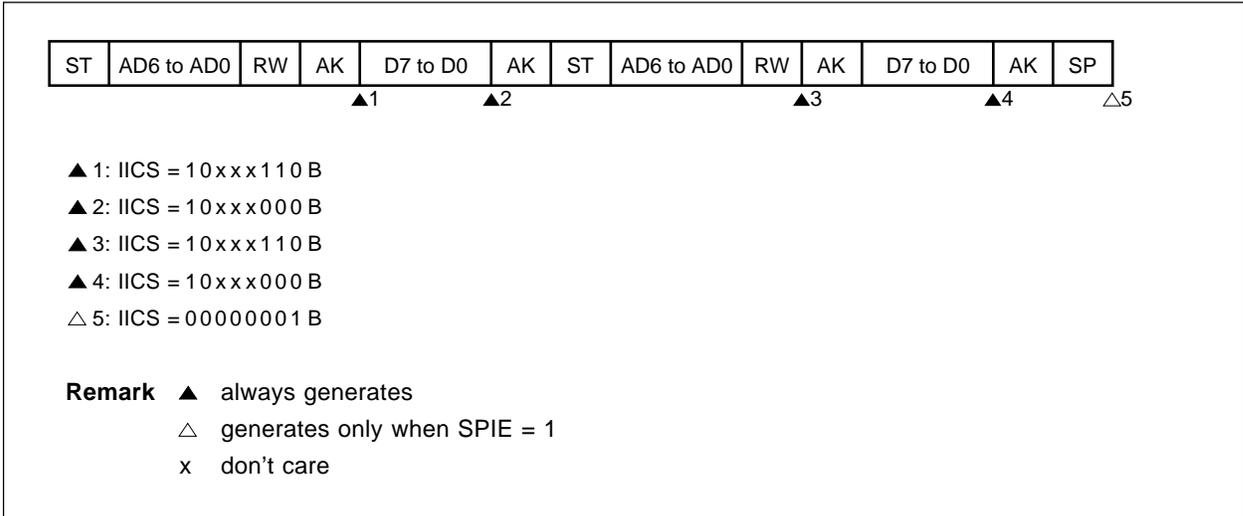


- ▲ 1: IICS = 10xxx110 B
- ▲ 2: IICS = 10xxx100 B
- ▲ 3: IICS = 10xxx000 B
- △ 4: IICS = 00000001 B

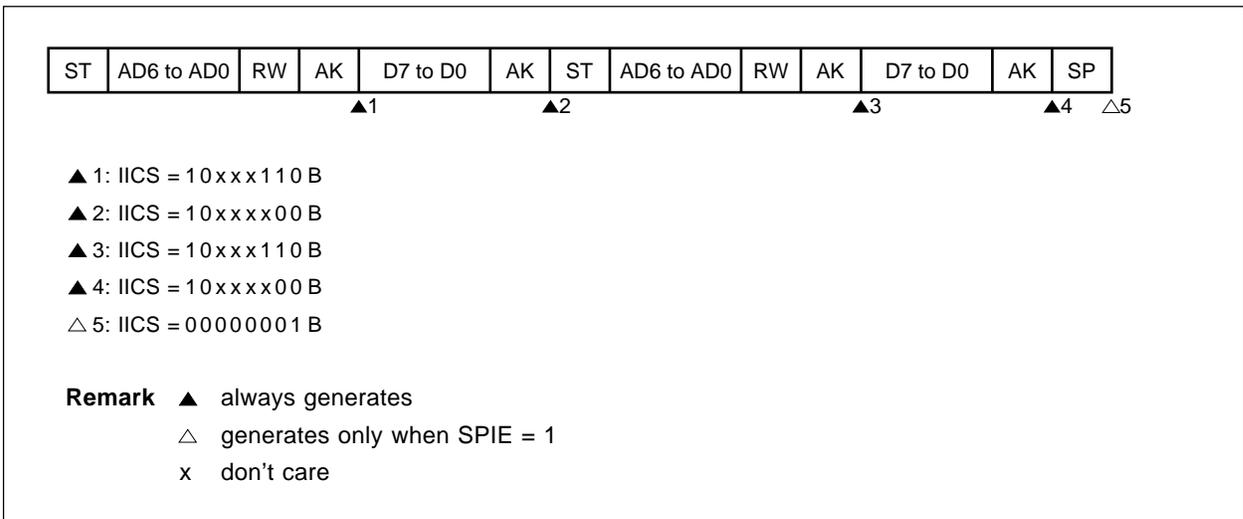
**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

(ii) Start-Address-Data-Start-Address-Data-Stop (restart)

<1> When WTIM = 0

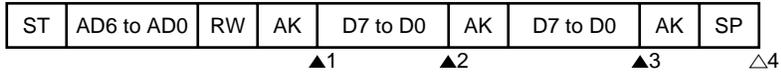


<2> When WTIM = 1



(iii) Start-Code-Data-Data-Stop (extension code transmission)

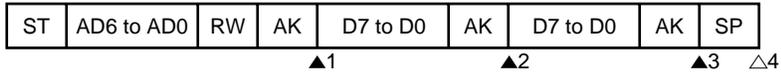
<1> When WTIM = 0



- ▲ 1: IICS = 1010x110 B
- ▲ 2: IICS = 1010x000 B
- ▲ 3: IICS = 1010x000 B
- △ 4: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

<2> When WTIM = 1



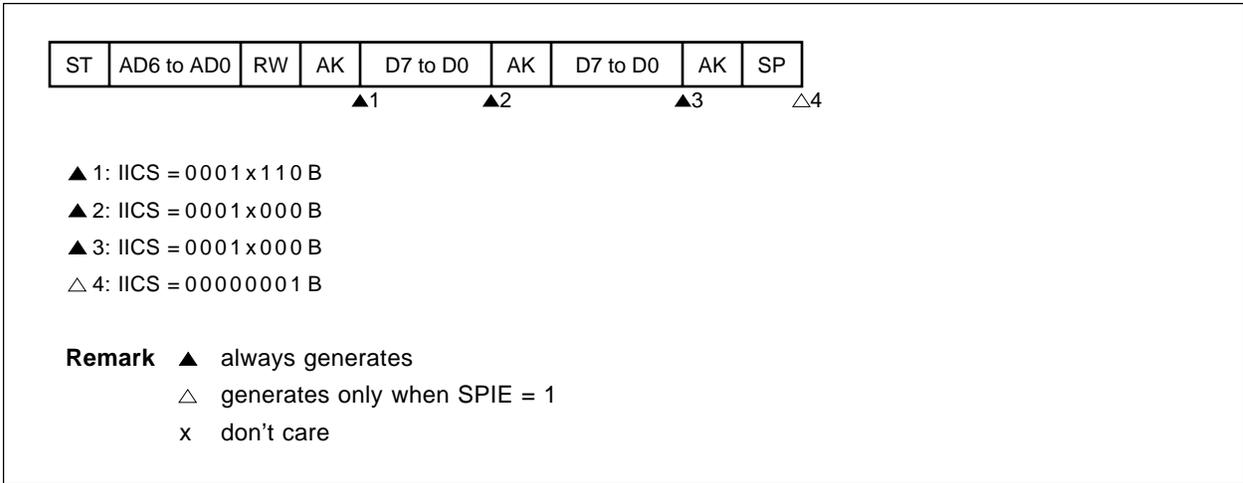
- ▲ 1: IICS = 1010x110 B
- ▲ 2: IICS = 1010x100 B
- ▲ 3: IICS = 1010xx00 B
- △ 4: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

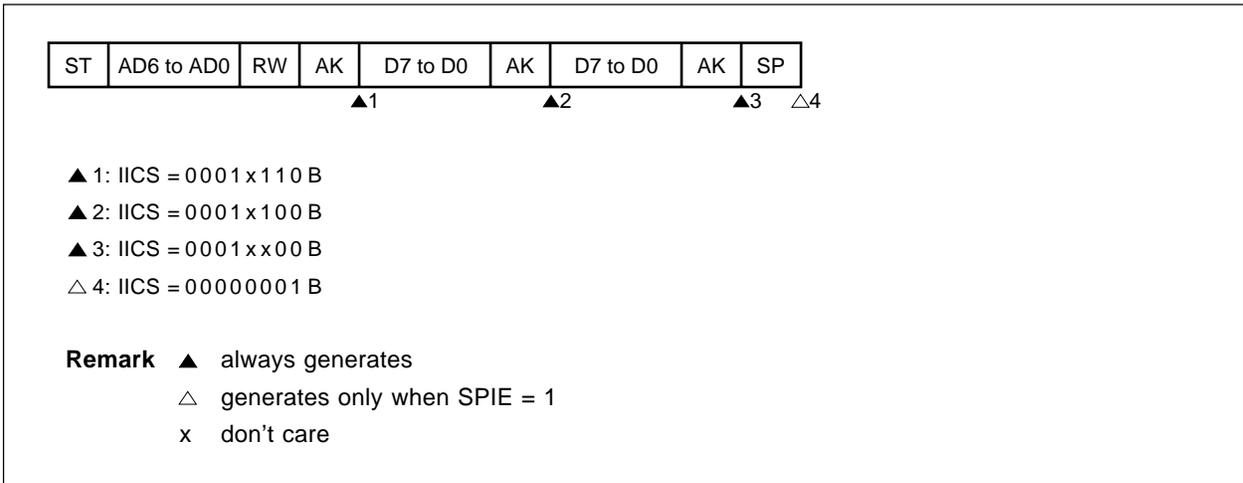
(b) Slave operation (when receiving slave address data (SVA coincides))

(i) Start-Address-Data-Data-Stop

<1> When WTIM = 0



<2> When WTIM = 1



(ii) Start-Address-Data-Start-Address-Data-Stop

<1> When WTIM = 0 (SVA coincides after restart)



▲ 1: IICS = 0001x110 B

▲ 2: IICS = 0001x000 B

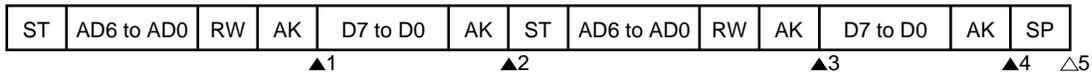
▲ 3: IICS = 0001x110 B

▲ 4: IICS = 0001x000 B

△ 5: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

<2> When WTIM = 1 (SVA coincides after restart)



▲ 1: IICS = 0001x110 B

▲ 2: IICS = 0001xx00 B

▲ 3: IICS = 0001x110 B

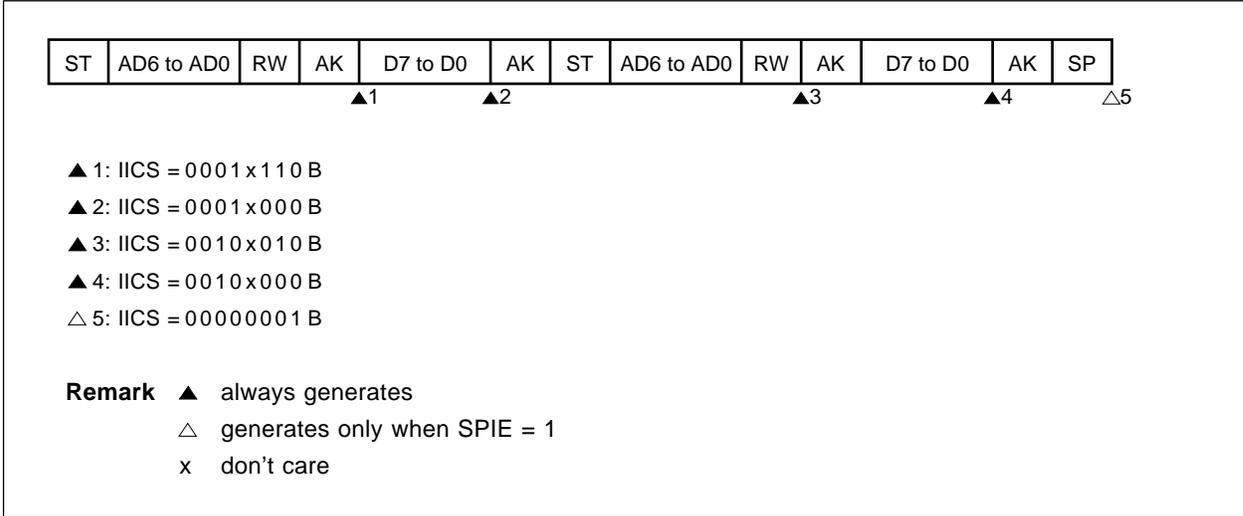
▲ 4: IICS = 0001xx00 B

△ 5: IICS = 00000001 B

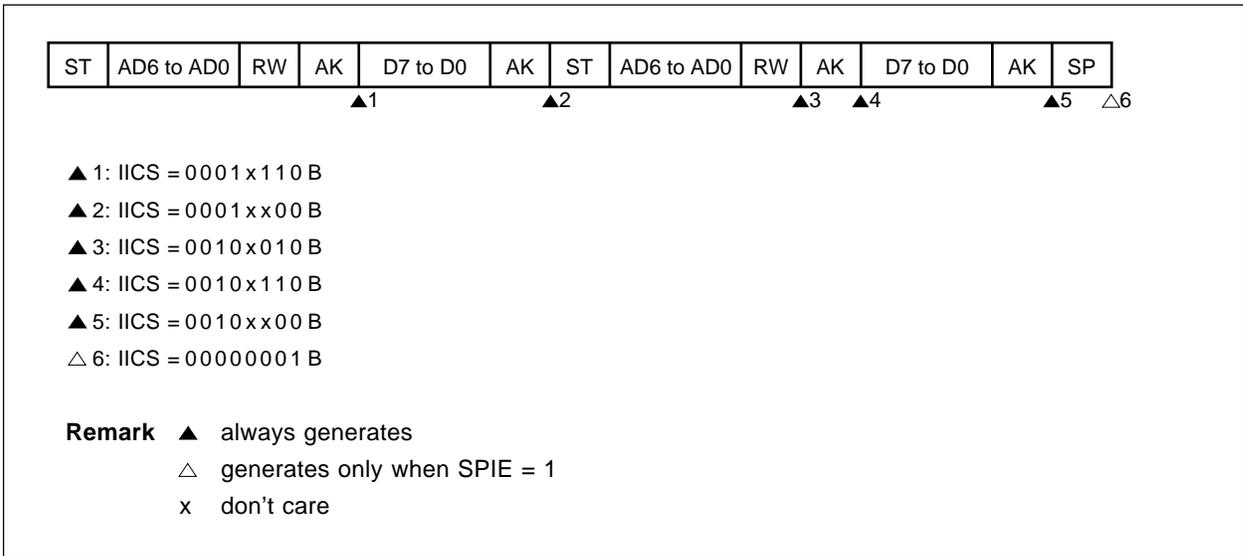
**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

(iii) Start-Address-Data-Start-Code-Data-Stop

<1> When WTIM = 0 (receives extension code after restart)



<2> When WTIM = 1 (receives extension code after restart)



(iv) Start-Address-Data-Start-Address-Data-Stop

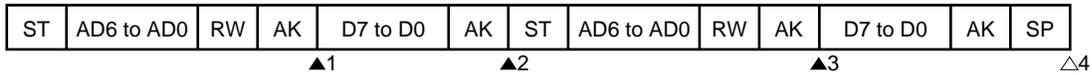
<1> When WTIM = 0 (address does not coincide after restart (except extension code))



- ▲ 1: IICS = 0001x110 B
- ▲ 2: IICS = 0001xx00 B
- ▲ 3: IICS = 0000xx10 B
- △ 4: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

<2> When WTIM = 1 (address does not coincide after restart (except extension code))



- ▲ 1: IICS = 0001x110 B
- ▲ 2: IICS = 0001xx00 B
- ▲ 3: IICS = 0000xx10 B
- △ 4: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

(c) Slave operation (when receiving extension code)

(i) Start-Code-Data-Data-Stop

<1> When WTIM = 0

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP
			▲1		▲2		▲3	△4

▲ 1: IICS = 0010x010 B  
 ▲ 2: IICS = 0010x000 B  
 ▲ 3: IICS = 0010x000 B  
 △ 4: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

<2> When WTIM = 1

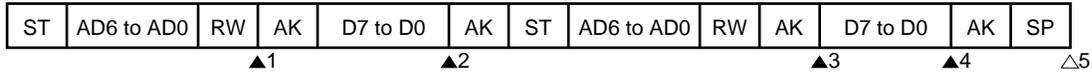
ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP
			▲1	▲2		▲3	▲4	△5

▲ 1: IICS = 0010x010 B  
 ▲ 2: IICS = 0010x110 B  
 ▲ 3: IICS = 0010xx00 B  
 ▲ 4: IICS = 0010xx00 B  
 △ 5: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

(ii) Start-Code-Data-Start-Address-Data-Stop

<1> When WTIM = 0 (SVA coincides after restart)



- ▲ 1: IICS = 0010x010 B
- ▲ 2: IICS = 0010x000 B
- ▲ 3: IICS = 0001x110 B
- ▲ 4: IICS = 0001x000 B
- △ 5: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

<2> When WTIM = 1 (SVA coincides after restart)

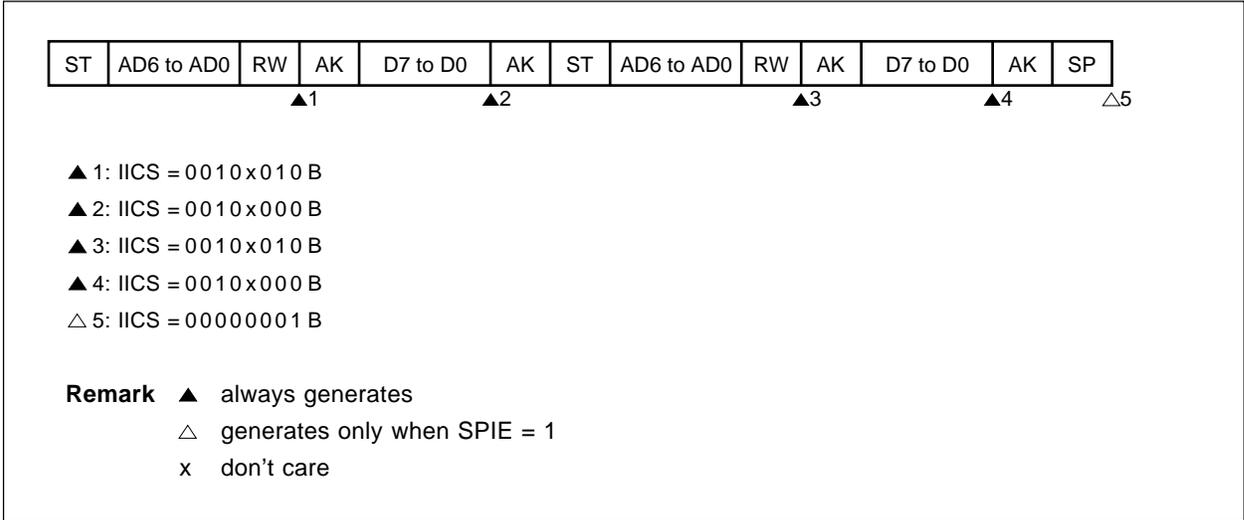


- ▲ 1: IICS = 0010x010 B
- ▲ 2: IICS = 0010x110 B
- ▲ 3: IICS = 0010xx00 B
- ▲ 4: IICS = 0001x110 B
- ▲ 5: IICS = 0001xx00 B
- △ 6: IICS = 00000001 B

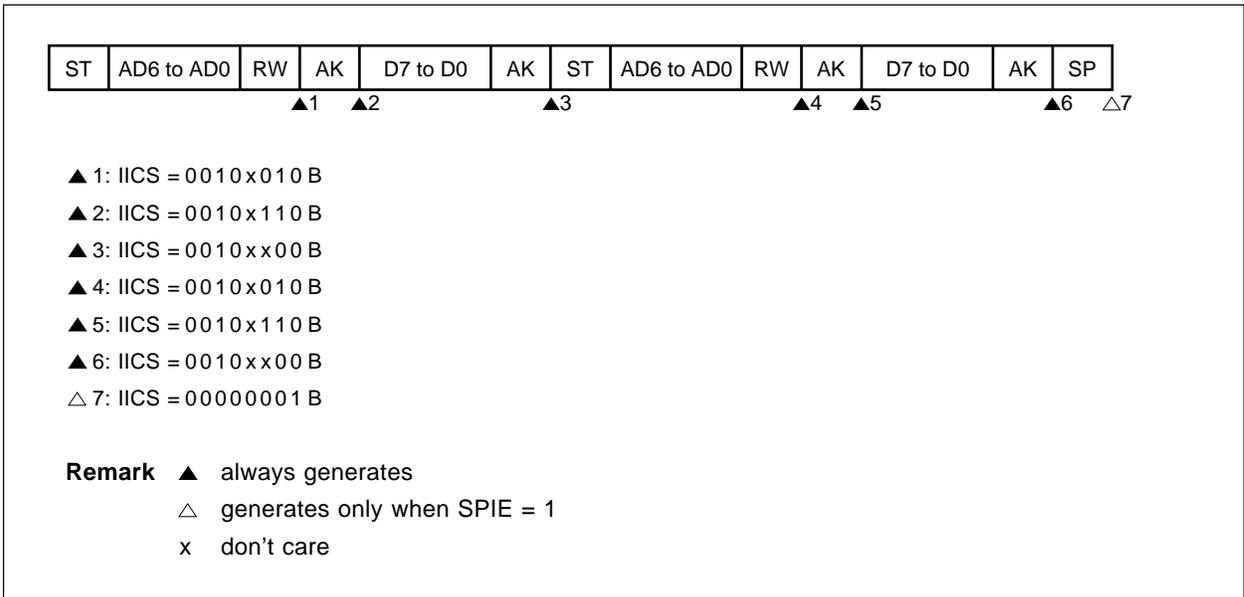
**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

(iii) Start-Code-Data-Start-Code-Data-Stop

<1> When WTIM = 0 (receives extension code after restart)



<2> When WTIM = 1 (receives extension code after restart)



(iv) Start-Code-Data-Start-Address-Data-Stop

<1> When WTIM = 0 (address does not coincides after restart (except extension code))



- ▲ 1: IICS = 0010x010 B
- ▲ 2: IICS = 0010x000 B
- ▲ 3: IICS = 00000x10 B
- △ 4: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

<2> When WTIM = 1 (address does not coincides after restart (except extension code))

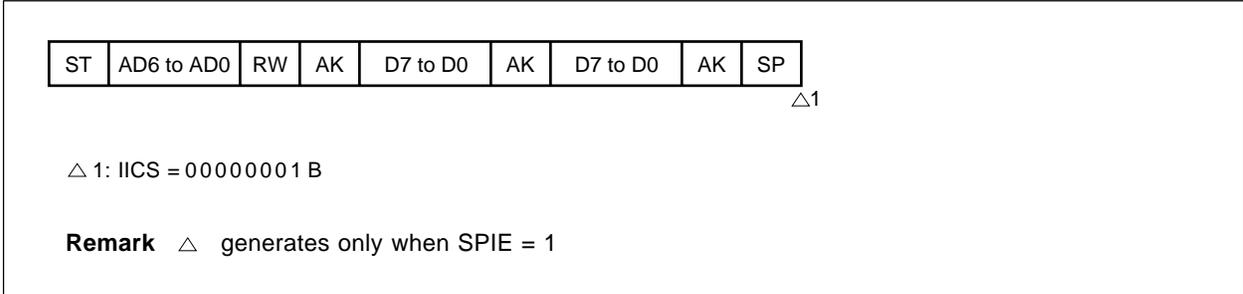


- ▲ 1: IICS = 0010x010 B
- ▲ 2: IICS = 0010x110 B
- ▲ 3: IICS = 0010xx00 B
- ▲ 4: IICS = 00000x10 B
- △ 5: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

(d) Operation of not joining communication

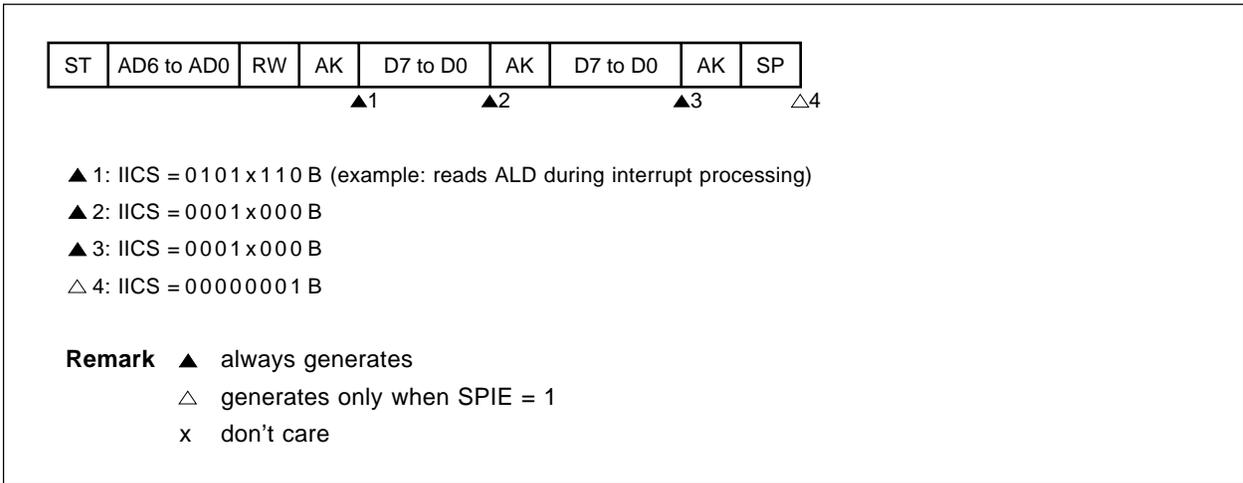
(i) Start-Code-Data-Data-Stop



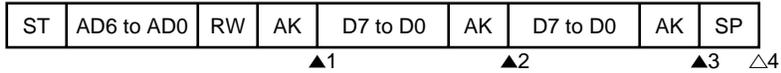
(e) Operation of arbitration defeat (operates as a slave after arbitration defeat)

(i) When defeated in arbitration during slave address data transmission

<1> When WTIM = 0



<2> When WTIM = 1



▲ 1: IICS = 0101x110 B (example: reads ALD during interrupt processing)

▲ 2: IICS = 0001x100 B

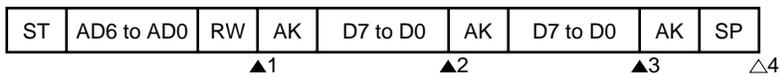
▲ 3: IICS = 0001xx00 B

△ 4: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

(ii) When defeated in arbitration during transmitting extension code

<1> When WTIM = 0



▲ 1: IICS = 0110x010 B (example: reads ALD during interrupt processing)

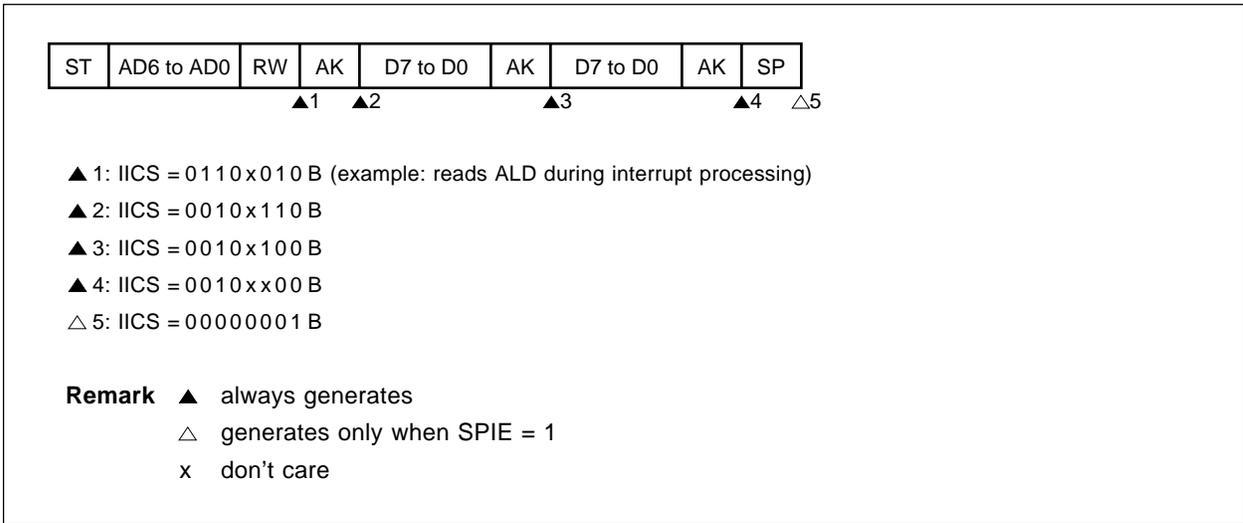
▲ 2: IICS = 0010x000 B

▲ 3: IICS = 0010x000 B

△ 4: IICS = 00000001 B

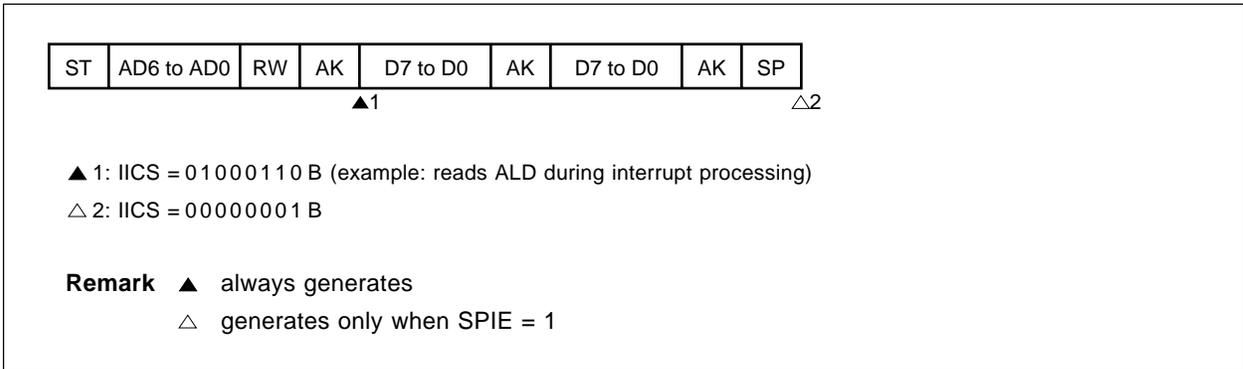
**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

<2> When WTIM = 1

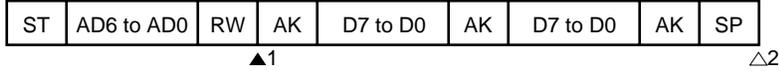


(f) Operation of arbitration defeat (does not join after arbitration defeat)

(i) When defeated in arbitration during transmitting slave address data



(ii) When defeated in arbitration during transmitting extension code



▲ 1: IICS = 01110x010 B (example: reads ALD during interrupt processing)

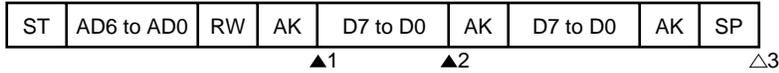
Set LREL =1 by software (when not joining communication)

△ 2: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care

(iii) When defeated in arbitration during transferring data

<1> When WTIM = 0



▲ 1: IICS = 10001110 B

▲ 2: IICS = 01000000 B (example: reads ALD during interrupt processing)

△ 3: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1

<2> When WTIM = 1

ST	AD6 to AD0	RW	AK	D7 to D0	AK	D7 to D0	AK	SP
			▲1		▲2			△3

▲ 1: IICS = 10001110 B  
 ▲ 2: IICS = 01000100 B (example: reads ALD during interrupt processing)  
 △ 3: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1

(iv) When defeated in restart condition during transferring data

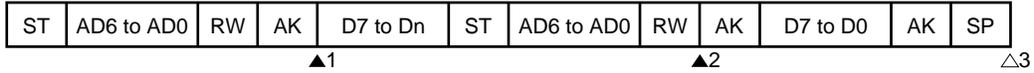
<1> Except extension code (example: SVA does not coincide)

ST	AD6 to AD0	RW	AK	D7 to Dn	ST	AD6 to AD0	RW	AK	D7 to D0	AK	SP
			▲1					▲2			△3

▲ 1: IICS = 1000x110 B  
 ▲ 2: IICS = 01000110 B (example: reads ALD during interrupt processing)  
 △ 3: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care  
 n = 0 to 6

<2> Extension code



▲ 1: IICS = 1000x110 B

▲ 2: IICS = 0110x010 B (example: reads ALD during interrupt processing)

Set IICC : LREL = 1 by software (when not joining communication)

△ 3: IICS = 00000001 B

**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care  
 n = 0 to 6

(v) When defeated in stop condition during transferring data



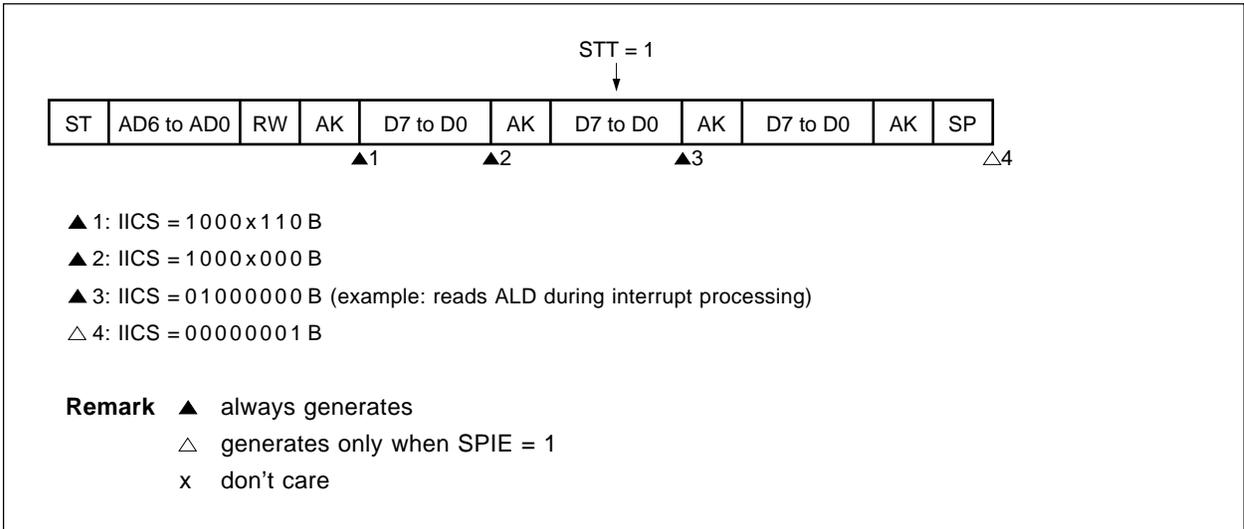
▲ 1: IICS = 1000x110 B

△ 2: IICS = 01000001 B

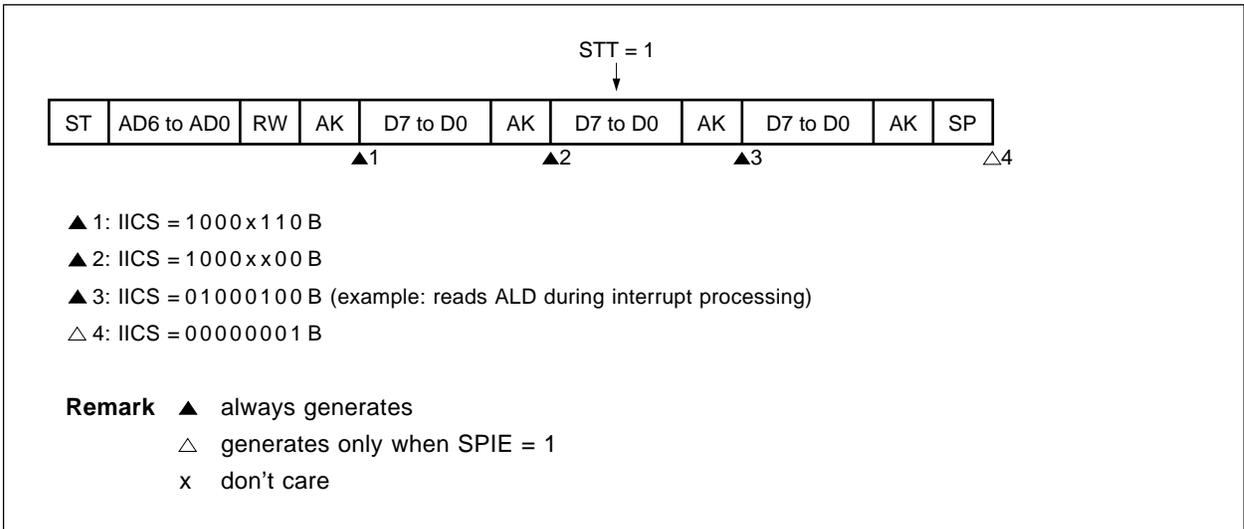
**Remark** ▲ always generates  
 △ generates only when SPIE = 1  
 x don't care  
 n = 0 to 6

(vi) When attempting to generate restart condition and defeated in arbitration because the data is low level

<1> When WTIM = 0

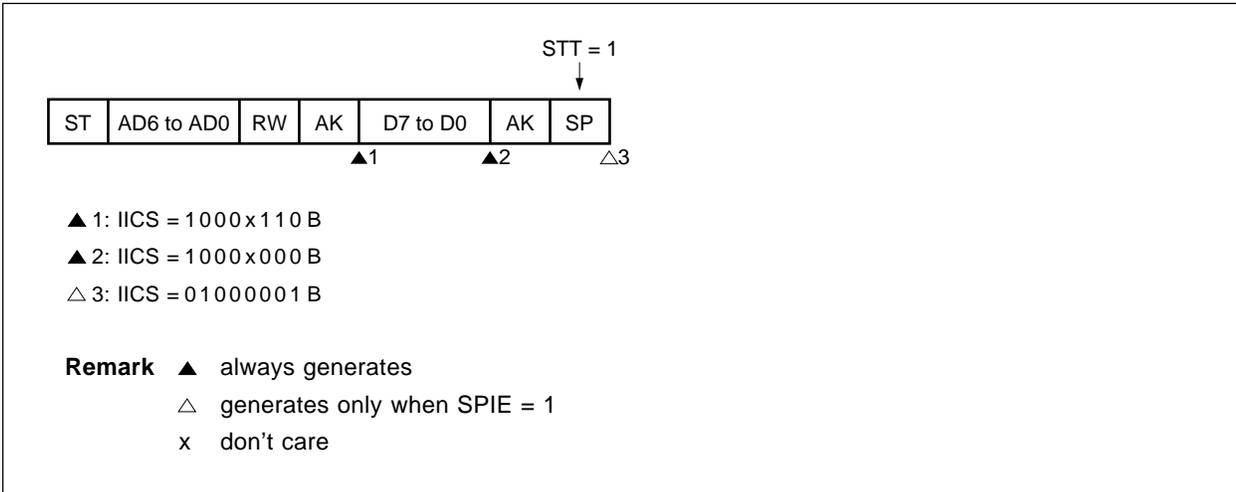


<2> When WTIM = 1

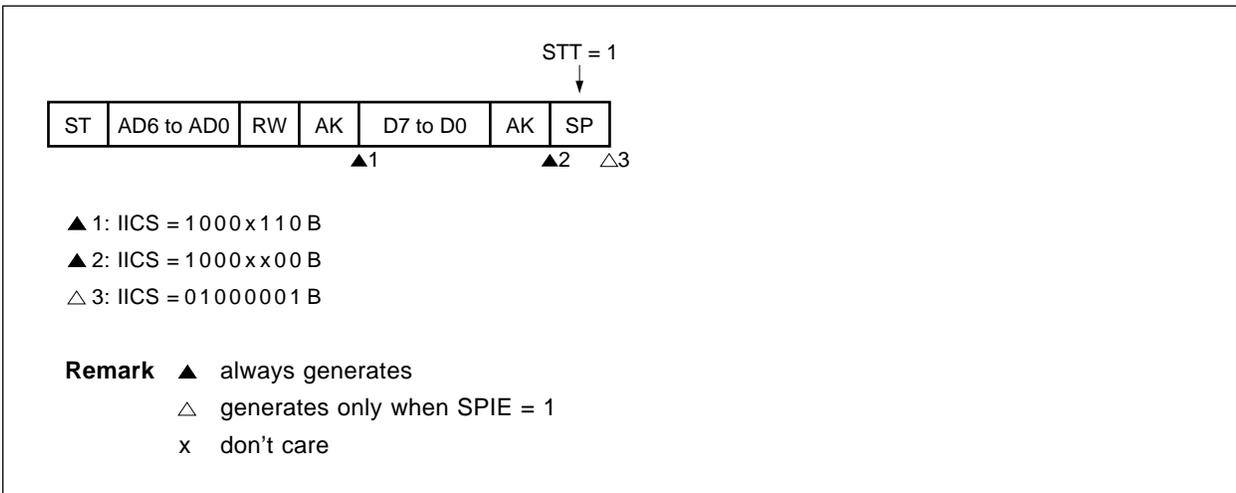


(vii) When attempting to generate restart condition and defeated in arbitration at stop condition because the data is low level

<1> When WTIM = 0

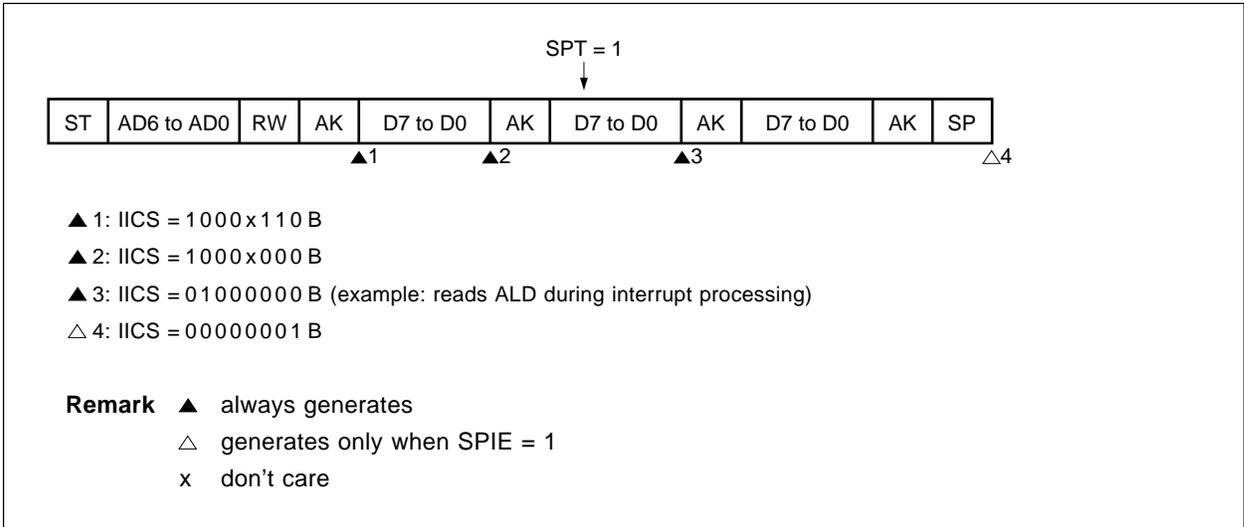


<2> When WTIM = 1

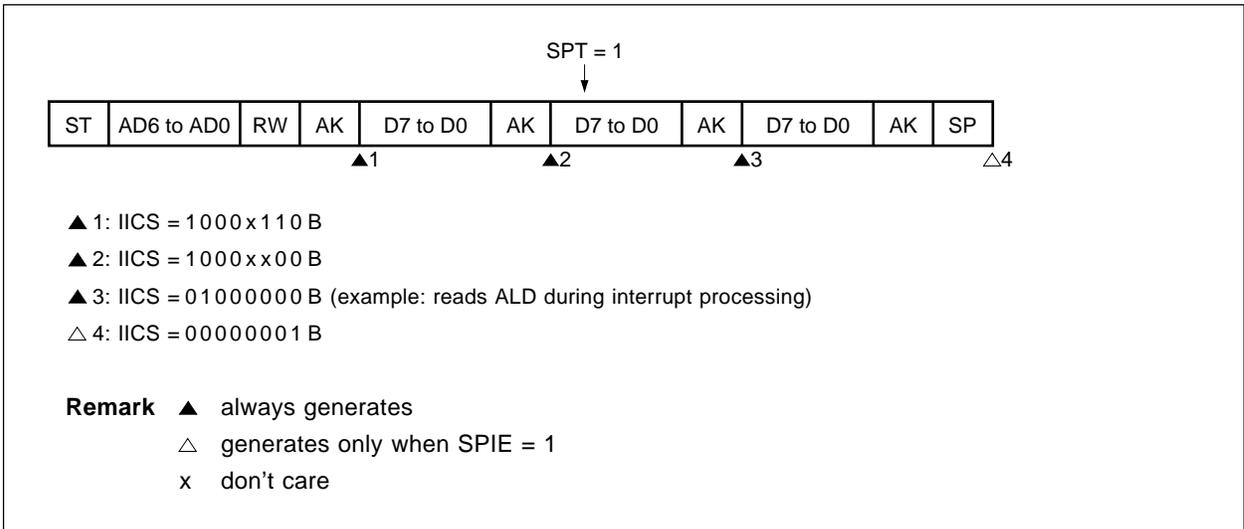


(viii) When attempting to generate stop condition and defeated in arbitration because the data is low level

<1> When WTIM = 0



<2> When WTIM = 1



**(8) Interrupt request (INTIIC) generation timing and wait control**

INTIIC generates and wait control is performed by the settings of the WTIM bit of the IIC control register (IICC) at the timings shown in Table 8-3.

**Table 8-3. INTIIC Generation Timing and Wait Control**

WTIM	In Slave Operation			In Master Operation		
	Address	Data Reception	Data Transmission	Address	Data Reception	Data Transmission
0	g <sup>Notes 1, 2</sup>	g <sup>Note 2</sup>	g <sup>Note 2</sup>	9	8	8
1	g <sup>Notes 1, 2</sup>	g <sup>Note 2</sup>	g <sup>Note 2</sup>	9	9	9

- Notes**
1. INTIIC signal and wait of the slave generate at the fall of the ninth clock only when they coincide with the address set in the slave address register (SVA).  
At this time,  $\overline{ACK}$  is output regardless of the setting of ACKE. The slave which has received an extension code generates INTIIC at the fall of the eighth clock.
  2. Neither INTIIC nor wait generates when SVA and the received address do not coincide.

**Remark** The numbers in the table represent the number of the clocks of the serial clock. Both the interrupt request and the wait control synchronize with the fall of the serial clock.

**(a) When transmitting/receiving address**

- In slave operation : The interrupt and the wait timings are determined regardless of the WTIM bit.
- In master operation : The interrupt and the wait timings generates at the fall of the ninth clock regardless of the WTIM bit.

**(b) When receiving data**

- In master/slave operation: The interrupt and the wait timings are determined by the WTIM bit.

**(c) When transmitting data**

- In master/slave operation: The interrupt and the wait timings are determined by the WTIM bit.

**(d) Wait release**

The following four methods are available for releasing wait.

- Set WREL of the control register (IICC) = 1
- Write operation of IIC shift register (IIC)
- Start condition (STT) set
- Stop condition (SPT) set

When 8-clock wait (WTIM = 0) is selected, the output level of  $\overline{ACK}$  should be determined before wait release.

**(e) Stop condition detection**

INTIIC generates when stop condition is detected.

**(9) Address coincidence detection**

The I<sup>2</sup>C bus can select the specific slave device when the master transmits the slave address.

Address coincidence detection can be automatically performed by hardware. If the local address is set to the slave address register (SVA), INTIIC interrupt request generates only when the slave address transmitted from the master and the address set in SVA coincide or when an extension code is received.

**(10) Error detection**

The I<sup>2</sup>C bus can detect transmission errors by comparing the IIC data before transmission starts and that after transmission ends because the status of the serial bus during transmission (SDA) is captured also to IIC shift register (IIC) of the transmitting device. In this case, if the two data differ, it is judged that a transmission error has occurred.

**(11) Extension code**

(a) Interrupt request (INTIIC) is issued at the fall of the eighth clock by setting the extension code reception flag (EXC) regarding it as reception of exception code when the higher 4 bits of the reception address is either “0000” or “1111”.

The local address stored in the slave address register (SVA) is not affected.

(b) The following is set when “111110xx” is set to SVA by 10-bit address transfer and “111110xx0” is transferred from the master. However, interrupt request (INTIIC) generates at the fall of the eighth clock.

- Coincidence of the higher 4-bit data : EXC = 1
- Coincidence of the 7-bit data : COI = 1

(c) The processing after interrupt request is issued differs depending on the data following the extension code. Therefore, it is performed by software.

For example, LREL is set to 1 and the next communication wait status enters not to operate a device as a slave after receiving extension code.

**Table 8-4. Definition of Extension Code Bit**

Slave Address	R/W Bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	x	CBUS address
0000 010	x	Address reserved for different bus format
1111 0xx	x	10-bit slave address specification

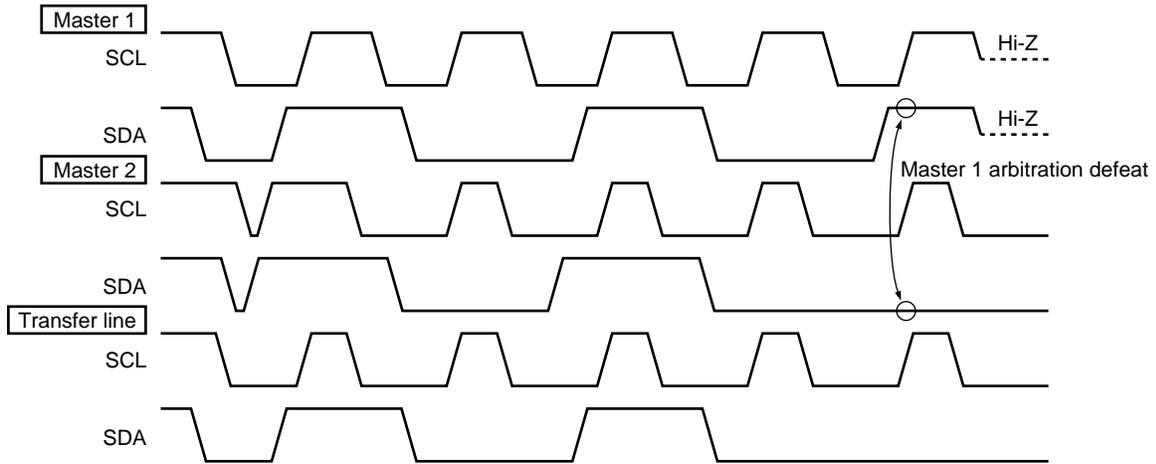
**(12) Arbitration**

When more than one master simultaneously output start conditions (when setting STT =1 before STD = 1 is set), master communication continues until the data differs while adjusting the clock. This operation is called arbitration.

The master which is defeated in arbitration sets the ALD bit of the IIC status register (IICS) at the timing at which the master is defeated in the arbitration, sets both SCL and SDA lines in Hi-Z status, and releases bus. The arbitration defeat is detected by software when ALD =1 at the next interrupt request generation timing (the eighth or ninth clock, stop condition detection, etc.).

For the interrupt generation timing, refer to **(7) I<sup>2</sup>C interrupt (INTIIC)**.

Figure 8-21. Example of Arbitration Timing



Status when Arbitration Occurs	Interrupt Request Issue Timing
Address transmitting	Fall of the eighth or ninth clock after byte transfer <sup>Note 1</sup>
Read/write information after address transmission	
Transferring extension code	
Read/write information after extension code transmission	
Transmitting data	
ACK transfer period after data transmission	
Transferring data, restart condition detection	
Transferring data, stop condition detection	Stop condition output (SPIE = 1) <sup>Note 2</sup>
Attempted to output restart condition but data is low level	Fall of the eighth or ninth clock after byte transfer <sup>Note 1</sup>
Attempted to output restart condition but stop condition is detected	Stop condition output (SPIE = 1) <sup>Note 2</sup>
Attempted to output stop condition but data is low level	Fall of the eighth or ninth clock after byte transfer
Attempted to restart condition but SCL is low level	

**Notes** 1. When WTIM (bit 3 of the IIC control register (IICC)) = 1, interrupt generates at the fall of the ninth clock. When WTIM = 0 and when slave address of extension code is received, interrupt generates at the fall of the eighth clock.

2. If arbitration may occur, set SPIE = 1 in master operation.

**(13) Wake-up function**

Wake-up function generates interrupt request (INTIIC) when the local address and an extension code are received in the slave function of I<sup>2</sup>C.

When address does not coincide, unnecessary interrupt does not generate, so that efficient processing is possible.

When start condition is detected, wake-up wait status is set. Even a master can become a slave as a result of arbitration defeat (in the case start condition is output), it enters wake-up wait status while transmitting address.

However, when a stop condition is detected, interrupt enable/disable is determined according to the setting of the SPIE bit regardless of the wake-up function.

**(14) Communication reservation**

- When a device could become neither a master or slave in arbitration.
- When a device does not operate as a slave receiving an extension code (when not sending back  $\overline{\text{ACK}}$  and releasing bus with LREL of IICC = 1).

When the STT bit of the IICC is set in the status not joining the bus, a start condition is automatically generated and wait status is entered after the bus is released (stop condition detection).

When the bus release is detected (stop condition detection), address transfer as a master is started by IIC write manipulation. At this time the SPIE bit of the IICC should be set.

When STT is set, whether it is operates as start condition or as communication reservation is determined by the bus status.

- When bus is released : start condition generation
- When bus is not released (wait status) : communication reservation

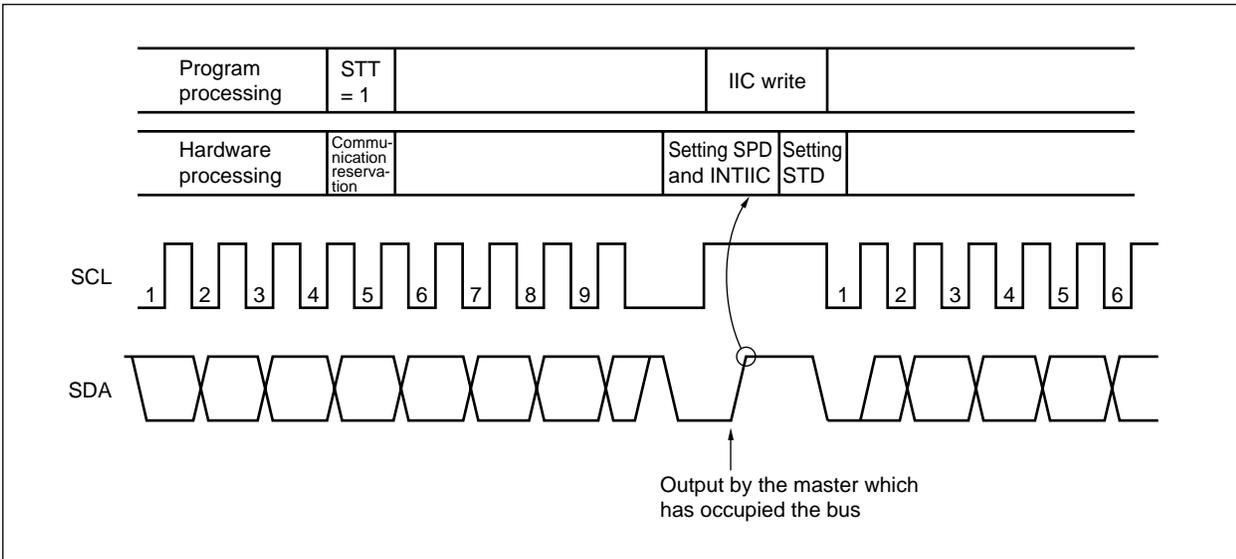
Which operation STT has performed is detected by checking the STT bit again after setting STT and giving wait time.

Secure the wait time as shown in Table 8-5 by software. Wait time can be set by SMC, CL0, and CL1 of IICCL.

**Table 8-5. Wait Time**

SMC	CL1	CL0	Wait Time
0	0	0	26 clocks
0	0	1	46 clocks
0	1	0	
0	1	1	37 clocks
1	0	0	16 clocks
1	0	1	
1	1	0	
1	1	1	13 clocks

Figure 8-22. Communication Reservation Timing



The communication reservation is acknowledged at the following timings. The communication reservation is made by setting the STT of IICC = 1 before stop condition detection after STD of IICS = 1 is set.

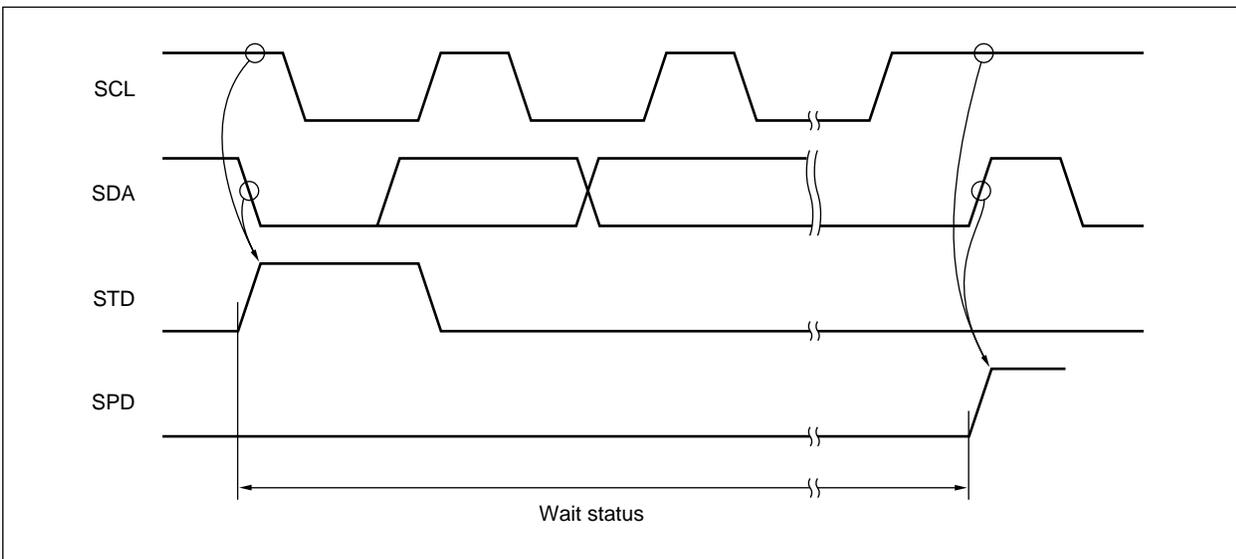
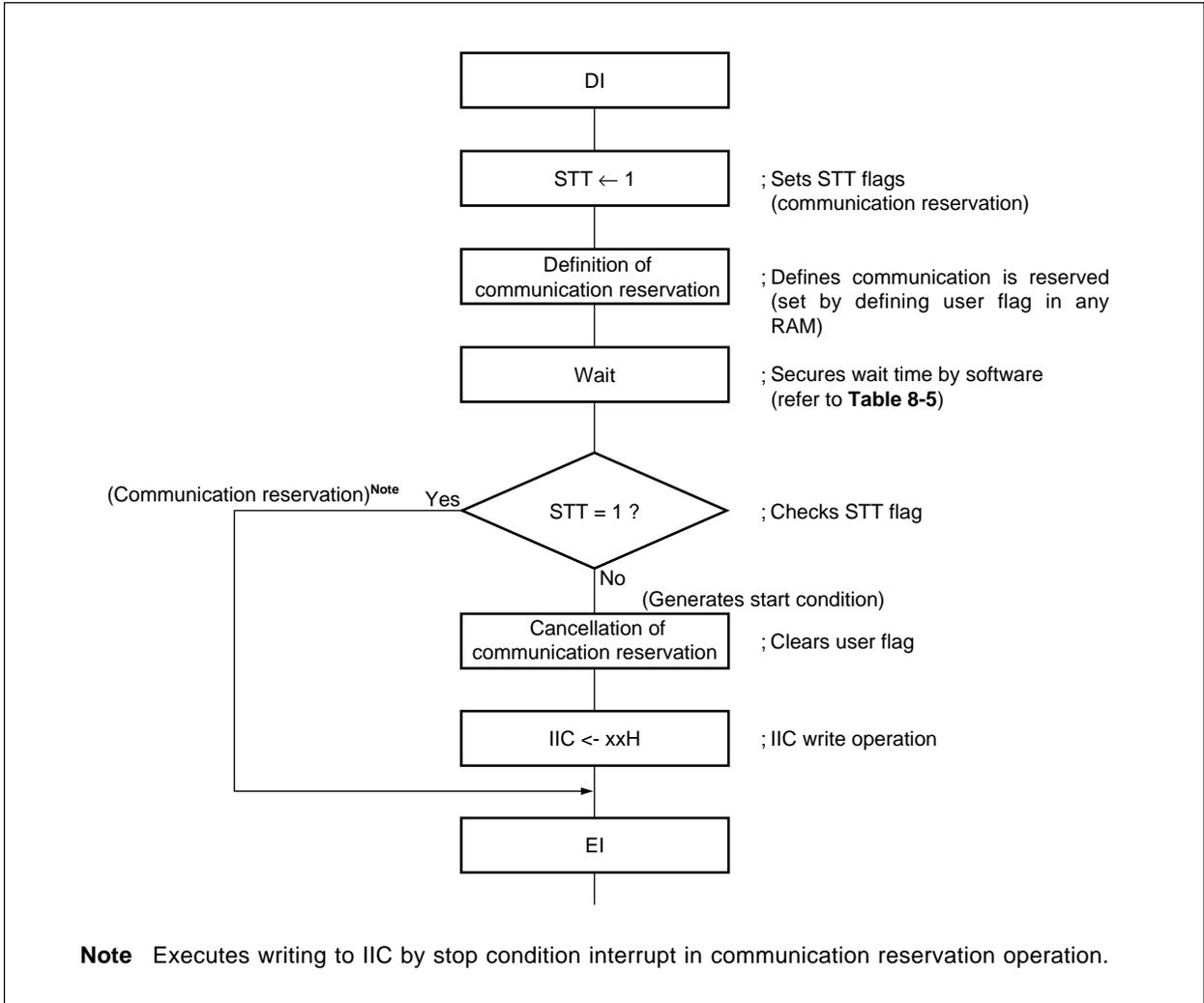


Figure 8-23 shows the communication reservation procedure.

**Figure 8-23. Communication Reservation Procedure**



**(15) Other precautions**

**(a) Multi-master communication**

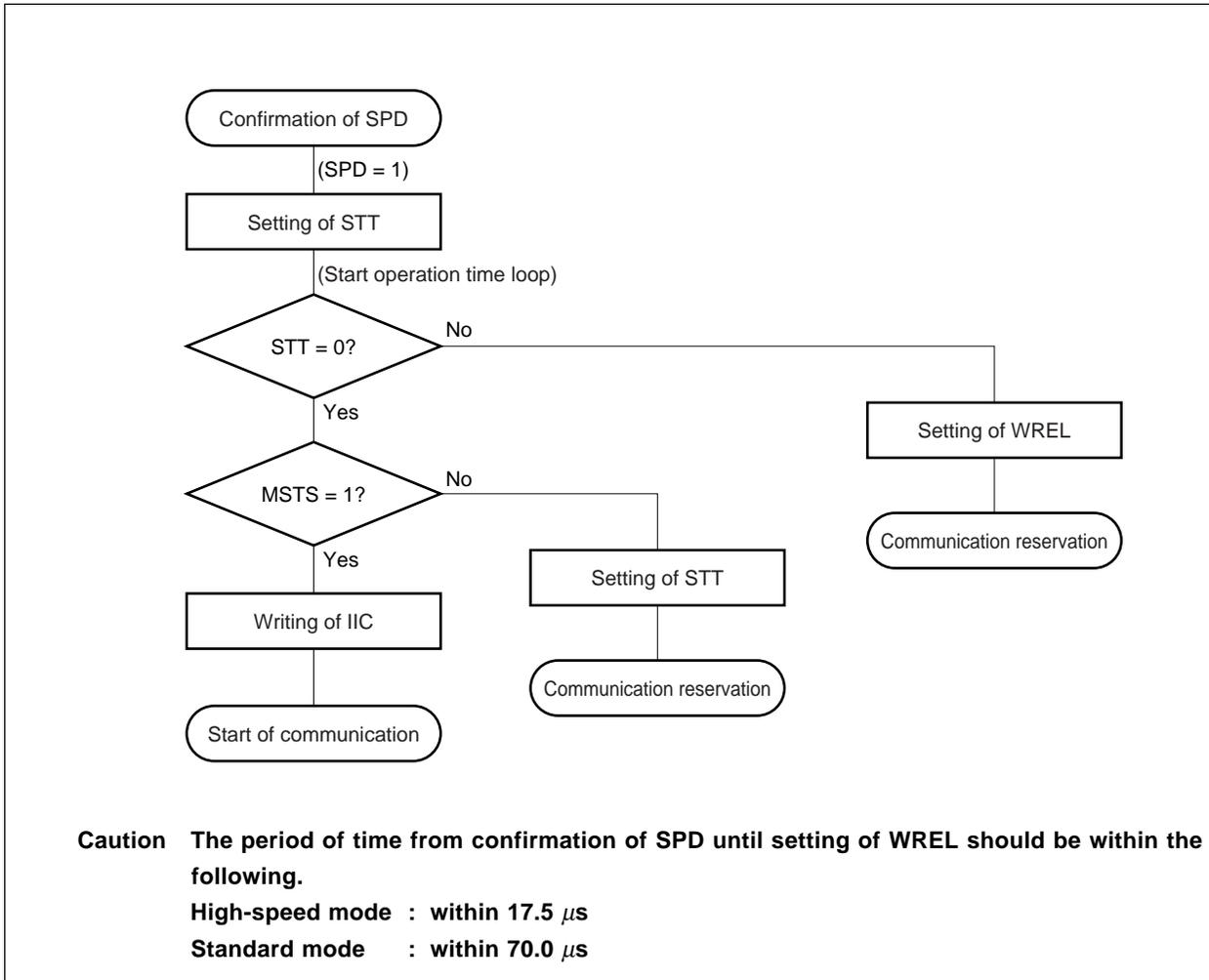
To perform master communication from the status in which stop condition is not detected (bus is not released) after reset, generate stop condition and release bus before starting master communication. In multi-master, master communication cannot be performed in the status in which bus is not released (stop condition is not detected).

Generate stop condition with the following procedure.

- <1> Setting IICCL
- <2> IICE of IICC ← 1
- <3> SPT of IICC ← 1

- ★ **(b) Operation during communication reservation (when a multi-master is used)**  
 If a slave is selected during communication reservation, when writing a “1” to the WREL bit, write a “0” to the STT bit at the same time. After that, following stop condition detection, write a “1” to the STT bit again.
- ★ **(c) Start operation after communication reservation (when a multi-master is used)**  
 If the stop condition is detected and the start condition is generated after that, after confirming that the MSTS bit is 1 (master communication state), confirm that the TRC bit is 1 (transmission operation). At this time, if the TRC bit is 0, set LREL to withdraw from communication, and carry out communication reservation (STT = 1) again after the stop condition is detected.
- ★ **(d) STT setting timing (when a multi-master is used)**  
 Arrange the program as shown in Figure 8-24.

Figure 8-24. STT Setting Timing Procedure

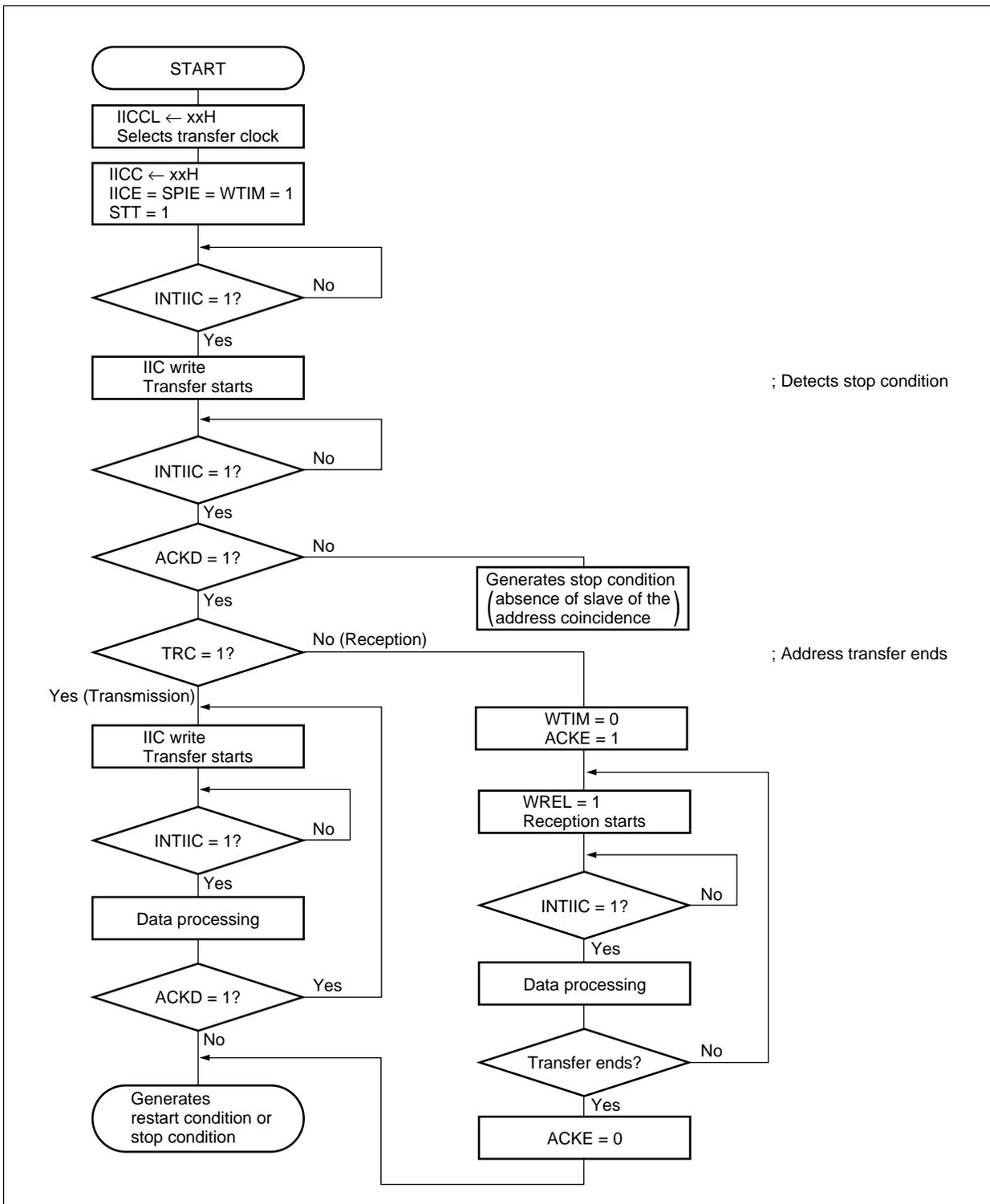


8.4.7 Communication operation

(1) Master operation

The following shows the master communication procedure.

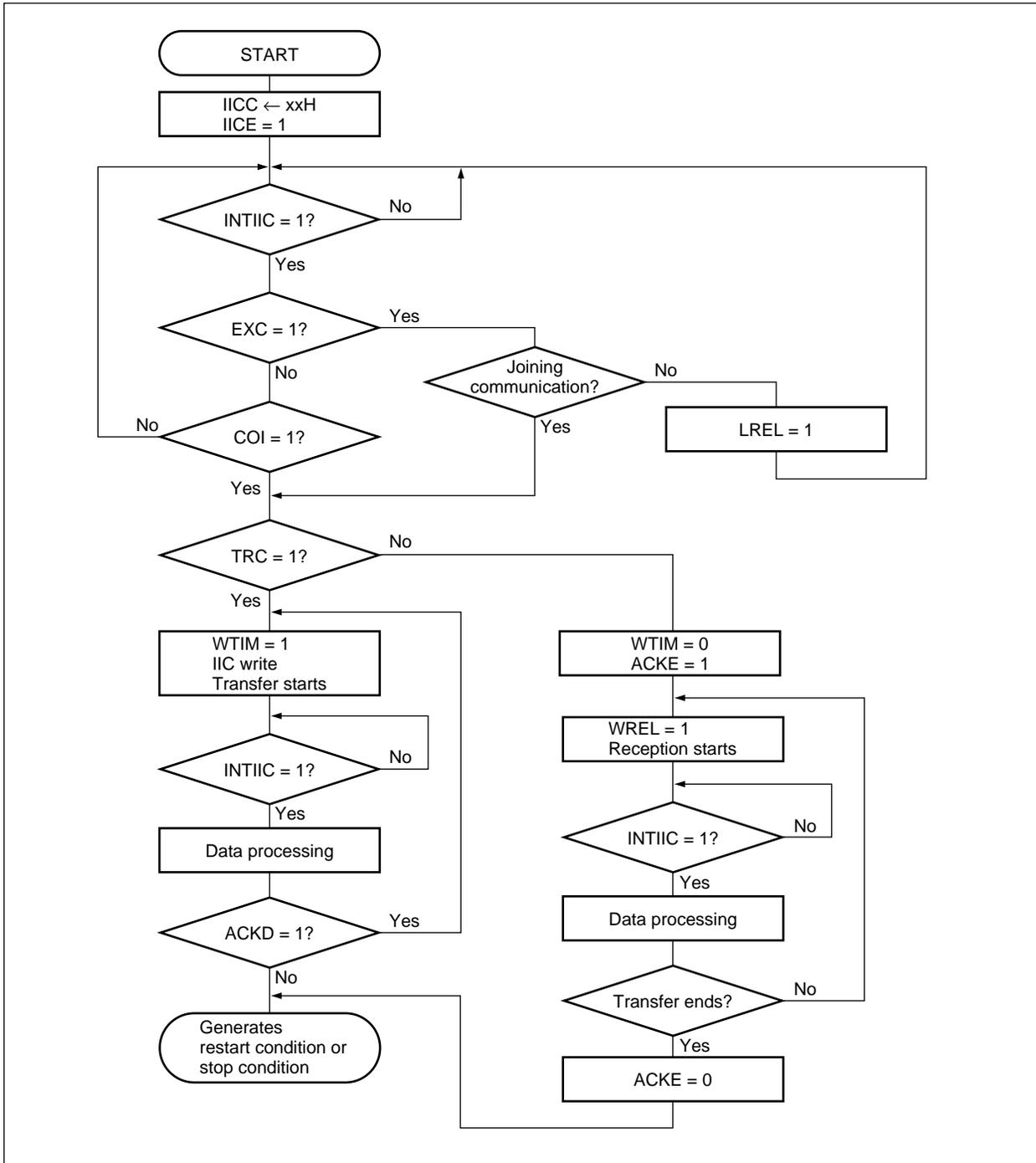
Figure 8-25. Master Operation Procedure



(2) Slave operation

The following shows the slave communication procedure.

Figure 8-26. Slave Operation Procedure



**8.4.8 Timing chart**

In the I<sup>2</sup>C bus mode, a target slave device is selected from more than one slave device when the master outputs an address to the serial bus.

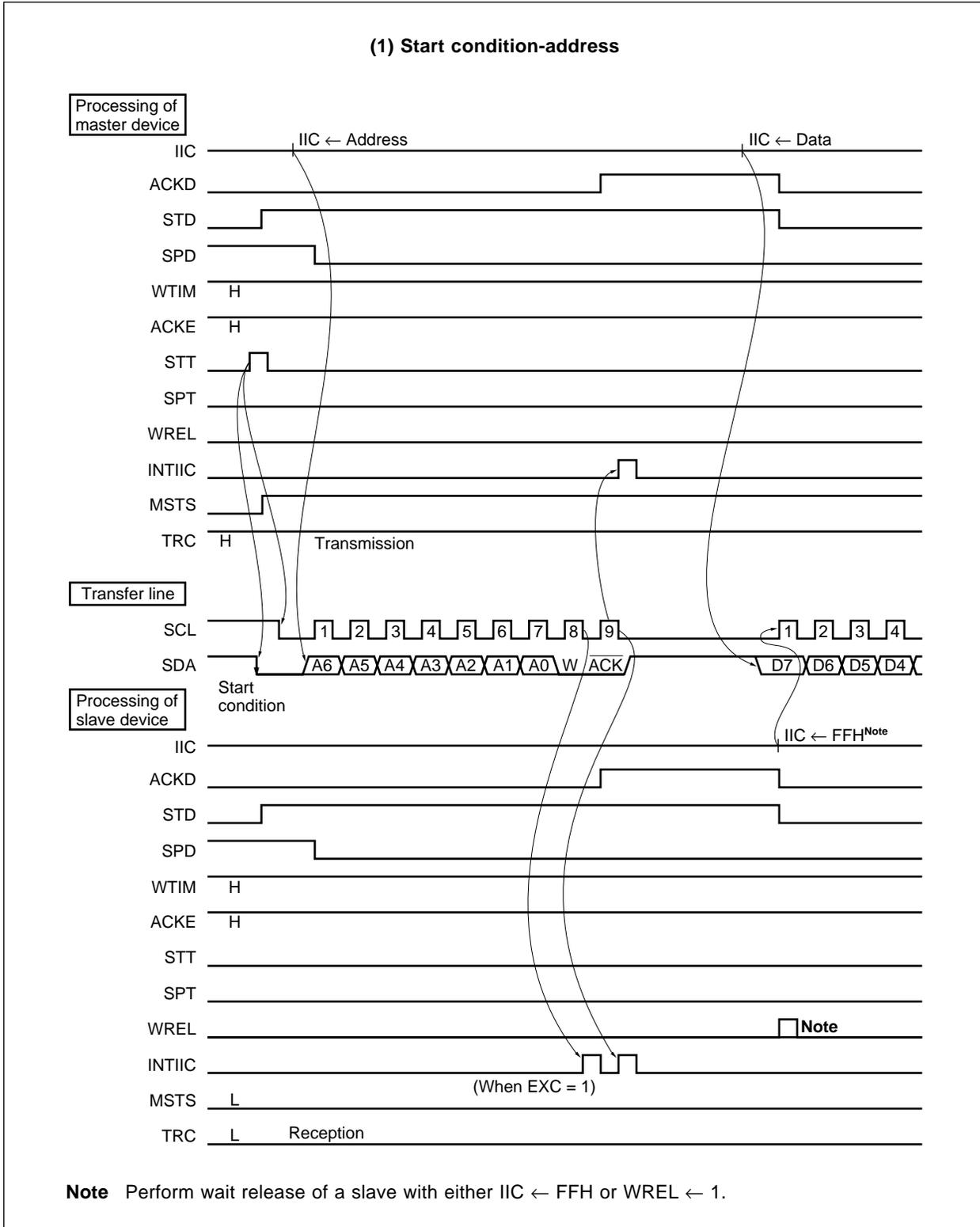
The master transmits the TRC bit that indicates the transfer direction of data following the slave address and starts serial communication with the slave.

Figures 8-27 and 8-28 show the timing charts of data communication.

The shift operation of the shift register (IIC) is performed in synchronization with the fall of the serial clock (SCK), the transmitted data is transferred to the SO latch, and output from the SDA pin with the MSB first.

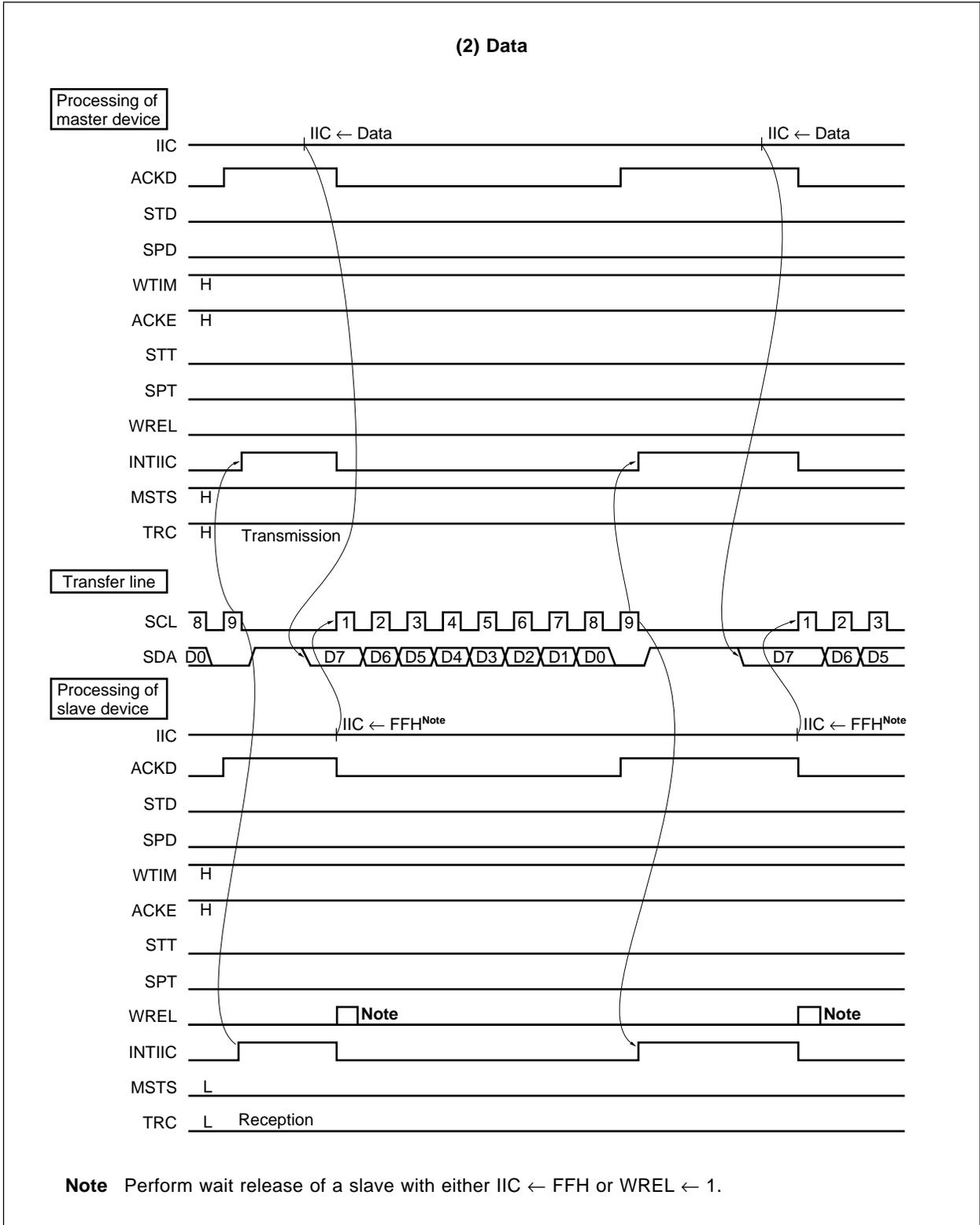
The data input to the SDA pin at the rise of SCL is captured by IIC.

Figure 8-27. Example of Master→ Slave Communication  
(9-clock wait is selected both for master and slave) (1/3)



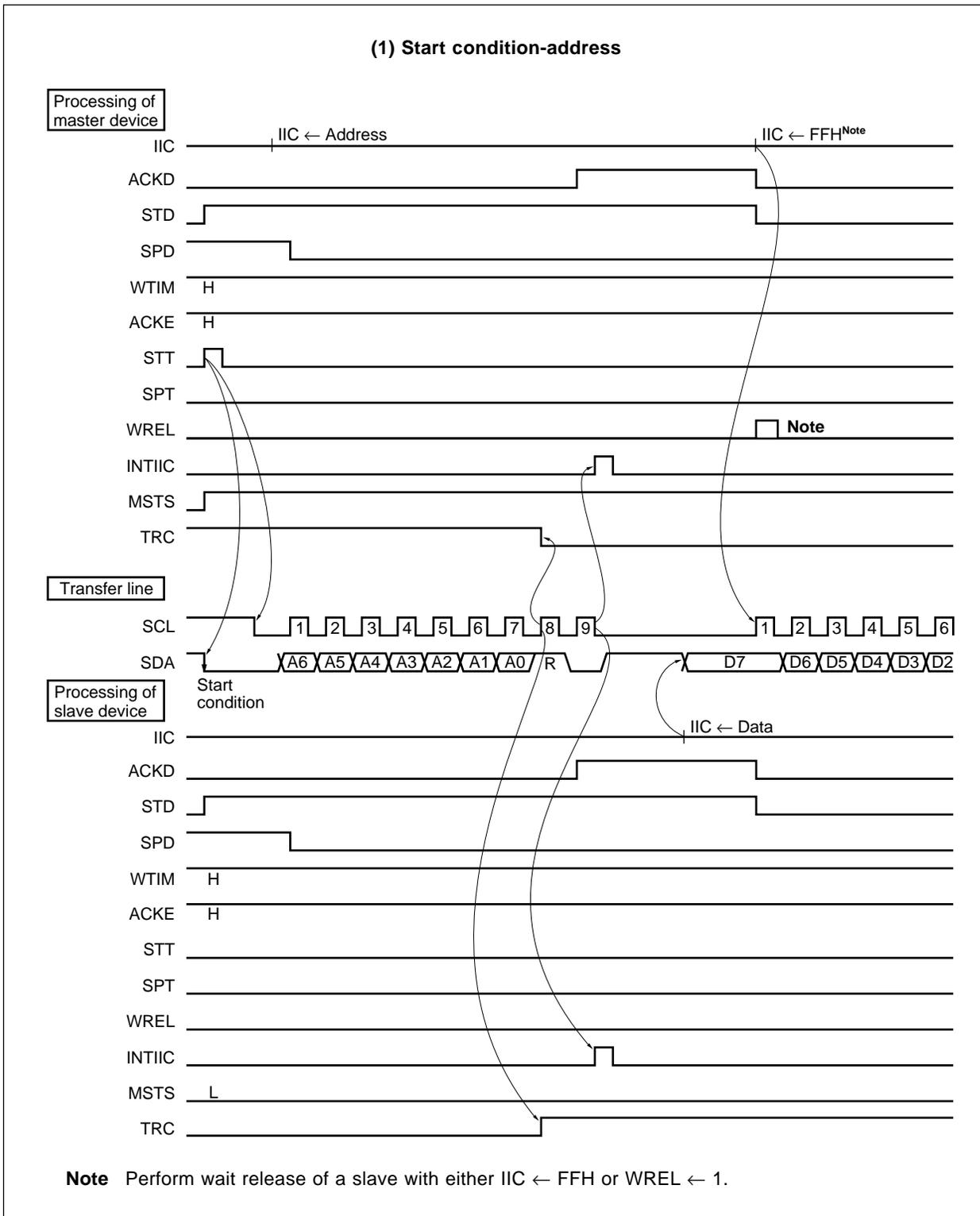
**Note** Perform wait release of a slave with either IIC ← FFH or WREL ← 1.

**Figure 8-27. Example of Master → Slave Communication  
(9-clock wait is selected both for master and slave) (2/3)**

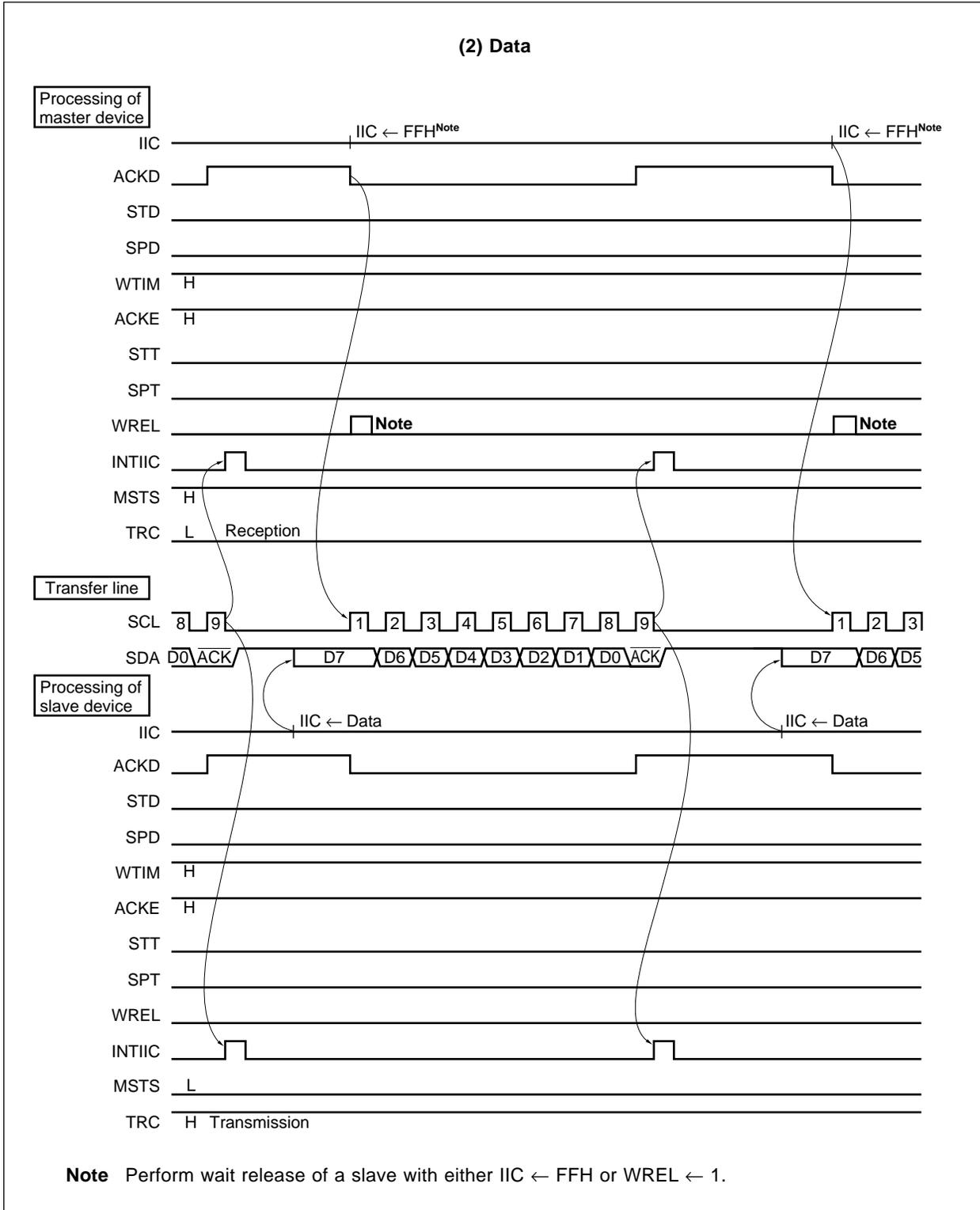




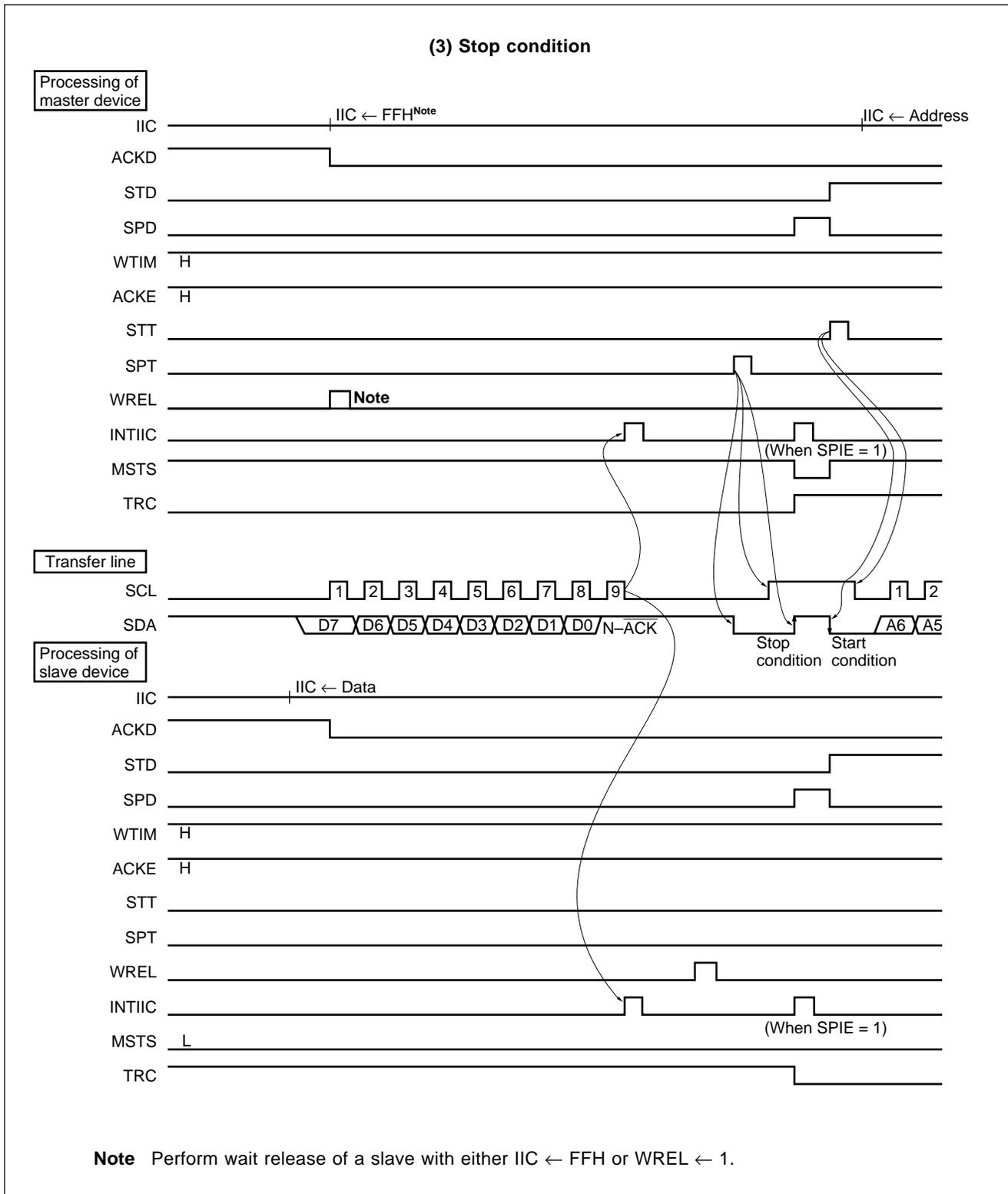
**Figure 8-28. Example of Slave → Master Communication  
(9-clock wait is selected both for master and slave) (1/3)**



**Figure 8-28. Example of Slave → Master Communication (9-clock wait is selected both for master and slave) (2/3)**



**Figure 8-28. Example of Slave → Master Communication  
(9-clock wait is selected both for master and slave) (3/3)**



## 8.5 Baud Rate Generator 0 to 3 (BRG0 to BRG3)

### 8.5.1 Configuration and function

The serial clock of the serial interface can be selected from the baud rate generator output or  $\phi$  (internal system clock) for each channel.

A baud rate generator is provided with the following four systems, which can be set independently.

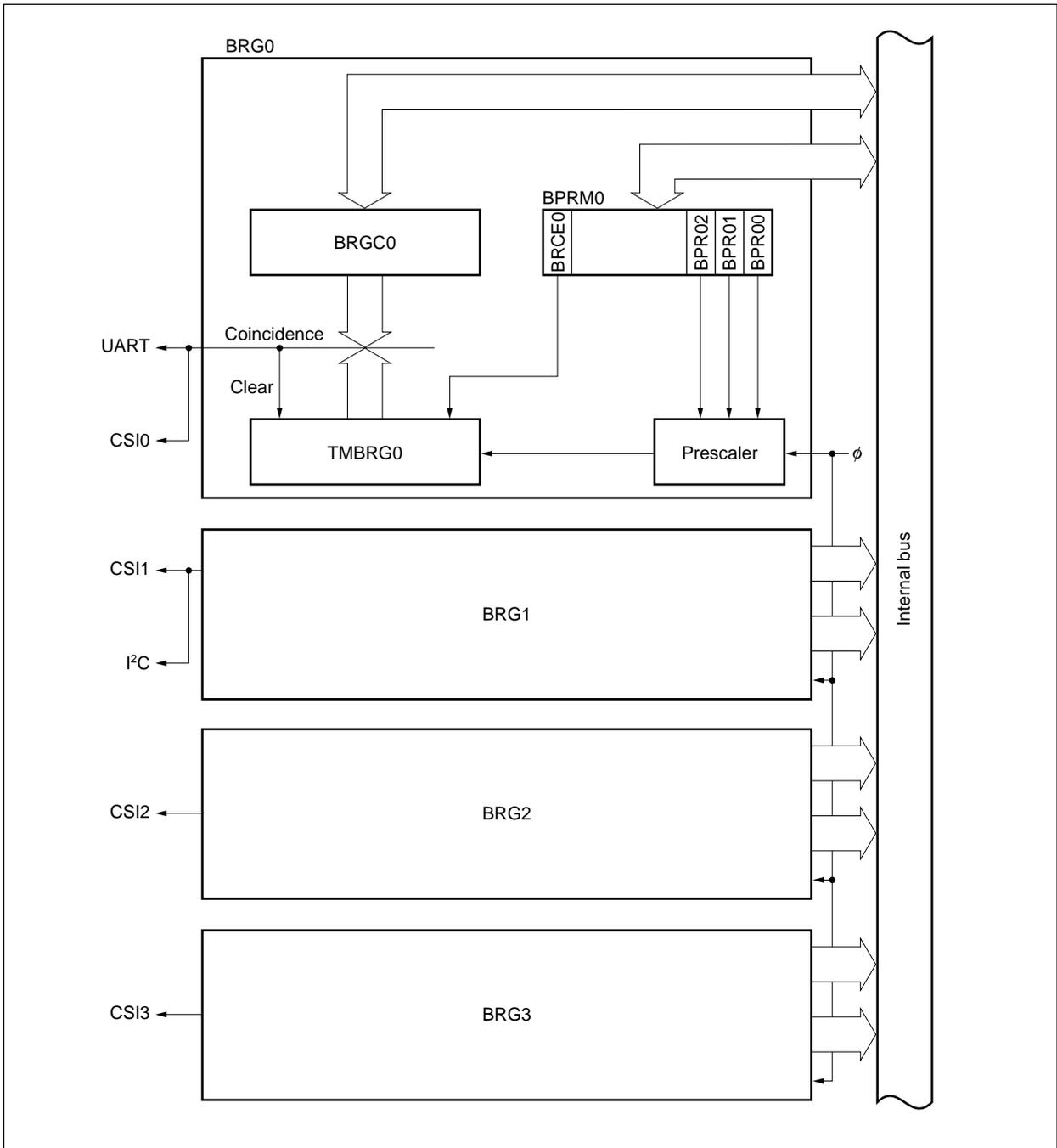
- For UART/CSI0 (BRG0)
- For I<sup>2</sup>C/CSI1 (BRG1)
- For CSI2 (BRG2)
- For CSI3 (BRG3)

The serial clock source for the UART is specified by the SCLS bits of the ASIMn registers. The serial clock source for the CSI is specified by the CLS bits of the CSIMn registers.

When the output of the baud rate generator is specified, the baud rate generator will be used as the clock source.

Because the serial clock for transmission/reception is shared by both the transmission and reception portions, the same baud rate is used for both transmission and reception.

Figure 8-29. Block Diagram of Baud Rate Generator



**(1) Dedicated baud rate generators (BRG0 to BRG3)**

The dedicated baud rate generators (BRGn) consist of an 8-bit timer (TMBRGn) that generates a serial clock for transmission/reception, a compare register (BRGCn), and a prescaler (n = 0 to 3).

**(a) Input clock**

Internal system clock ( $\phi$ ) is input to the BRGn.

**(b) Set-up value of BRGn**

**(i) UART**

If the dedicated baud rate generator is specified for UART as a serial clock source, the actual baud rate can be calculated by the following expression, because a sample rate of x16 is used:

$$\text{Baud rate} = \frac{\phi}{m \times 2^k \times 16 \times 2} \quad [\text{bps}]$$

where,

$\phi$ : Internal system clock frequency [Hz]

m: BRGC0 set-up value ( $1 \leq m \leq 256^{\text{Note}}$ )

k: BPR00 to BPR02 prescaler set-up value

**Note** 256 is set by writing 0 to the BRGC0 register.

**(ii) CSI0 to CSI3**

If the dedicated baud rate generator is specified for CSIn, the actual baud rate can be calculated by the following expression (n = 0 to 3):

$$\text{Baud rate} = \frac{\phi}{m \times 2^k \times 2} \quad [\text{bps}]$$

where,

$\phi$ : Internal system clock frequency [Hz]

m: BRGCn set-up value ( $1 \leq m \leq 256^{\text{Note 1}}$ )

k: BPRn0 to BPRn2 prescaler set-up value<sup>Note 2</sup>

**Notes** 1. m = 256 is set by writing 0 to the BRGCn register.

2. Setting k = 0 is prohibited.

Table 8-6 shows the setup values of the baud rate generator when the typical clocks are used.

Table 8-6. Baud Rate Generators 0 to 3 Set-up Values (when typical clocks are used) (1/2)

(a) UART

Baud Rate [bps]	$\phi = 33$ MHz			$\phi = 25$ MHz			$\phi = 16$ MHz			$\phi = 12.5$ MHz		
	BPR	BRGC	Error	BPR	BRGC	Error	BPR	BRGC	Error	BPR	BRGC	Error
110	—	—	—	5	222	0.02%	5	142	0.03%	4	222	0.02%
150	5	215	0.07%	5	163	0.15%	4	208	0.16%	4	163	0.15%
300	4	215	0.07%	4	163	0.15%	3	208	0.16%	3	163	0.15%
600	3	215	0.07%	3	163	0.15%	2	208	0.16%	2	163	0.15%
1200	2	215	0.07%	2	163	0.15%	1	208	0.16%	1	163	0.15%
2400	1	215	0.07%	1	163	0.15%	0	208	0.16%	0	162	0.47%
4800	0	215	0.07%	0	163	0.15%	0	104	0.16%	0	82	0.76%
9600	0	107	0.39%	0	81	0.47%	0	52	0.16%	0	40	1.73%
10400	0	100	0.84%	0	75	0.16%	0	48	0.16%	0	38	1.16%
19200	0	54	0.54%	0	41	0.72%	0	26	0.16%	0	20	1.73%
31250	0	33	0.00%	0	25	0.00%	0	16	0.00%	0	13	3.85%
38400	0	27	0.54%	0	20	1.73%	0	13	0.16%	0	10	1.73%
76800	0	13	3.29%	0	10	1.73%	0	6	6.99% <sup>Note</sup>	0	5	1.73%
153600	0	7	1.30%	0	5	1.73%	0	3	8.51% <sup>Note</sup>	—	—	—

Baud Rate [bps]	$\phi = 19.660$ MHz			$\phi = 14.746$ MHz			$\phi = 12.288$ MHz			$\phi = 9.830$ MHz		
	BPR	BRGC	Error	BPR	BRGC	Error	BPR	BRGC	Error	BPR	BRGC	Error
110	5	176	0.83%	5	131	0.07%	4	218	0.08%	4	176	0.02%
150	5	128	0.00%	4	192	0.00%	4	160	0.00%	4	128	0.00%
300	4	128	0.00%	3	192	0.00%	3	160	0.00%	3	128	0.00%
600	3	128	0.00%	2	192	0.00%	2	160	0.00%	2	128	0.00%
1200	2	128	0.00%	1	192	0.00%	1	160	0.00%	1	128	0.00%
2400	1	128	0.00%	0	192	0.00%	0	160	0.00%	0	128	0.00%
4800	0	128	0.00%	0	96	0.00%	0	80	0.00%	0	64	0.00%
9600	0	64	0.00%	0	48	0.00%	0	40	0.00%	0	32	0.00%
10400	0	60	1.54%	0	44	0.70%	0	37	0.21%	0	30	1.54%
19200	0	32	0.00%	0	24	0.00%	0	20	0.00%	0	16	0.00%
31250	0	20	1.70%	0	15	1.69%	0	12	2.40%	0	10	1.70%
38400	0	16	0.00%	0	12	0.00%	0	10	0.00%	0	8	0.00%
76800	0	8	0.00%	0	6	0.00%	0	5	0.00%	0	4	0.00%
153600	0	4	0.00%	0	3	0.00%	—	—	—	0	2	0.00%
307200	0	2	0.00%	—	—	—	—	—	—	0	1	0.00%
614400	0	1	0.00%	—	—	—	—	—	—	—	—	—

**Note** Cannot be used because the error is too great.

**Remark**  $\phi$ : Internal system clock

Table 8-6. Baud Rate Generators 0 to 3 Set-up Values (when typical clocks are used) (2/2)

(b) CSI

Baud Rate [bps]	$\phi = 33$ MHz			$\phi = 25$ MHz			$\phi = 16$ MHz			$\phi = 12.5$ MHz		
	BPR	BRGC	Error	BPR	BRGC	Error	BPR	BRGC	Error	BPR	BRGC	Error
1760	—	—	—	5	222	0.02%	5	142	0.03%	4	222	0.02%
2400	5	215	0.07%	5	163	0.15%	4	208	0.16%	4	163	0.15%
4800	4	215	0.07%	4	163	0.15%	3	208	0.16%	3	163	0.15%
9600	3	215	0.07%	3	163	0.15%	2	208	0.16%	2	163	0.15%
19200	2	215	0.07%	2	163	0.15%	1	208	0.16%	1	163	0.15%
38400	1	215	0.07%	1	163	0.15%	1	104	0.16%	1	81	0.47%
76800	1	107	0.39%	1	81	0.47%	1	52	0.16%	1	41	0.76%
153600	1	54	0.54%	1	41	0.76%	1	26	0.16%	1	20	1.73%
166400	1	50	0.84%	1	38	1.16%	1	24	0.16%	1	19	1.16%
307200	1	27	0.54%	1	20	1.73%	1	13	0.16%	1	10	1.73%
614400	1	13	3.29%	1	10	1.73%	1	7	6.99% <sup>Note</sup>	1	5	1.73%
1228800	1	7	4.09%	1	5	1.73%	—	—	—	1	3	15.2% <sup>Note</sup>
2457600	1	3	11.30% <sup>Note</sup>	1	2	27.2% <sup>Note</sup>	—	—	—	—	—	—

Baud Rate [bps]	$\phi = 20$ MHz			$\phi = 14.746$ MHz			$\phi = 12.288$ MHz			$\phi = 9.830$ MHz		
	BPR	BRGC	Error	BPR	BRGC	Error	BPR	BRGC	Error	BPR	BRGC	Error
1760	5	178	0.25%	5	131	0.07%	4	218	0.08%	4	176	0.26%
2400	5	130	0.16%	4	192	0.00%	4	160	0.00%	4	128	0.00%
4800	4	130	0.16%	3	192	0.00%	3	160	0.00%	3	128	0.00%
9600	3	130	0.16%	2	192	0.00%	2	160	0.00%	2	128	0.00%
19200	2	130	0.16%	1	192	0.00%	1	160	0.00%	1	128	0.00%
38400	1	130	0.16%	1	96	0.00%	1	80	0.00%	1	64	0.00%
76800	1	65	0.16%	1	48	0.00%	1	40	0.00%	1	32	0.00%
153600	1	33	1.36%	1	24	0.00%	1	20	0.00%	1	16	0.00%
166400	1	30	0.16%	1	22	0.70%	1	18	2.60%	1	15	1.50%
307200	1	16	1.73%	1	12	0.00%	1	10	0.00%	1	8	0.00%
614400	1	8	1.73%	1	6	0.00%	1	5	0.00%	1	4	0.00%
1228800	1	4	1.73%	1	3	0.00%	1	3	16.7% <sup>Note</sup>	1	2	0.00%
2457600	1	2	1.73%	1	2	25% <sup>Note</sup>	—	—	—	1	1	0.00%

**Note** Cannot be used because the error is too great.

**Remark**  $\phi$ : Internal system clock

**(c) Error of baud rate**

The error of the baud rate is calculated as follows:

$$\text{Error [\%]} = \left( \frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (normal baud rate)}} - 1 \right) \times 100$$

**Example:**  $(9520/9600 - 1) \times 100 = -0.833 \text{ [\%]}$   
 $(5000/4800 - 1) \times 100 = +4.167 \text{ [\%]}$

**(2) Allowable error range of baud rate**

The allowable error range depends on the number of bits of one frame.

The basic limit is  $\pm 5\%$  of baud rate error and  $\pm 4.5\%$  of sample timing with an accuracy of 16 bits. However, the practical limit should be  $\pm 2.3 \%$  of baud rate error, assuming that both the transmission and reception sides contain an error.

**8.5.2 Baud rate generator compare registers 0 to 3 (BRGC0 to BRGC3)**

These are 8-bit compare registers that set a timer/count value for the dedicated baud rate generator.

These registers can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
BRGC0	BRG07	BRG06	BRG05	BRG04	BRG03	BRG02	BRG01	BRG00	Address FFFFF084H	After reset Undefined
BRGC1	BRG17	BRG16	BRG15	BRG14	BRG13	BRG12	BRG11	BRG10	Address FFFFF094H	After reset Undefined
BRGC2	BRG27	BRG26	BRG25	BRG24	BRG23	BRG22	BRG21	BRG20	Address FFFFF0A4H	After reset Undefined
BRGC3	BRG37	BRG36	BRG35	BRG34	BRG33	BRG32	BRG31	BRG30	Address FFFFF0B4H	After reset Undefined

**Caution** The internal timer (TMBR<sub>n</sub>) is cleared by writing the BRGC registers. Therefore, do not rewrite or program the BRGC<sub>n</sub> registers during transmission/reception operation.

**Remark** n = 0 to 3

**8.5.3 Baud rate generator prescaler mode registers 0 to 3 (BPRM0 to BPRM3)**

These registers control the timer/count operation of the dedicated baud rate generator and select a count clock. They can be read/written in 8- or 1-bit units.

BPRM0	7	6	5	4	3	2	1	0	Address FFFFF086H	After reset 00H
	BRCE0	0	0	0	0	BPR02	BPR01	BPR00		
BPRM1	7	6	5	4	3	2	1	0	Address FFFFF096H	After reset 00H
	BRCE1	0	0	0	0	BPR12	BPR11	BPR10		
BPRM2	7	6	5	4	3	2	1	0	Address FFFFF0A6H	After reset 00H
	BRCE2	0	0	0	0	BPR22	BPR21	BPR20		
BPRM3	7	6	5	4	3	2	1	0	Address FFFFF0B6H	After reset 00H
	BRCE3	0	0	0	0	BPR32	BPR31	BPR30		

Bit Position	Bit Name	Function																																
7	BRCE <sub>n</sub>	Baud Rate Generator Count Enable Controls count operation of BRG <sub>n</sub> . 0 : Stops count operation with cleared 1 : Enables count operation																																
2 to 0	BPR <sub>n</sub> 3 to BPR <sub>n</sub> 0	Baud Rate Generator Prescaler Specifies count clock input to internal timer (TMBRG <sub>n</sub> ). <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>BPR<sub>n</sub>2</th> <th>BPR<sub>n</sub>1</th> <th>BPR<sub>n</sub>0</th> <th>Count Clock</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td><math>\phi</math> (k = 0): CSI use disabled</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td><math>\phi/2</math> (k = 1)</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td><math>\phi/4</math> (k = 2)</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td><math>\phi/8</math> (k = 3)</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td><math>\phi/16</math> (k = 4)</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td><math>\phi/32</math> (k = 5)</td> </tr> <tr> <td colspan="3">Others</td> <td>Setting prohibited</td> </tr> </tbody> </table> <p style="margin-left: 20px;">k: Set value of prescaler, <math>\phi</math>: Internal system clock</p>	BPR <sub>n</sub> 2	BPR <sub>n</sub> 1	BPR <sub>n</sub> 0	Count Clock	0	0	0	$\phi$ (k = 0): CSI use disabled	0	0	1	$\phi/2$ (k = 1)	0	1	0	$\phi/4$ (k = 2)	0	1	1	$\phi/8$ (k = 3)	1	0	0	$\phi/16$ (k = 4)	1	0	1	$\phi/32$ (k = 5)	Others			Setting prohibited
BPR <sub>n</sub> 2	BPR <sub>n</sub> 1	BPR <sub>n</sub> 0	Count Clock																															
0	0	0	$\phi$ (k = 0): CSI use disabled																															
0	0	1	$\phi/2$ (k = 1)																															
0	1	0	$\phi/4$ (k = 2)																															
0	1	1	$\phi/8$ (k = 3)																															
1	0	0	$\phi/16$ (k = 4)																															
1	0	1	$\phi/32$ (k = 5)																															
Others			Setting prohibited																															

**Caution** Do not change the count clock during transmission/reception operation.

**Remark** n = 0 to 3

## 8.6 Selection of Operational Serial Interface

CSI0 and CSI1 of the V854 are alternate pins for UART and I<sup>2</sup>C. Therefore, they are used selecting either UART or I<sup>2</sup>C.

The selection is made by the following registers.

### (1) Selecting CSI0 or UART

The setting is made by the ASIM0 register and the CSIM0 register.

ASIM0 Register		CSIM0 Register		Selection of Operational Peripheral I/O
TXE	RXE	CTXE0	CRXE0	
0	0	0	0	Operation stops
0	1	0	0	Selects UART
1	0	0	0	
1	1	0	0	
0	0	0	1	Selects CSI
0	0	1	0	
0	0	1	1	
Others				Setting prohibited

### (2) Selecting CSI1 or I<sup>2</sup>C

The setting is made by the IICC register and the CSIM1 register.

IICC Register	CSIM1 Register		Selection of Operational Peripheral I/O
IICE	CTXE1	CRXE1	
0	0	0	Operation stops
1	0	0	Selects I <sup>2</sup> C
0	0	1	Selects CSI
0	1	0	
0	1	1	
Others			Setting prohibited

**[MEMO]**

## CHAPTER 9 A/D CONVERTER

### 9.1 Features

- Analog input: 16 channels
- 8-bit A/D converter
- On-chip A/D conversion result register (ADCR0 to ADCR7)  
8 bits x 8
- A/D conversion trigger mode
  - A/D trigger mode
  - Timer trigger mode
  - External trigger mode
- Sequential conversion

### 9.2 Configuration

The A/D converter of the V854 adopts the sequential conversion method, and uses the A/D converter mode registers (ADM0, ADM1), and ADCRn register to perform A/D conversion operations (n = 0 to 7).

#### (1) Input circuit

Selects the analog input (ANI0 to ANI15) according to the mode set to the ADM0 and ADM1 registers and then sends it to the sample and hold circuit.

#### (2) Sample and hold circuit

Samples the analog input sent from the input circuit one by one and sends it to the comparator. During A/D conversion operations, holds the analog input sampled.

#### (3) Voltage comparator

Compares the voltage difference between the input analog input and voltage tap of the serial resistor string output.

#### (4) Serial resistor string

Generates voltage to coincide with analog input.

The serial resistor string is connected between the reference voltage pin for A/D converter ( $AV_{REF}$ ) and GND pin for A/D converter ( $AV_{SS}$ ). The serial resistor consists of 255 equivalent resistors and two resistors with half the resistance so that the connection between the two pins is made of 256 equal voltage steps.

The voltage tap of the serial resistor string is selected by a tap selector controlled by the successive approximation register (SAR).

#### (5) Successive approximation register (SAR)

8-bit register for setting data for which the value of the voltage tap of the serial resistor string coincides with the voltage value of the analog input from the most significant bit (MSB) in 1-bit units.

When the setting is made to the least significant bit (LSB) (A/D conversion ends), the contents of the SAR (conversion result) are held in the A/D conversion result register (ADCRn).

**(6) A/D Conversion Result Register n (ADCRn)**

8-bit register for retaining the A/D conversion result. The conversion result is loaded from the sequential conversion register (SAR) each time A/D conversion ends.

This register becomes undefined by  $\overline{\text{RESET}}$  input.

**(7) Controller**

Selects the analog input, generates the sample hold circuit operation timing, and controls the conversion trigger according to the mode set to the ADM0/ADM1 register.

**(8) ANI0 to ANI15 pins**

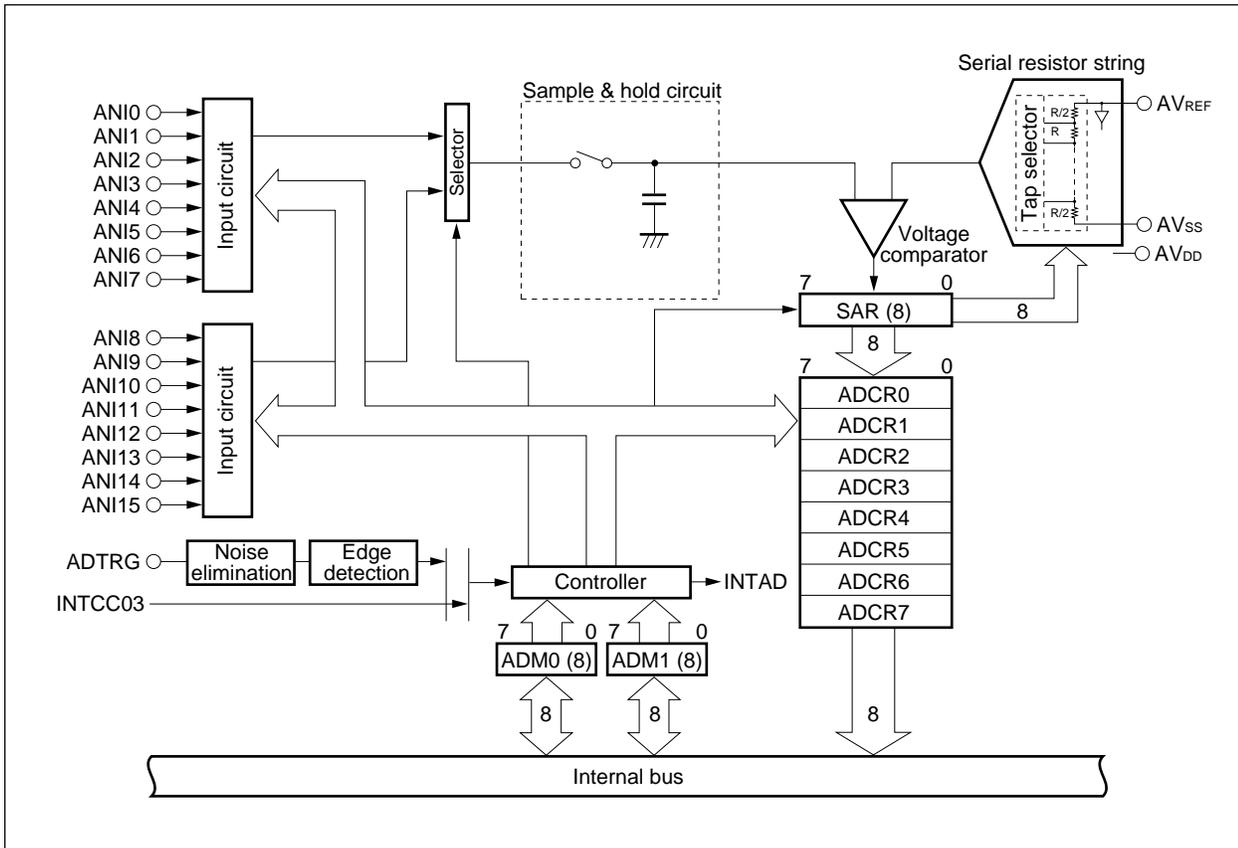
16-channel analog input pins for the A/D converter. Input the analog signal to be A/D converted.

**Caution** Use ANI0 to ANI15 input voltages within the specification. Especially, if the voltage exceeding  $V_{DD}$  or less than  $V_{SS}$  (even if it is within the range of the absolute maximum rating) is input, the conversion value of the channel may become undefined or the conversion value of other channels may be affected.

**(9) AVREF pin**

Pin for inputting the reference voltage of the A/D converter. Converts signals input to the ANI0 to ANI15 pins to digital signals based on the voltage applied between AVREF and AVSS.

Figure 9-1. Block Diagram of A/D Converter



### 9.3 Control Register

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

The ADM0 register is an 8-bit register which executes the selection of the analog input pin, specification of the operation mode, and conversion operations.

This register can be read/written in 8- or 1-bit units. However, when the data is written to the ADM0 register during A/D conversion operations, the conversion operation is initialized and conversion is executed from the beginning. Bit 6 cannot be written in and writing executed is ignored.

	7	6	5	4	3	2	1	0		
ADM0	CE	CS	BS	MS	PS	ANIS2	ANIS1	ANIS0	Address FFFFF380H	After reset 00H

Bit Position	Bit Name	Function
7	CE	Convert Enable Enables or disables A/D conversion operation. 0: Disabled 1: Enabled
6	CS	Converter Status Indicates the status of A/D converter. This bit is read only. 0: Stops 1: Operates
5	BS	Buffer Select Specifies buffer mode in the select mode. 0: 1-buffer mode 1: 4-buffer mode
4	MS	Mode Select Specifies operation mode of A/D converter. 0: Scan mode 1: Select mode
3	PS	Pin Select Switches the analog input pin. 0: Selects ANI0 to ANI7 1: Selects ANI8 to ANI15

Bit Position	Bit Name	Function				
2 to 0	ANIS2 to ANIS0	Analog Input Select Specifies analog input pin to A/D convert.				
		ANIS2	ANIS1	ANIS0	Select Mode	Scan Mode
		0	0	0	ANI0/ANI8	ANI0/ANI8
		0	0	1	ANI1/ANI9	ANI0, ANI1/ANI8, ANI9
		0	1	0	ANI2/ANI10	ANI0 to ANI2/ANI8 to ANI10
		0	1	1	ANI3/ANI11	ANI0 to ANI3/ANI8 to ANI11
		1	0	0	ANI4/ANI12	ANI0 to ANI4/ANI8 to ANI12
		1	0	1	ANI5/ANI13	ANI0 to ANI5/ANI8 to ANI13
		1	1	0	ANI6/ANI14	ANI0 to ANI6/ANI8 to ANI14
		1	1	1	ANI7/ANI15	ANI0 to ANI7/ANI8 to ANI15

**Caution** When the CE bit is 1 in the timer trigger mode and external trigger mode, the trigger signal standby state is set. To clear the CE bit, write “0” or reset.

In the A/D trigger mode, the conversion trigger is set by writing 1 to the CE bit. After the operation, when the mode is changed to the timer trigger mode or external trigger mode without clearing the CE bit, the trigger input standby state is set immediately after the change.

**(2) A/D converter mode register 1 (ADM1)**

The ADM1 register is an 8-bit register which specifies the conversion operation time and trigger mode. This register can be read/written in 8- or 1-bit units. However, when the data is written to the ADM1 register during A/D conversion, the conversion operation is initialized and conversion is executed from the beginning again.

	7	6	5	4	3	2	1	0		
ADM1	0	0	TRG1	TRG0	0	FR2	FR1	FR0	Address FFFFF382H	After reset 07H

Bit Position	Bit Name	Function																																																																		
5 and 4	TRG1 and TRG0	Trigger Mode Specifies trigger mode. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">TRG1</th> <th style="width: 10%;">TRG0</th> <th style="width: 80%;">Trigger Mode</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>A/D trigger mode</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Timer trigger mode</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>External trigger mode</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	TRG1	TRG0	Trigger Mode	0	0	A/D trigger mode	1	1	Timer trigger mode	1	0	External trigger mode	1	1	Setting prohibited																																																			
TRG1	TRG0	Trigger Mode																																																																		
0	0	A/D trigger mode																																																																		
1	1	Timer trigger mode																																																																		
1	0	External trigger mode																																																																		
1	1	Setting prohibited																																																																		
2 to 0	FR2 to FR0	Frequency Specifies conversion operation time. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th rowspan="2" style="width: 10%;">FR2</th> <th rowspan="2" style="width: 10%;">FR1</th> <th rowspan="2" style="width: 10%;">FR0</th> <th rowspan="2" style="width: 15%;">Number of Conversion Clock</th> <th colspan="3" style="width: 50%;">Conversion Operation Time (<math>\mu</math>s)</th> </tr> <tr> <th style="width: 16.6%;"><math>\phi = 33</math> MHz</th> <th style="width: 16.6%;"><math>\phi = 25</math> MHz</th> <th style="width: 16.6%;"><math>\phi = 16</math> MHz</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">50</td> <td style="text-align: center;">-</td> <td style="text-align: center;">-</td> <td style="text-align: center;">3.1</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">60</td> <td style="text-align: center;">-</td> <td style="text-align: center;">-</td> <td style="text-align: center;">3.8</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">80</td> <td style="text-align: center;">-</td> <td style="text-align: center;">3.2</td> <td style="text-align: center;">5.0</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">100</td> <td style="text-align: center;">3.0</td> <td style="text-align: center;">4.0</td> <td style="text-align: center;">6.3</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">120</td> <td style="text-align: center;">3.6</td> <td style="text-align: center;">4.8</td> <td style="text-align: center;">7.5</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">140</td> <td style="text-align: center;">4.2</td> <td style="text-align: center;">5.6</td> <td style="text-align: center;">8.8</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">180</td> <td style="text-align: center;">5.4</td> <td style="text-align: center;">7.2</td> <td style="text-align: center;">11.3</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">200</td> <td style="text-align: center;">6.0</td> <td style="text-align: center;">8.0</td> <td style="text-align: center;">12.5</td> </tr> </tbody> </table>	FR2	FR1	FR0	Number of Conversion Clock	Conversion Operation Time ( $\mu$ s)			$\phi = 33$ MHz	$\phi = 25$ MHz	$\phi = 16$ MHz	0	0	0	50	-	-	3.1	0	0	1	60	-	-	3.8	0	1	0	80	-	3.2	5.0	0	1	1	100	3.0	4.0	6.3	1	0	0	120	3.6	4.8	7.5	1	0	1	140	4.2	5.6	8.8	1	1	0	180	5.4	7.2	11.3	1	1	1	200	6.0	8.0	12.5
FR2	FR1	FR0					Number of Conversion Clock	Conversion Operation Time ( $\mu$ s)																																																												
			$\phi = 33$ MHz	$\phi = 25$ MHz	$\phi = 16$ MHz																																																															
0	0	0	50	-	-	3.1																																																														
0	0	1	60	-	-	3.8																																																														
0	1	0	80	-	3.2	5.0																																																														
0	1	1	100	3.0	4.0	6.3																																																														
1	0	0	120	3.6	4.8	7.5																																																														
1	0	1	140	4.2	5.6	8.8																																																														
1	1	0	180	5.4	7.2	11.3																																																														
1	1	1	200	6.0	8.0	12.5																																																														

★

**(3) A/D conversion result register (ADCR0 to ADCR7)**

The ADCRn register is an 8-bit register holding the A/D conversion results. It is provided with eight 8-bit registers.

This register can only be read in 8-/1-bit units.

	7	6	5	4	3	2	1	0		
ADCRn	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	Address FFFFF390H to FFFFF39EH	After reset Undefined

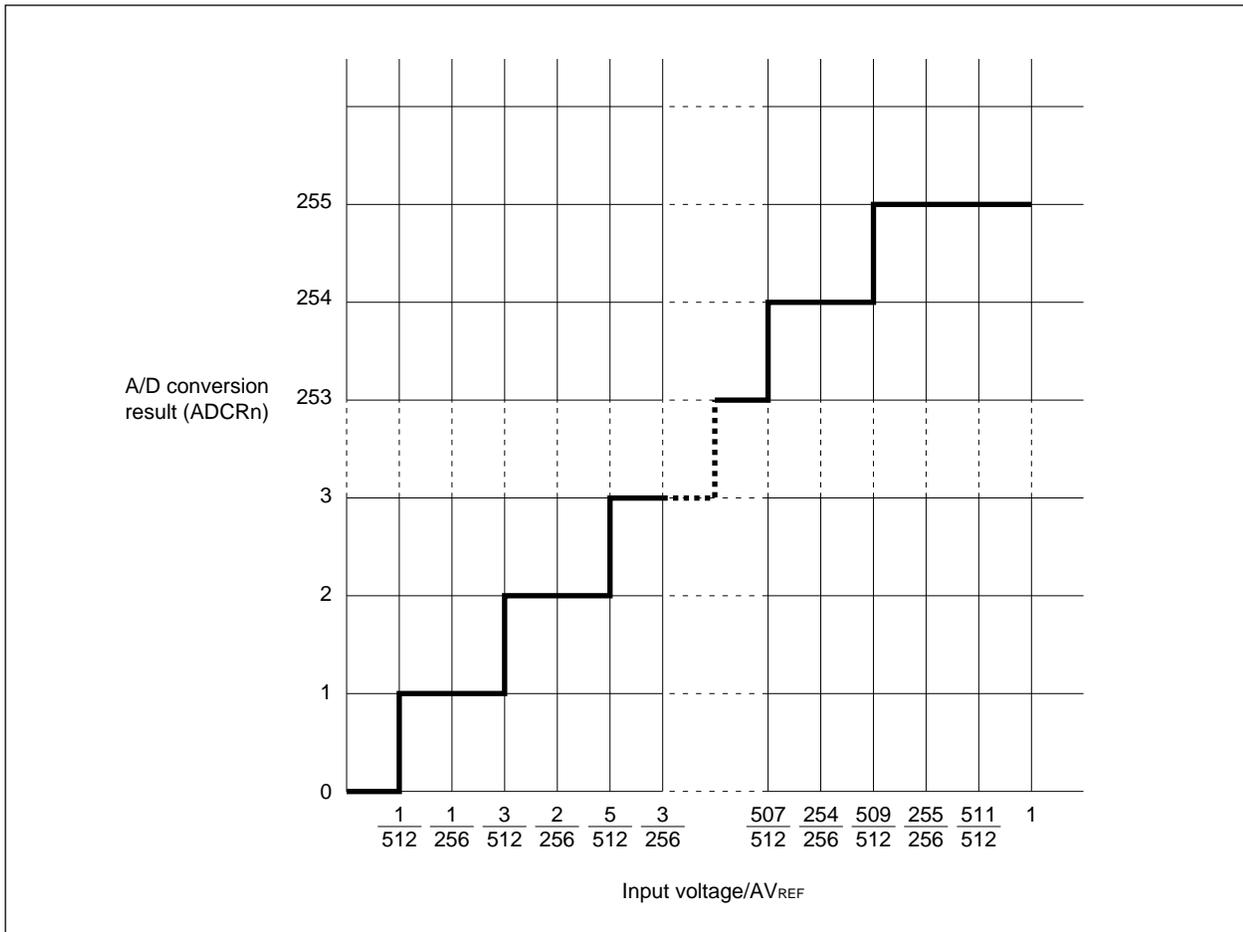
**Remark** n = 0 to 7

The following shows the correspondence of each analog input pin to the ADCRn register (except 4-buffer mode).

Analog Input Pin		ADCRn Register
PS = 0	PS = 1	
ANI0	ANI8	ADCR0
ANI1	ANI9	ADCR1
ANI2	ANI10	ADCR2
ANI3	ANI11	ADCR3
ANI4	ANI12	ADCR4
ANI5	ANI13	ADCR5
ANI6	ANI14	ADCR6
ANI7	ANI15	ADCR7

Figure 9-2 shows the relation between the analog input voltage and the A/D conversion result.

**Figure 9-2. Relation between Analog Input Voltage and A/D Conversion Result**



## 9.4 A/D Converter Operation

### 9.4.1 Basic operation of A/D converter

A/D conversion is executed in the following order.

- (1) The selection of the analog input and specification of the operation mode and trigger mode, etc., should be set in the ADMn register<sup>Note 1</sup> (n = 0, 1).  
When the CE bit of the ADM0 register is set (1), A/D conversion starts during the A/D trigger mode. During the timer trigger mode and external trigger mode, the trigger standby state<sup>Note 2</sup> is set.
- (2) The voltage generated from the voltage tap of the serial resistor string and analog input are compared by the comparator.
- (3) When the comparison of the 8 bits ends, the conversion results are stored in the ADCRn register. When A/D conversion is performed for the specified number of times, the A/D conversion end interrupt (INTAD) is generated (n = 0 to 7).

- Notes**
1. When the ADMn register (n = 0, 1) is changed during A/D conversion, the A/D conversion operation started before the change is stopped and the conversion results are not stored in the ADCRn register (n = 0 to 7).
  2. During the timer trigger mode and external trigger mode, if the CE bit of the ADM0 register is set to 1, the mode changes to the trigger standby state. The A/D conversion operation is started by the trigger signal, and the trigger standby state is returned when the A/D conversion operation ends.

### 9.4.2 Operation mode and trigger mode

The A/D converter can specify various conversion operations by specifying the operation mode and trigger mode. The operation mode and trigger mode are set by the ADMn register (n = 0, 1).

The following shows the relation between the operation mode and trigger mode.

	Trigger Mode	Operation Mode		Setting Value		Analog Input
				ADM0	ADM1	
★	A/D trigger	Select	1 buffer	xx01xxxxB	00000xxxB	ANI0 to ANI15
			4 buffers	xx11xxxxB	00000xxxB	
		Scan	xxx0xxxxB	00000xxxB		
★	Timer trigger	Select	1 buffer	xx01xxxxB	00010xxxB	
			4 buffers	xx11xxxxB	00010xxxB	
		Scan	xxx0xxxxB	00010xxxB		
★	External trigger	Select	1 buffer	xx01xxxxB	00100xxxB	
			4 buffers	xx11xxxxB	00100xxxB	
		Scan	xxx0xxxxB	00100xxxB		

**(1) Trigger mode**

There are three types of trigger modes which serve as the start timing of A/D conversion processing: A/D trigger mode, timer trigger mode, and external trigger mode. These trigger modes are set by the ADM0 register.

**(a) A/D trigger mode**

Generates the conversion timing of the analog input for the ANI0 to ANI15 pins inside the A/D converter unit.

**(b) Timer trigger mode**

Specifies the conversion timing of the analog input set for the ANI0 to ANI15 pins using the values set to the RPU compare register.

This register creates the analog input conversion timing by generating the coincidence interrupts of the capture/compare registers (CC03) connected to the 24-bit TM10.

**(c) External trigger mode**

Mode which specifies the conversion timing of the analog input to the ANI0 to ANI15 pins using the ADTRG pin.

**(2) Operation mode**

There are two types of operation modes which set the ANI0 to ANI15 pins: select mode and scan mode. The select mode has sub-modes including the one-buffer mode and four-buffer mode. These modes are set by the ADM0 register.

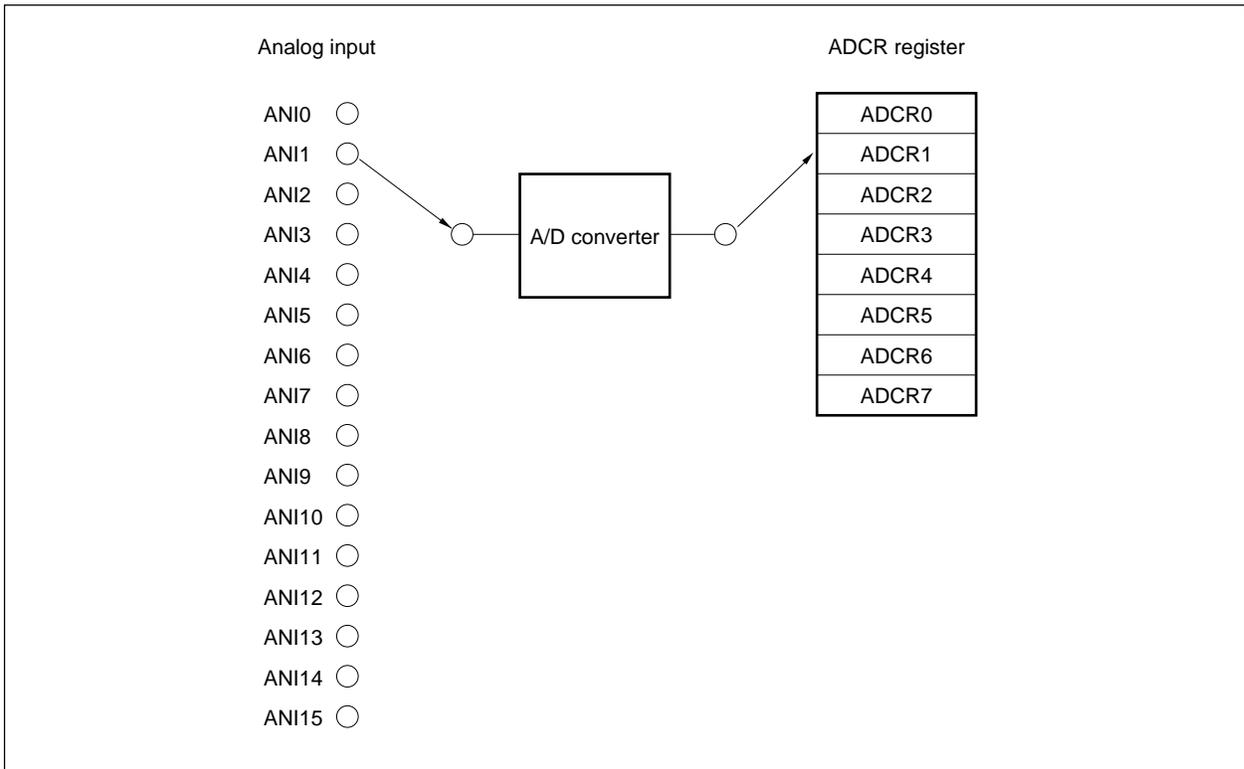
**(a) Select mode**

A/D converts one analog input specified by the ADM0 register. The conversion results are stored in the ADCRn register corresponding to the analog input (n = 0 to 7). For this mode, the one-buffer mode and four-buffer mode are provided for storing the A/D conversion results.

- **One-buffer mode**

A/D converts one analog input specified by the ADM0 register. The conversion results are stored in the ADCRn register corresponding to the analog input. The analog input and ADCRn register correspond one to one, and an A/D conversion end interrupt (INTAD) is generated each time one A/D conversion ends.

**Figure 9-3. Operation Timing Example of Select Mode: 1-Buffer Mode (ANI1)**



- **Four-buffer mode**

A/D converts one analog input four times and stores the results in the four registers corresponding to analog input. The A/D conversion end interrupt (INTAD) is generated when the four A/D conversions end.

**Figure 9-4. Operation Timing Example of Select Mode: 4-Buffer Mode (ANI6) (1/2)**

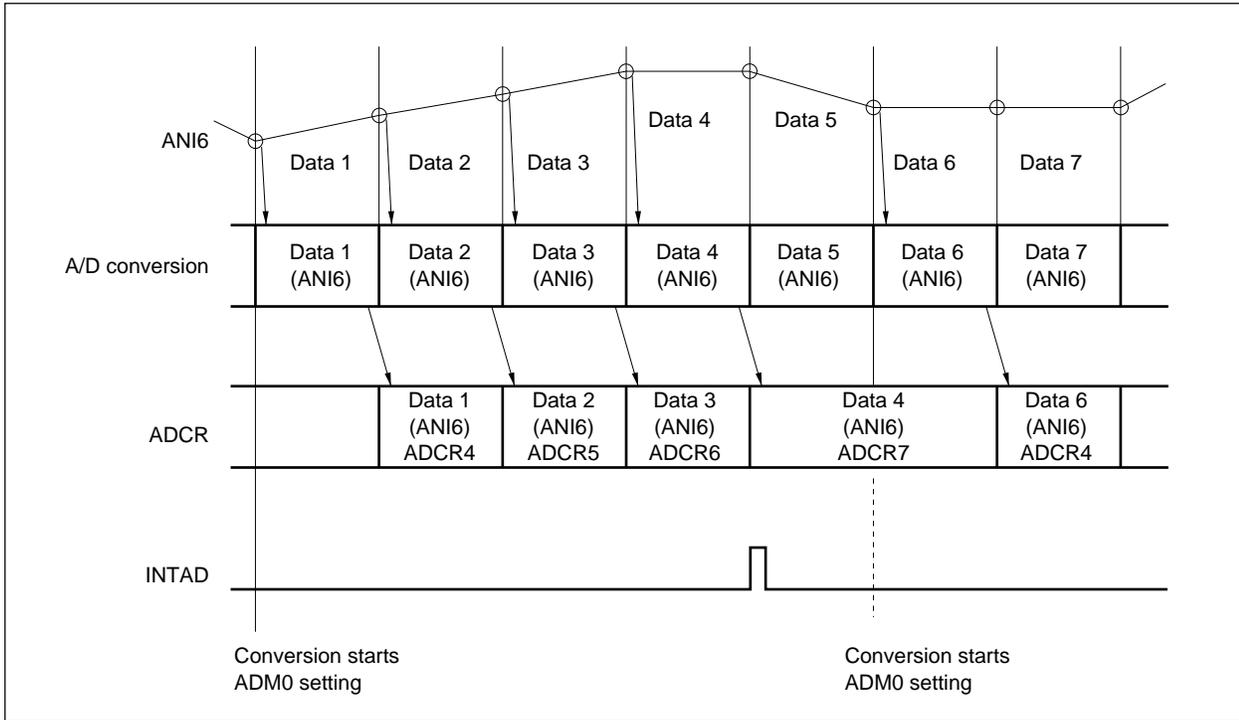
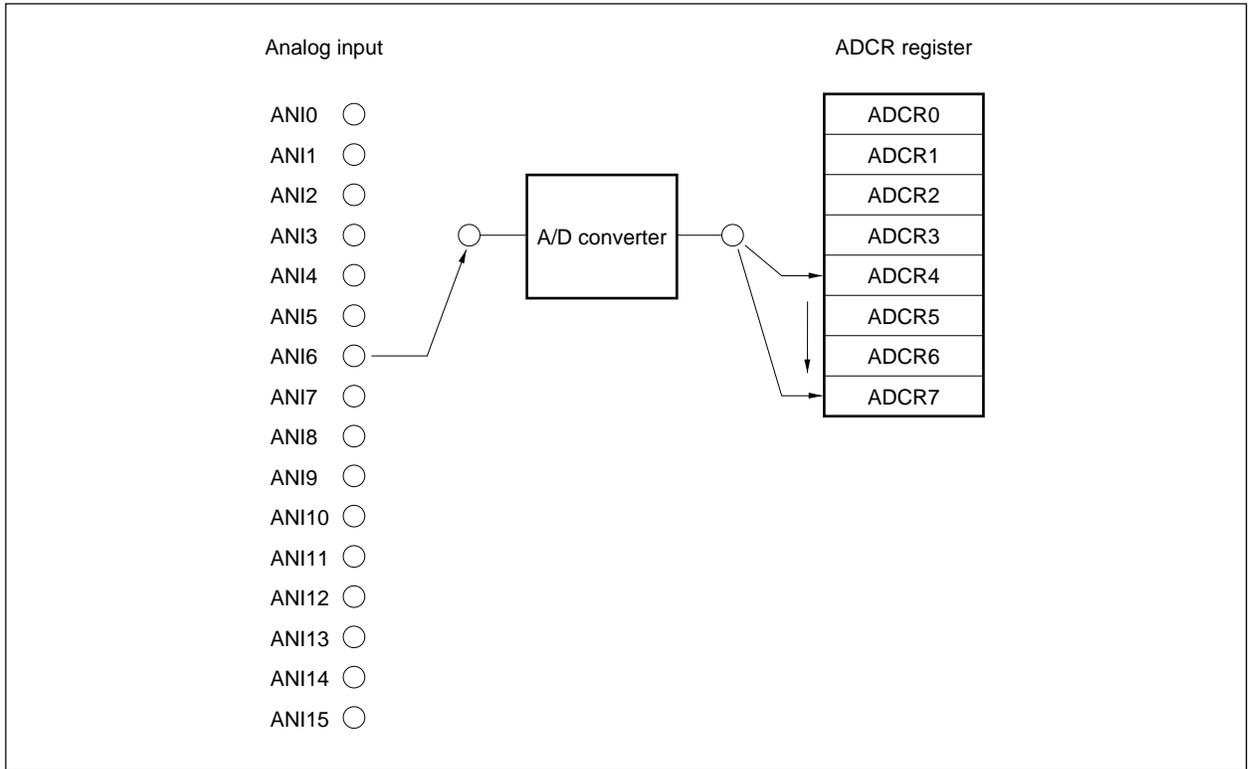


Figure 9-4. Operation Timing Example of Select Mode: 4-Buffer Mode (ANI6) (2/2)



**(b) Scan mode**

Selects the analog inputs specified by the ADM0 register sequentially from the ANI0 pin, and A/D conversion is executed. The A/D conversion results are stored in the ADCRn register corresponding to the analog input. When the conversion of the specified analog input ends, the INTAD interrupt is generated (n = 0 to 7).

**Figure 9-5. Operation Timing Example of Scan Mode: 4-Channel Scan (ANI0 to ANI3) (1/2)**

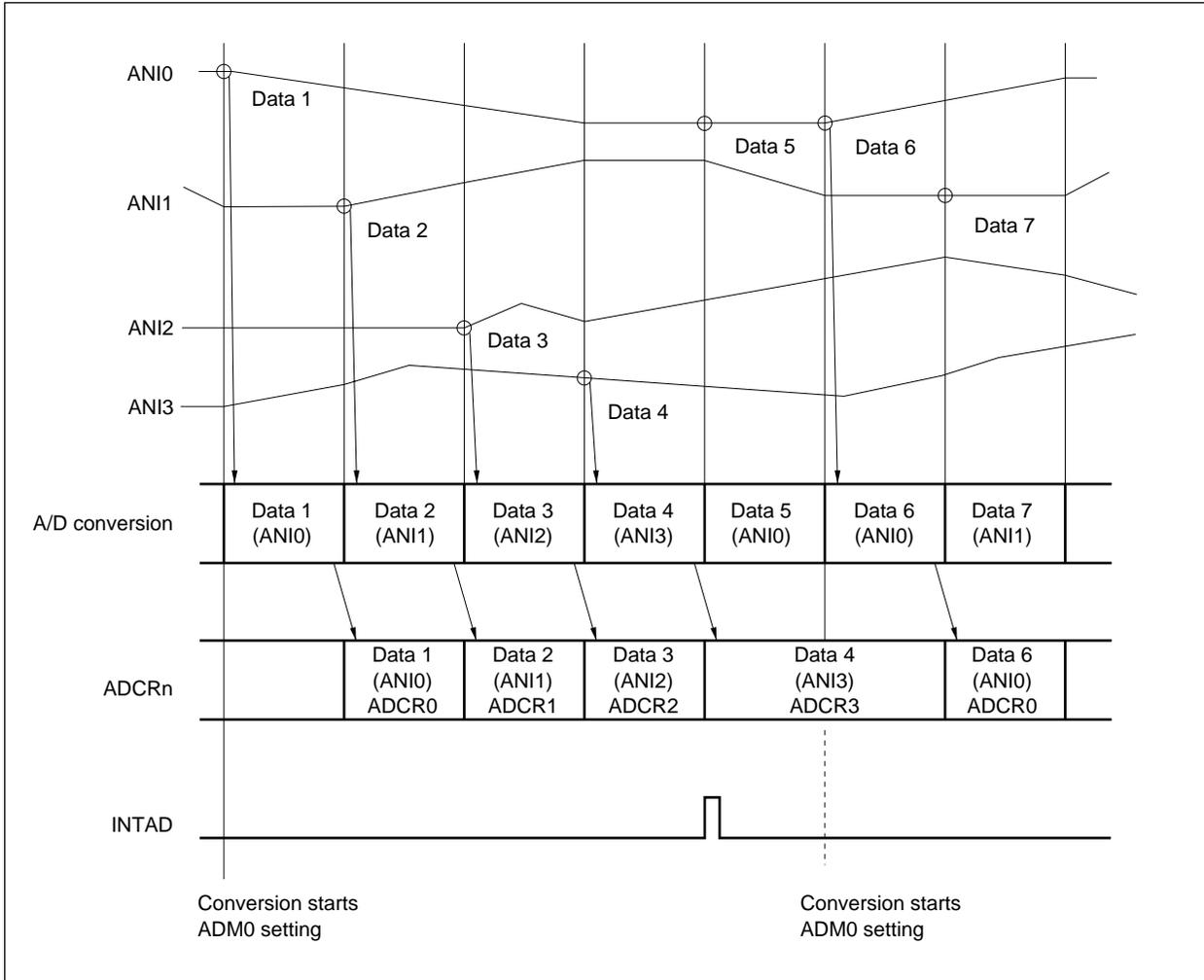
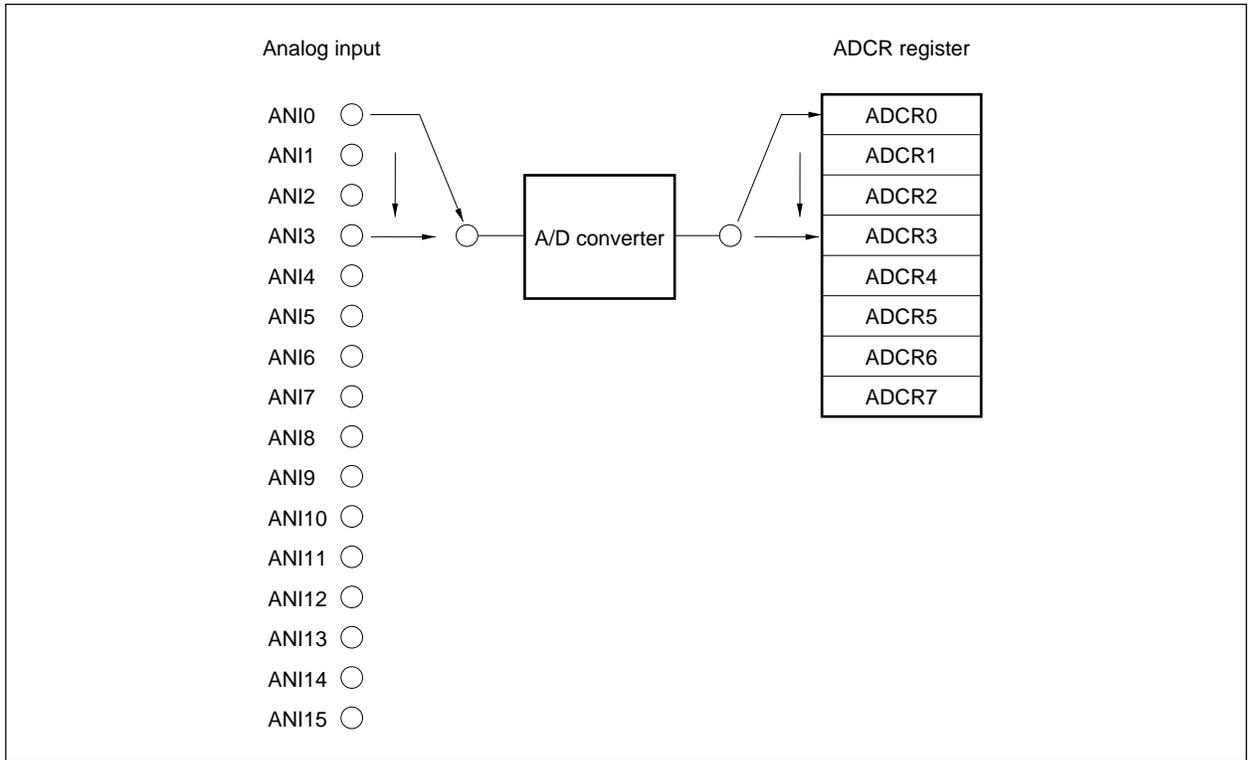


Figure 9-5. Operation Timing Example of Scan Mode: 4-Channel Scan (ANI0 to ANI3) (2/2)



## 9.5 Operation in the A/D Trigger Mode

When the CE bit of the ADM0 register is set to 1, A/D conversion is started.

### 9.5.1 Select mode operation

The A/D converter converts the analog input specified by the ADM0 register. The conversion results are stored in the ADCRn register corresponding to the analog input. In the select mode, the one-buffer mode and four-buffer mode are supported according to the storing method employed for the A/D conversion results (n = 0 to 7).

#### (1) 1-buffer mode (A/D trigger select: 1-buffer)

The A/D converter converts one analog input once. The conversion results are stored in one ADCRn register. The analog input and ADCRn register correspond one to one. (Refer to **Table 9-1, Figure 9-6.**)

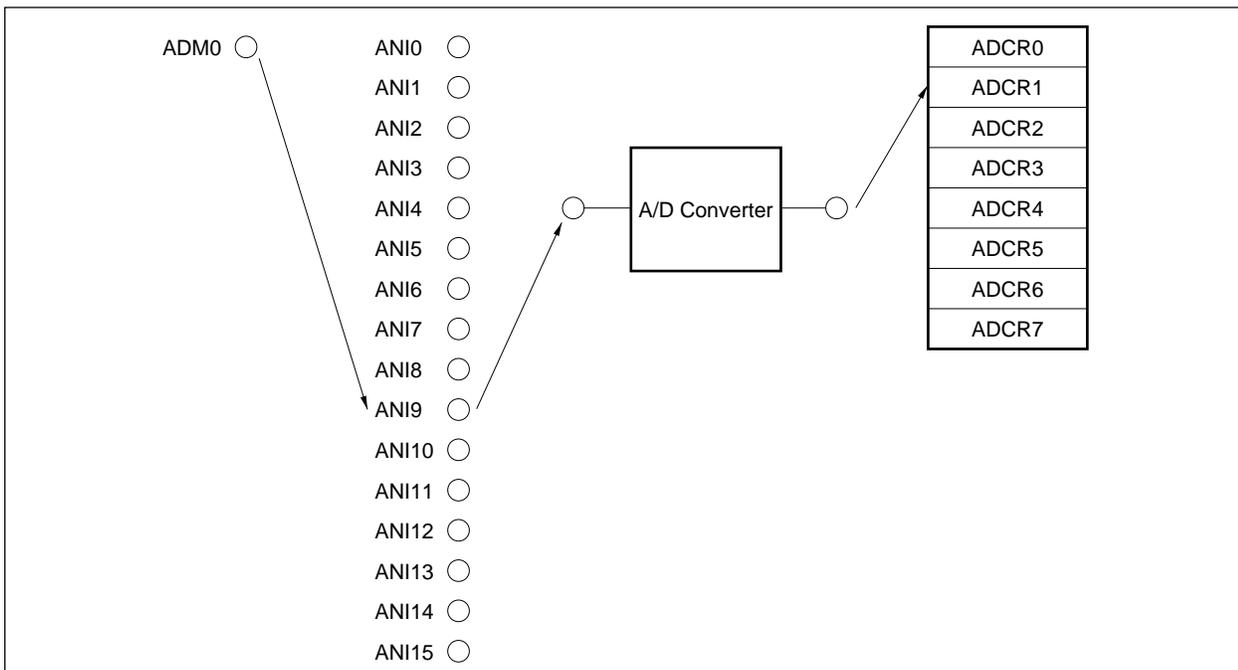
Each time an A/D conversion is executed, an INTAD interrupt is generated and the AD conversion terminates. When 1 is written to the CE bit of the ADM0 register, A/D conversion can be restarted.

This mode is suitable for applications which read out the result in each A/D conversion.

★ **Table 9-1. Correspondence between Analog Input Pin and ADCRn Register (1-buffer mode (A/D trigger select 1-buffer))**

Analog Input	A/D Conversion Results Register
ANI0/ANI8	ADCR0
ANI1/ANI9	ADCR1
ANI2/ANI10	ADCR2
ANI3/ANI11	ADCR3
ANI4/ANI12	ADCR4
ANI5/ANI13	ADCR5
ANI6/ANI14	ADCR6
ANI7/ANI15	ADCR7

★ **Figure 9-6. Example of 1-Buffer Mode (A/D trigger select 1-buffer) Operation**



**(2) 4-buffer mode (A/D trigger select: 4-buffer)**

The A/D converter converts one analog input four times and stores the results in four ADCRn registers. (Refer to **Table 9-2, Figure 9-7.**) When A/D conversion ends four times, an INTAD interrupt is generated and the A/D conversion terminates.

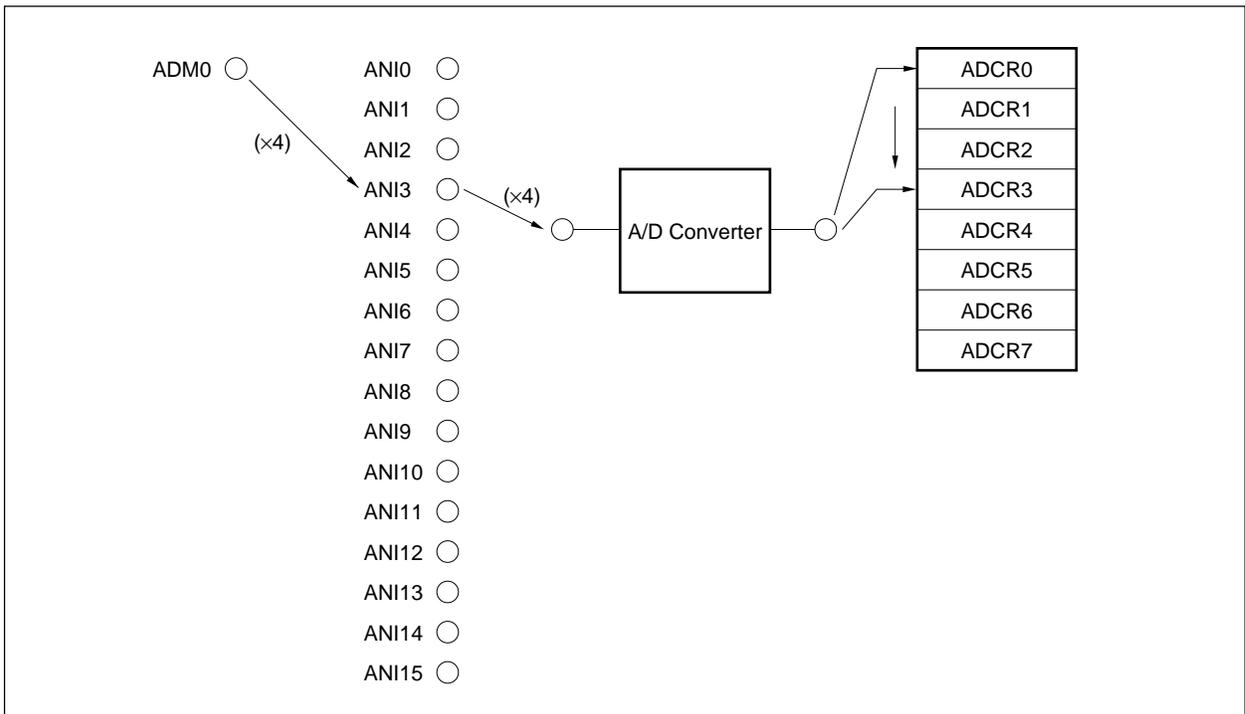
When 1 is written to the CE bit of the ADM0 register, A/D conversion is ended.

This mode is suitable for applications which calculate the average of the A/D conversion result.

★ **Table 9-2. Correspondence between Analog Input Pin and ADCRn Register (4-buffer mode (A/D trigger select 4-buffer))**

Analog Input	A/D Conversion Results Register
ANI0 to ANI3/ANI8 to ANI11	ADCR0 (First time)
	ADCR1 (Second time)
	ADCR2 (Third time)
	ADCR3 (Fourth time)
ANI4 to ANI7/ANI12 to ANI15	ADCR4 (First time)
	ADCR5 (Second time)
	ADCR6 (Third time)
	ADCR7 (Fourth time)

★ **Figure 9-7. Example of 4-Buffer Mode (A/D trigger select 4-buffer) Operation**



**9.5.2 Scan mode operation**

The analog inputs from ANI0/ANI8 to the analog input specified with the ADM0 register are selected sequentially and converted to digital. The A/D conversion results are stored in the ADCRn register corresponding to the analog input. (Refer to **Table 9-3, Figure 9-8.**)

When the conversion of all the specified analog inputs ends, the INTAD interrupt is generated, and A/D conversion is ended.

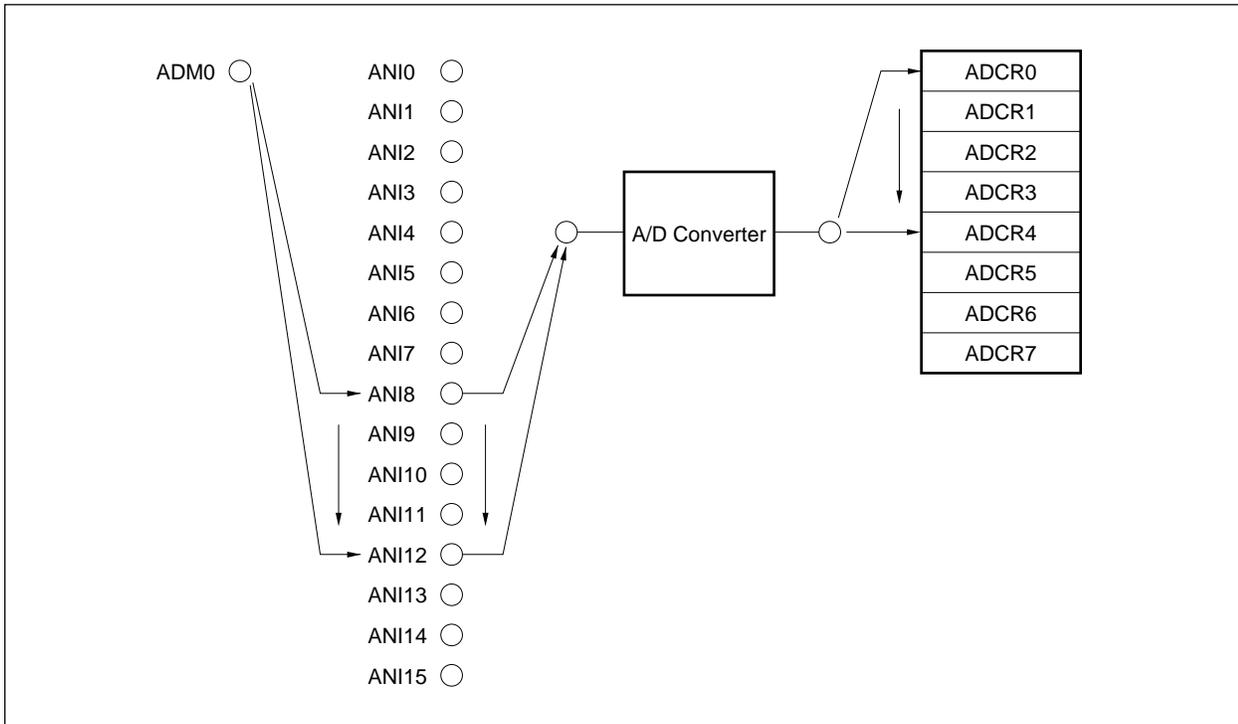
When 1 is written to the CE bit of the ADM0 register, A/D conversion can be restarted.

This mode is suitable for applications which always monitor two or more analog inputs.

★ **Table 9-3. Correspondence between Analog Input Pin and ADCRn Register (scan mode (A/D trigger scan))**

Analog Input	A/D Conversion Results Register
ANI0/ANI8	ADCR0
ANI1/ANI9	ADCR1
ANI2/ANI10	ADCR2
ANI3/ANI11	ADCR3
ANI4/ANI12	ADCR4
ANI5/ANI13	ADCR5
ANI6/ANI14	ADCR6
ANI7/ANI15	ADCR7

★ **Figure 9-8. Example of Scan Mode (A/D trigger scan) Operation**



## 9.6 Operation in the Timer Trigger Mode

The A/D converter can set conversion timings with the coincidence interrupt signals of the RPU compare register. TM0 and the capture/compare register (CC03) are used for the timer for specifying the analog conversion trigger. The following two modes are provided according to the specification of the TMC00 register.

### (1) One-shot mode

To use the one-shot mode, 1 should be set to the OST0 bit of the TMC00 register (one-shot mode).

When the A/D conversion period is longer than the TM0 period, the TM0 generates an overflow, holds 000000H and stops. Thereafter, TM0 does not output the coincidence interrupt signal INTCC03 (A/D conversion trigger) of the compare register, and the A/D converter goes into the A/D conversion standby state. The TM0 count operation restarts when the valid edge of the TCLR0 pin input is detected or when 1 is written to the CE0 bit of the TMC00 register.

### (2) Loop mode

To use the loop mode, 0 should be set to the OST0 bit (normal mode) of the TMC00 register.

When the TM0 generates an overflow, the TM0 starts counting from 000000H again, and the coincidence interrupt signal INTCC03 (A/D conversion trigger) of the compare register is repeatedly output and A/D conversion is also repeated.

Coincidence of the compare register can also clear TM0 and restart it.

### 9.6.1 Select mode operation

The A/D converter converts an analog input (ANI0 to ANI15) specified by the ADM0 register. The conversion results are stored in the ADCRn register corresponding to the analog input. For the select mode, the one-buffer mode and four-buffer mode are provided according to the storing method employed for the A/D conversion results.

#### (1) 1-buffer mode operation (Timer trigger select: 1-buffer)

The A/D converter converts one analog input once and stores the conversion results in one ADCRn register (Refer to **Table 9-4**, **Figure 9-9**).

The A/D converter converts one analog input once using the trigger of the coincidence interrupt signal (INTCC03) and stores the results in one ADCRn register.

An INTAD interrupt is generated for each A/D conversion and the A/D conversion is ended.

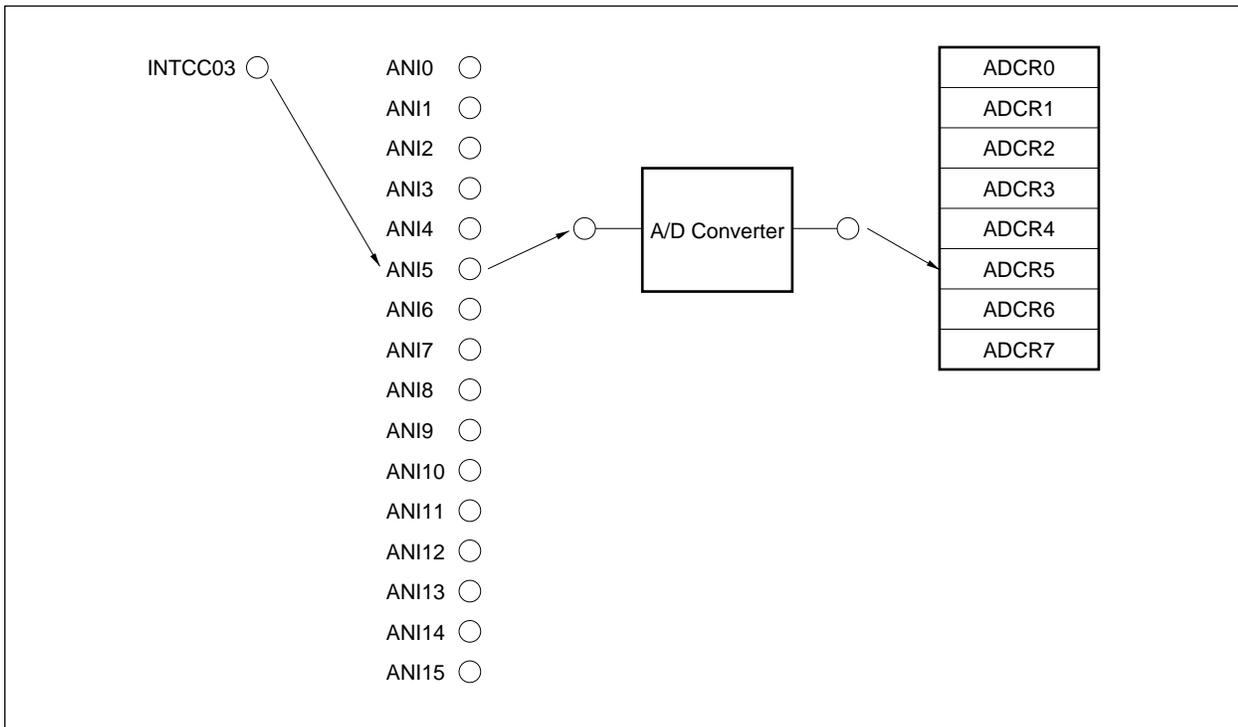
When TM0 is set to the one-shot mode, A/D conversion is ended after one conversion operation. To restart the A/D conversion, input the valid edge to the TCLR0 pin or write 1 to the CE0 bit of the TMC00 register.

When TM0 is set to the loop mode, A/D conversion is repeated each time the coincidence interrupt is generated, unless the CE bit of the ADM0 register is set to 0.

★ **Table 9-4. Correspondence between Analog Input Pin and ADCRn Register (1-buffer mode (timer trigger select 1-buffer))**

Trigger	Analog Input	A/D Conversion Results Register
INTCC03 interrupt	ANI0/ANI8	ADCR0
INTCC03 interrupt	ANI1/ANI9	ADCR1
INTCC03 interrupt	ANI2/ANI10	ADCR2
INTCC03 interrupt	ANI3/ANI11	ADCR3
INTCC03 interrupt	ANI4/ANI12	ADCR4
INTCC03 interrupt	ANI5/ANI13	ADCR5
INTCC03 interrupt	ANI6/ANI14	ADCR6
INTCC03 interrupt	ANI7/ANI15	ADCR7

★ **Figure 9-9. Example of 1-Buffer Mode (timer trigger select 1-buffer) Operation**



**(2) 4-buffer mode operation (Timer trigger select: 4-buffer)**

A/D conversion of one analog input is executed four times, and the results are stored in the ADCRn register (Refer to **Table 9-5, Figure 9-10**).

The A/D converter converts one analog input four times using the coincidence interrupt signal (INTCC03) as a trigger, and stores the results in four ADCRn registers.

An INTAD interrupt is generated when the four A/D conversion operations end, and the A/D conversion is ended.

When the TM0 is set to the one-shot mode, and less than four coincidence interrupts are generated, if the CE bit is set to 1, the INTAD interrupt is not generated and the standby state is set.

This mode is suitable for applications which calculate the average of the A/D conversion result.

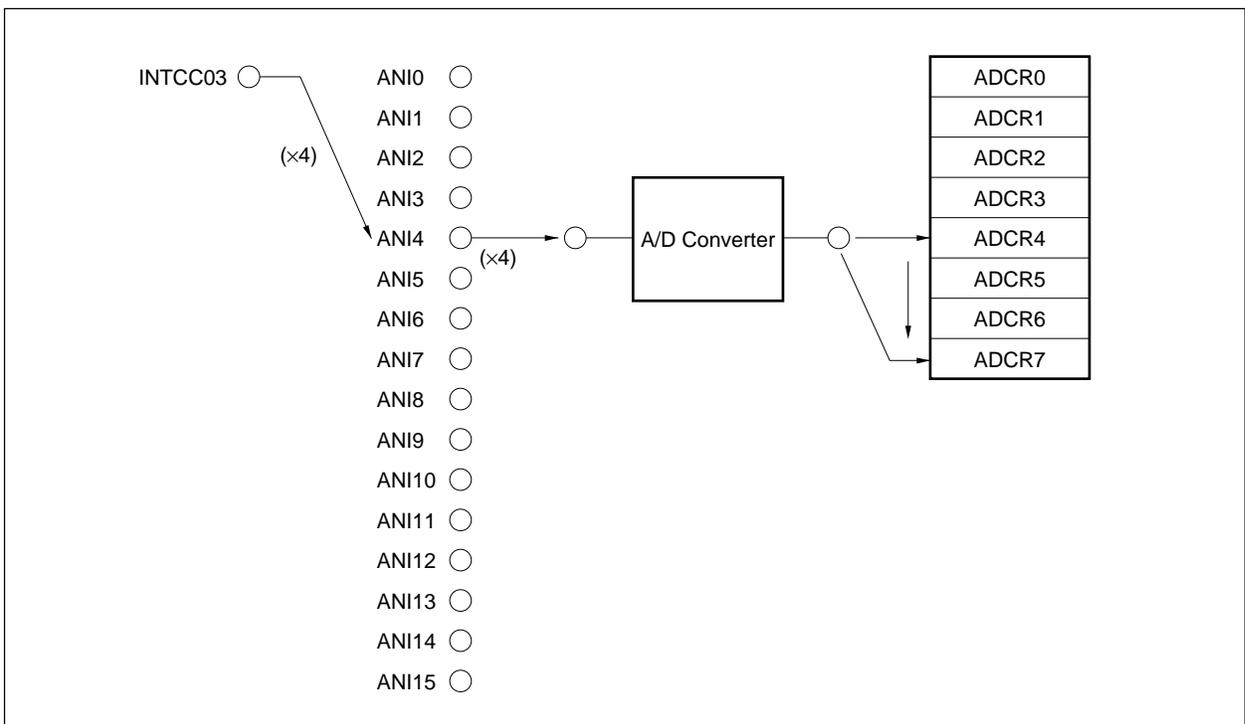
★

**Table 9-5. Correspondence between Analog Input Pin and ADCRn Register (4-buffer mode (timer trigger select 4-buffer))**

Trigger	Analog Input	A/D Conversion Results Register
INTCC03 interrupt	ANI0 to ANI3/ANI8 to ANI11	ADCR0 (First time)
		ADCR1 (Second time)
		ADCR2 (Third time)
		ADCR3 (Fourth time)
INTCC03 interrupt	ANI4 to ANI7/ANI12 to ANI15	ADCR4 (First time)
		ADCR5 (Second time)
		ADCR6 (Third time)
		ADCR7 (Fourth time)

★

**Figure 9-10. Example of Operation in 4-Buffer Mode (timer trigger select 4-buffer)**



**9.6.2 Scan mode operation**

The analog inputs from ANI0/ANI8 to the analog input specified with the ADM0 register are selected sequentially, and A/D conversion is executed the number of times specified using the coincidence interrupt as trigger.

The conversion results are stored in the ADCRn register corresponding to the analog input (refer to **Table 9-6**, **Figure 9-11**). When the conversion of all the specified analog inputs has been ended, the INTAD interrupt is generated and A/D conversion is ended. When the coincidence interrupt is generated after all the specified A/D conversion operations end, A/D conversion is restarted.

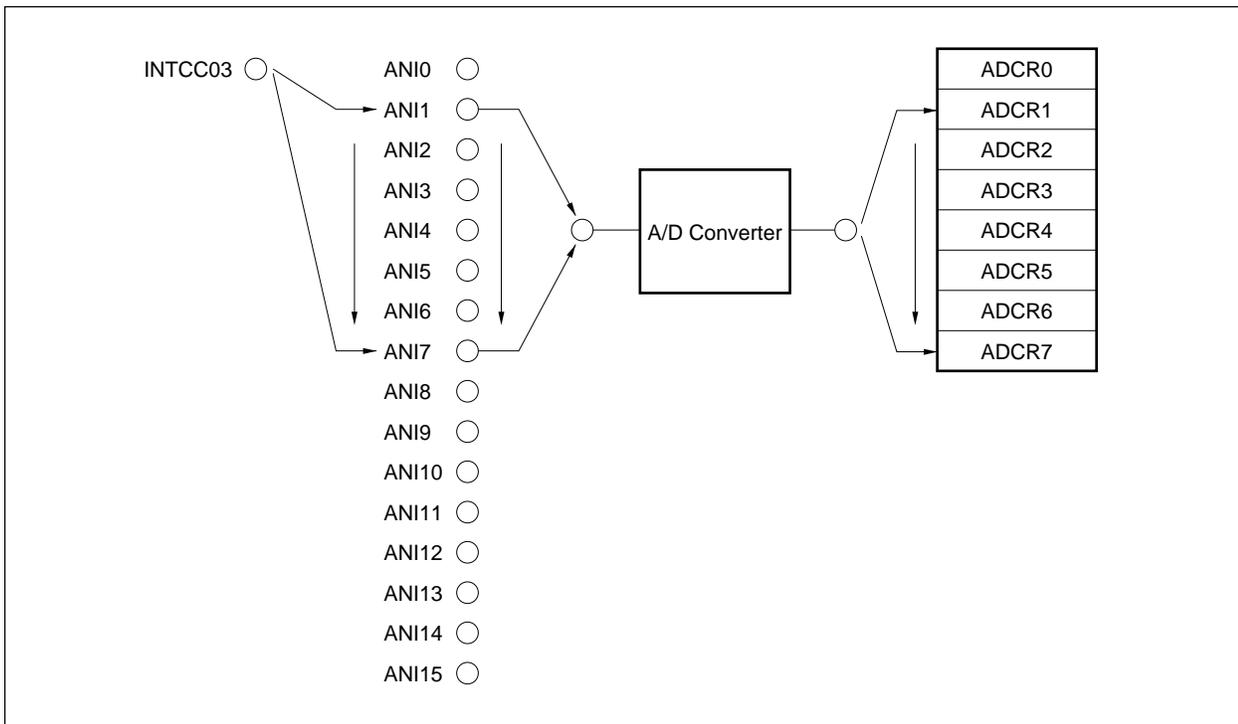
When TM0 is set to the one-shot mode, and less than the specified number of coincidence interrupts are generated the INTAD interrupt is not generated and the standby state is set if the CE bit is set to 1.

This mode is suitable for applications which always monitor two or more analog inputs.

★ **Table 9-6. Correspondence between Analog Input Pin and ADCRn Register (scan mode (timer trigger scan))**

Trigger	Analog Input	A/D Conversion Results Register
INTCC03 interrupt	ANI0/ANI8	ADCR0
INTCC03 interrupt	ANI1/ANI9	ADCR1
INTCC03 interrupt	ANI2/ANI10	ADCR2
INTCC03 interrupt	ANI3/ANI11	ADCR3
INTCC03 interrupt	ANI4/ANI12	ADCR4
INTCC03 interrupt	ANI5/ANI13	ADCR5
INTCC03 interrupt	ANI6/ANI14	ADCR6
INTCC03 interrupt	ANI7/ANI15	ADCR7

★ **Figure 9-11. Example of Scan Mode (timer trigger scan) Operation**



## 9.7 Operation in the External Trigger Mode

In the external trigger mode, the analog inputs (ANI0 to ANI3) are A/D converted by the ADTRG pin input timing.

The ADTRG pin is also used as the P22 pin. To set the external trigger mode, set the PMC22 bit of the PMC2 register to 1 and bits TRG1 to TRG0 of the ADM1 register to 10.

For the valid edge of the external input signal during the external trigger mode, the rising edge, falling edge, or both rising and falling edges can be specified using the ESAD1 and ESAD0 bits of the INTM5 register. For details, refer to **5.3.8 (2) External interrupt mode registers 1 to 6 (INTM1 to INTM6)**.

### 9.7.1 Select mode operation (External trigger select)

The A/D converter converts one analog input (ANI0 to ANI15) specified by the ADM0 register. The conversion results are stored in the ADCRn register corresponding to the analog input. Two select modes, one-buffer mode and four-buffer mode are available for storing the conversion results.

#### (1) 1-buffer mode (External trigger select: 1-buffer)

The A/D converter converts one analog input using the ADTRG signal as a trigger. The conversion results are stored in one ADCRn register (refer to **Table 9-7, Figure 9-12**). The analog input and the A/D conversion results register correspond one to one. An INTAD interrupt is generated after one A/D conversion, and A/D conversion ends.

While the CE bit of the ADM0 register is 1, A/D conversion is repeated every time a trigger is input from the ADTRG pin.

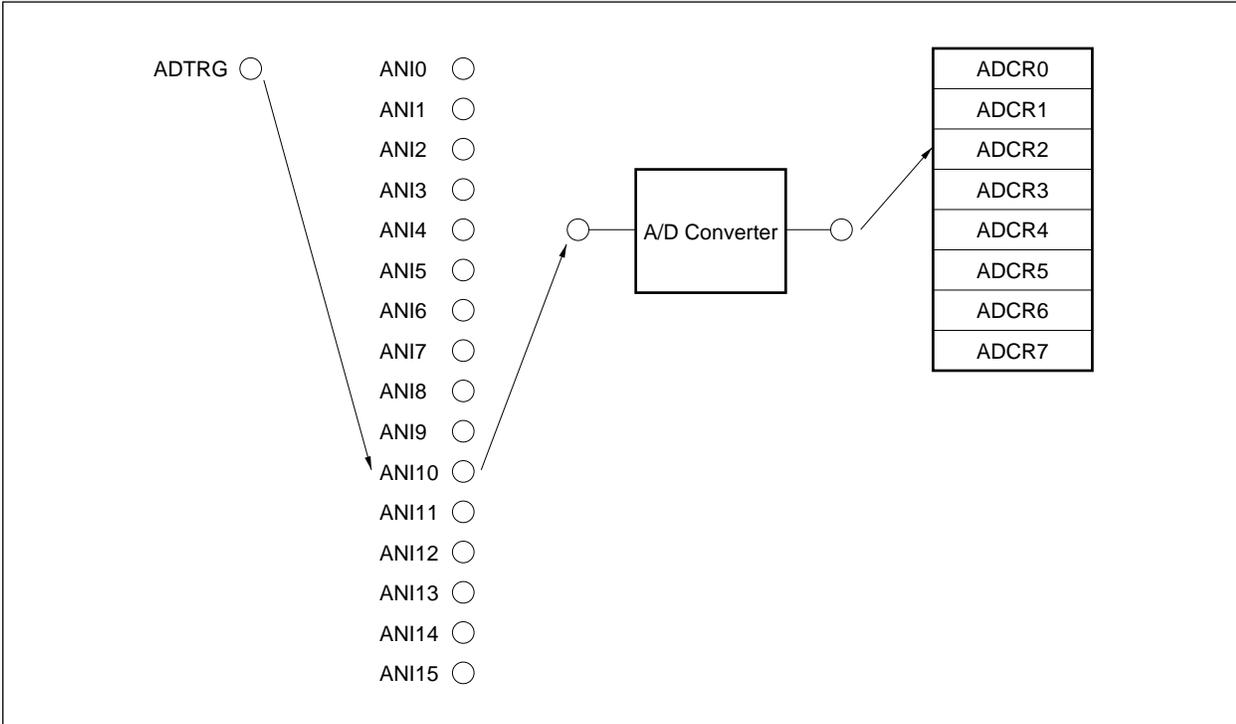
This mode is suitable for applications which read out the result in each A/D conversion.

★

**Table 9-7. Correspondence between Analog Input Pin and ADCRn Register (1-buffer mode (external trigger select 1-buffer))**

Trigger	Analog Input	A/D Conversion Results Register
ADTRG signal	ANI0/ANI8	ADCR0
ADTRG signal	ANI1/ANI9	ADCR1
ADTRG signal	ANI2/ANI10	ADCR2
ADTRG signal	ANI3/ANI11	ADCR3
ADTRG signal	ANI4/ANI12	ADCR4
ADTRG signal	ANI5/ANI13	ADCR5
ADTRG signal	ANI6/ANI14	ADCR6
ADTRG signal	ANI7/ANI15	ADCR7

★ **Figure 9-12. Example of 1-Buffer Mode (external trigger select 1-buffer) Operation**



**(2) 4-buffer mode (External trigger select: 4-buffer)**

The A/D converter converts one analog input four times using the ADTRG signal as a trigger and stores the results in four ADCRn registers (refer to **Table 9-8, Figure 9-13**). The INTAD interrupt is generated and conversion ends when the four A/D conversions end.

While the CE bit of the ADM0 register is 1, A/D conversion is repeated every time a trigger is input from the ADTRG pin.

This mode is suitable for applications which calculate the average of the A/D conversion result.

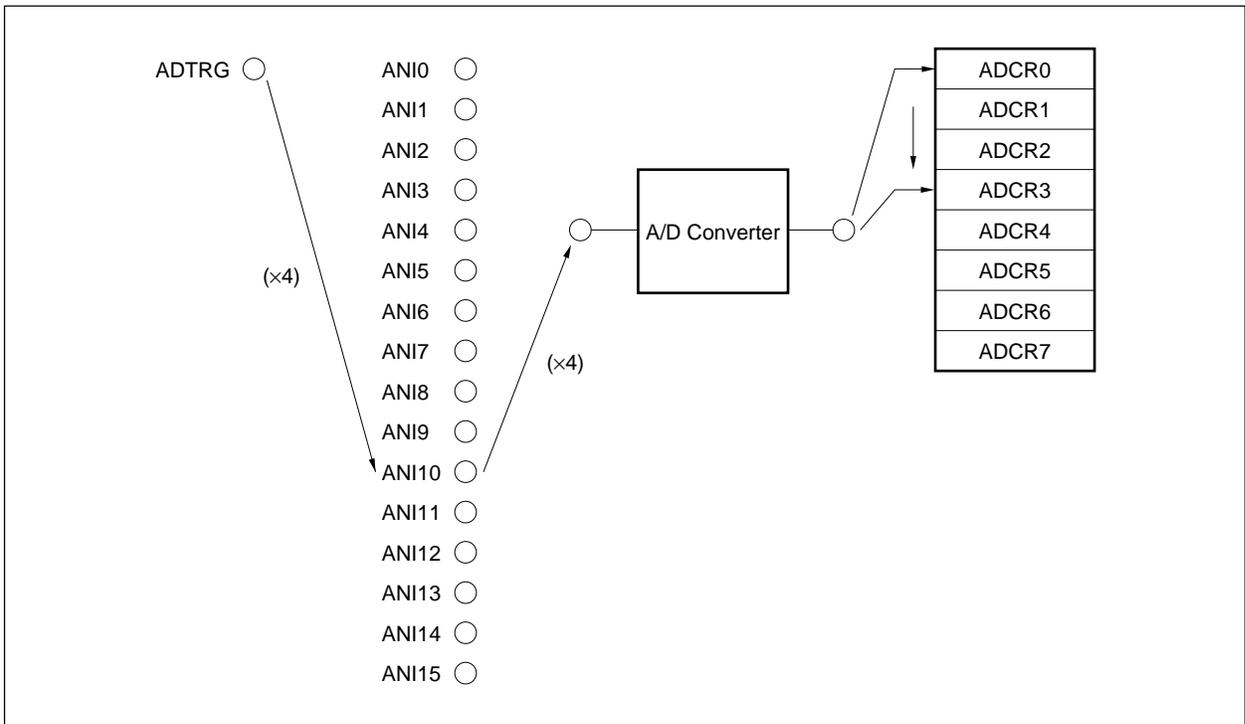
★

**Table 9-8. Correspondence between Analog Input Pin and ADCRn Register (4-buffer mode (external trigger select 4-buffer))**

Trigger	Analog Input	A/D Conversion Results Register
ADTRG signal	ANI0 to ANI3/ANI8 to ANI11	ADCR0 (First time)
		ADCR1 (Second time)
		ADCR2 (Third time)
		ADCR3 (Fourth time)
ADTRG signal	ANI4 to ANI7/ANI12 to ANI15	ADCR4 (First time)
		ADCR5 (Second time)
		ADCR6 (Third time)
		ADCR7 (Fourth time)

★

**Figure 9-13. Example of 4-Buffer Mode (external trigger select 4-buffer) Operation**



**9.7.2 Scan mode operation (External trigger scan)**

The analog inputs from ANI0/ANI8 to the analog input specified with the ADM0 register are selected sequentially and converted to digital when triggered by the ADTRG signal. The A/D conversion results are stored in the ADCRn register corresponding to the analog input (refer to **Table 9-9, Figure 9-14**). When the conversion of all the specified analog inputs ends, the INTAD interrupt is generated and A/D conversion is ended.

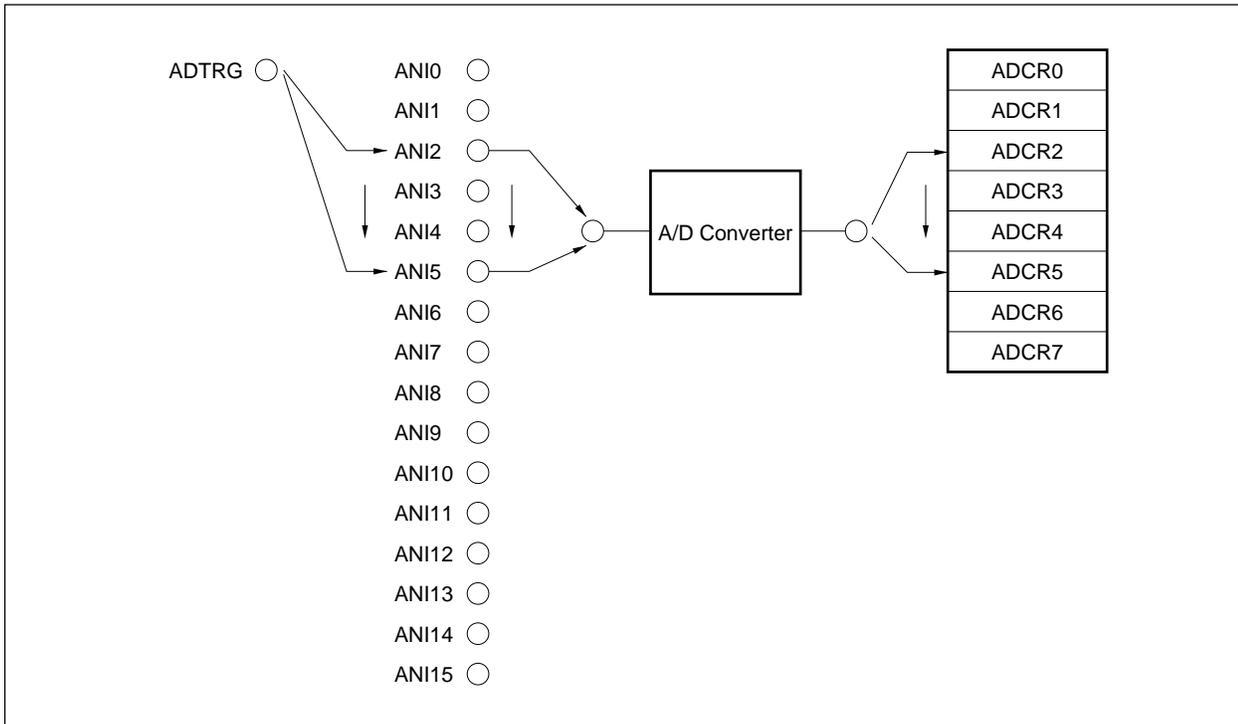
While the CE bit of the ADM0 register is 1, A/D conversion is restarted every time a trigger is input from the ADTRG pin.

This mode is suitable for applications which always monitor two or more analog inputs.

★ **Table 9-9. Correspondence between Analog Input Pin and ADCRn Register (scan mode (external trigger scan))**

Trigger	Analog Input	A/D Conversion Results Register
ADTRG signal	ANI0/ANI8	ADCR0
ADTRG signal	ANI1/ANI9	ADCR1
ADTRG signal	ANI2/ANI10	ADCR2
ADTRG signal	ANI3/ANI11	ADCR3
ADTRG signal	ANI4/ANI12	ADCR4
ADTRG signal	ANI5/ANI13	ADCR5
ADTRG signal	ANI6/ANI14	ADCR6
ADTRG signal	ANI7/ANI15	ADCR7

★ **Figure 9-14. Example of Scan Mode (external trigger scan) Operation**



## 9.8 Precautions Regarding Operations

### 9.8.1 Stop of conversion operations

When 0 is written to the CE bit of the ADM0 register during conversion, conversion stops and the conversion results are not stored in the ADCRn register (n = 0 to 7).

### ★ 9.8.2 Interval of the external/timer trigger

Set the interval (input time interval) of the trigger during the external or timer trigger mode longer than the conversion time specified by the FR2 to FR0 bits of the ADM1 register.

#### When $0 < \text{interval} \leq \text{conversion operation time}$

When the next external trigger or timer trigger is input during conversion, conversion stops and conversion starts according to the last timer trigger input.

When conversion operations are stopped, the conversion results are not stored in the ADCRn register (n = 0 to 7). However, the number of triggers input are counted, and when an interrupt is generated, the value at which conversion ended is stored in the ADCRn register.

### 9.8.3 Operation in the standby mode

#### (1) HALT mode

Continues A/D conversion operations. When canceled by NMI input, the ADM0/ADM1 register and ADCRn register hold the value (n = 0 to 7).

#### (2) IDLE mode, STOP mode

As clock supply to the A/D converter is stopped, no conversion operations are performed. When canceled using NMI input, the ADM0/ADM1 register and the ADCRn register hold the value (n = 0 to 7). However, when these modes are set during conversion, conversion stops. At this time, if canceled using the NMI input, the conversion operation resumes, but the conversion result written to the ADCRn register will become undefined. In the IDLE and STOP modes, operation of the serial resistor string is also stopped to reduce the power consumption. To further reduce current consumption, set the voltage of the AVREF to VSS.

### 9.8.4 Compare coincide interrupt in the timer trigger mode

The coincidence interrupt of the compare register becomes the A/D conversion start trigger and conversion operations are started. At this time, the coincidence interrupt of the compare register also functions as the coincidence interrupt of the compare register for the CPU. To prevent generation of the coincidence interrupt of the compare register for the CPU, set interrupt disable using the interrupt mask bit (CC0MK3) of the interrupt control register (CC0IC3).

**[MEMO]**

## CHAPTER 10 REAL-TIME OUTPUT FUNCTION

### 10.1 Configuration and Function

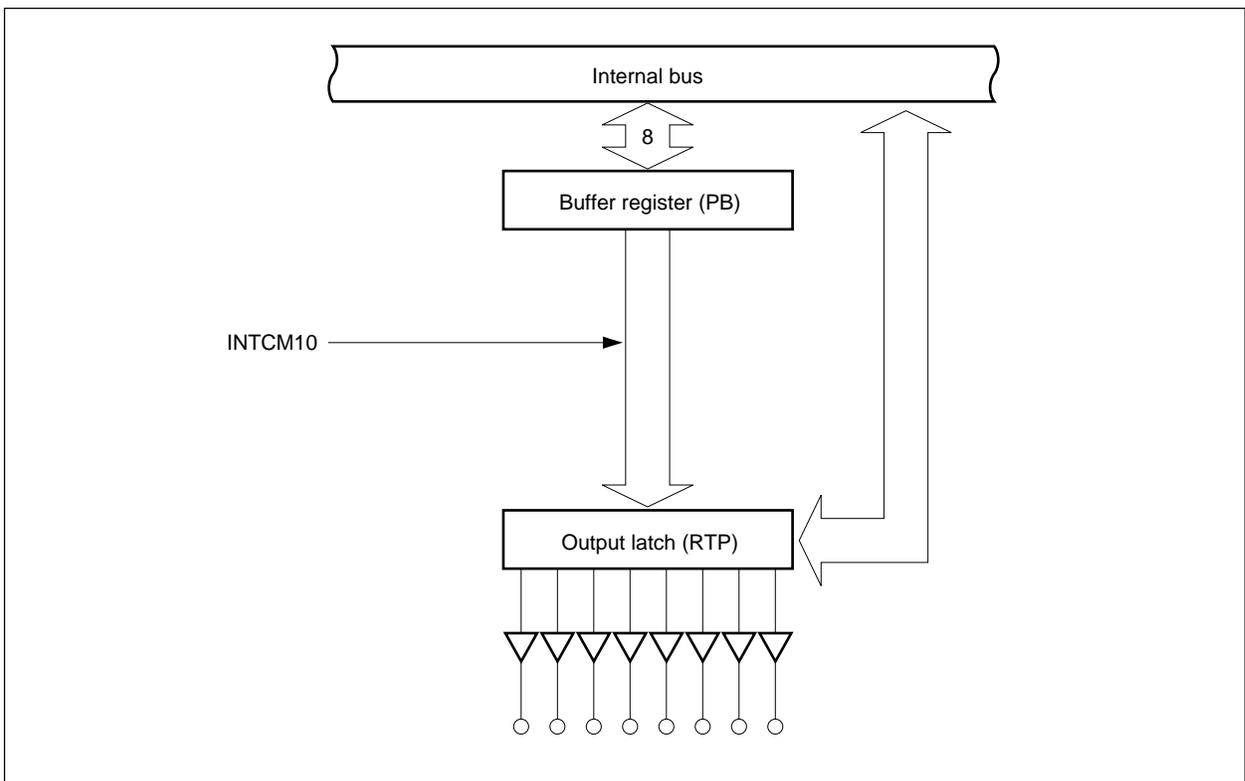
The real-time output function is realized by hardware consisting principally of the buffer register (PB) and the output latch (RTP) as shown in Figure 10-1.

The real-time output function is a procedure that transfers the data previously prepared in the PB register to the output latch by hardware simultaneously with the generation of CM10 coincidence interrupt of timer 1, and outputs it to external. The pins to output the data to external are called a real time output port.

The real-time output function can handle 8-bit real-time output data.

Figure 10-1 shows the block diagram of the real-time output port.

**Figure 10-1. Block Diagram of Real-Time Output Port**

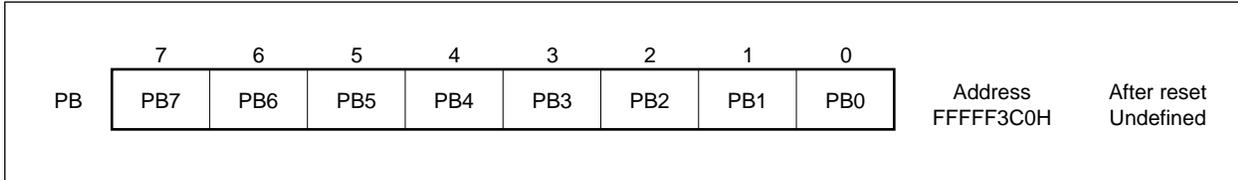


## 10.2 Control Register

### (1) Buffer register (PB)

The buffer register is a register to which the data to be output from the real-time output port is written beforehand.

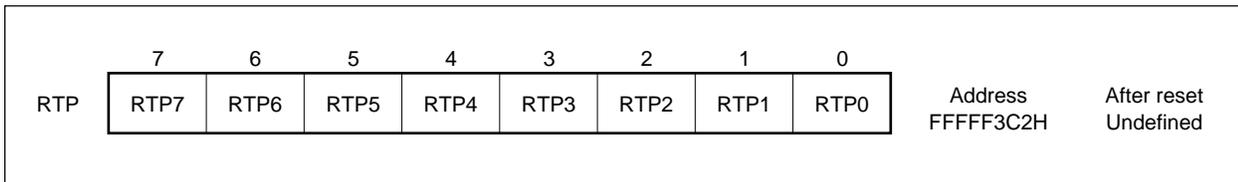
This register can be read/written in 8- or 1-bit units.



### (2) Output latch (RTP)

The output latch is a register to which the data of the PB register is transferred by the coincidence signal with the CM10 register. Write the value to be output to external to the RTP register before setting the port as a real-time output port with the PMC13 register.

This register can be read/written in 8- or 1-bit units.



## 10.3 Operation

When the corresponding bit is set to the control mode by the PMC13 register, the output pins can be used as a real-time output port. These pins can be accessed by the PB register and the RTP register.

The data is output by the following procedure.

- <1> The data is written to the PB register.
- <2> The contents of the PB register is transferred to the RTP register at the generation timing of the compare coincidence interrupt of timer (INTCM10).
- <3> The contents of the RTP register is output to the pins set as the real time output port.

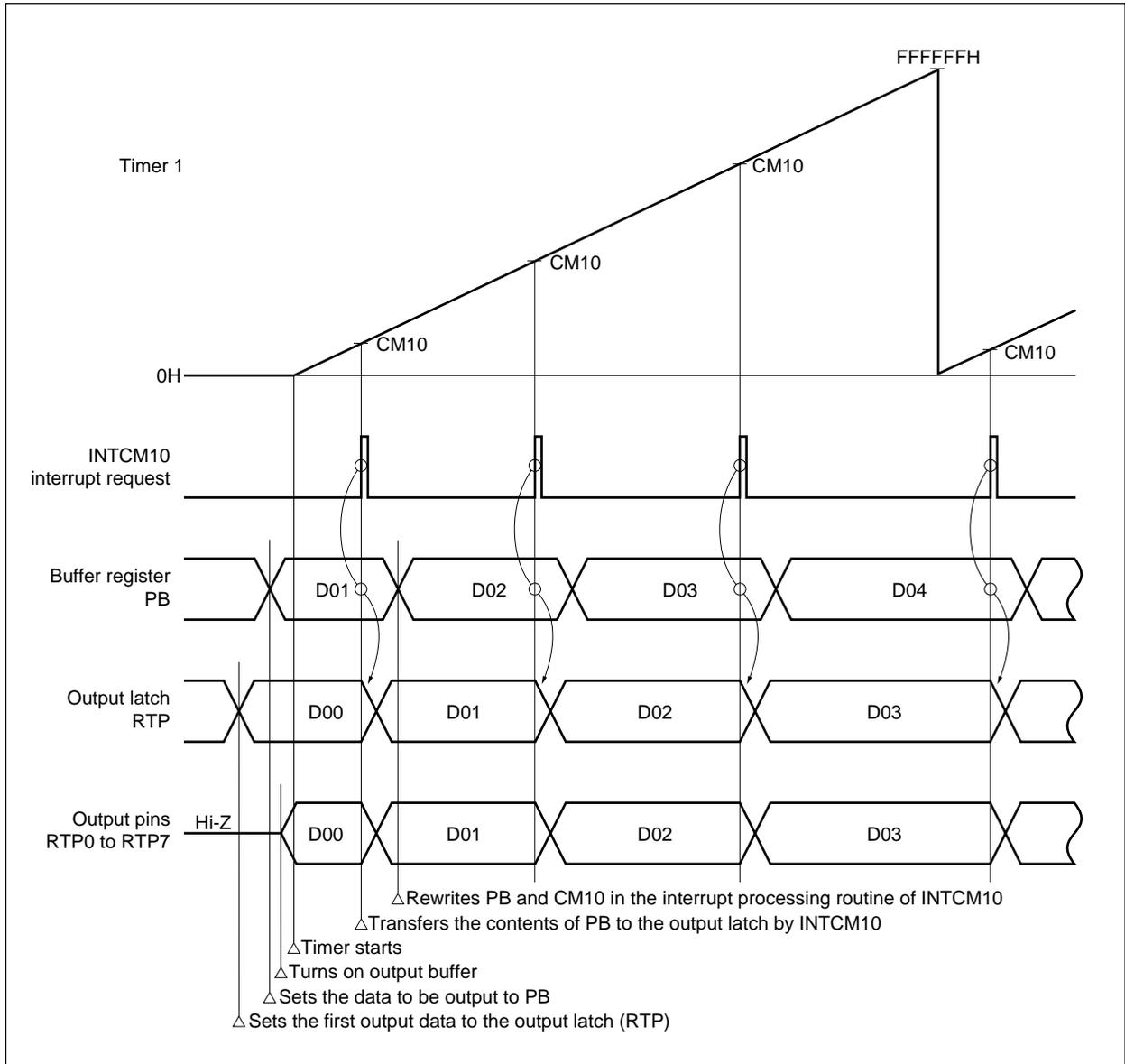
### 10.4 Example

The following shows an example of using RTP0 to RTP7 as an 8-bit real-time output port.

The contents of the buffer register (PB) is output to RTP0 to RTP7 in each coincidence of the contents of TM1 and CM10 of timer 1. At this time, the interrupt processing routine is generated in which the data to be output next and the timing to change the output next are set (refer to **Figure 10-2**).

For the use of timer 1, refer to **7.2.2 Timer 1**.

**Figure 10-2. Operational Timings of Real-Time Output Port**



**[MEMO]**

## CHAPTER 11 PWM UNIT

### 11.1 Features

- PWMn: 4 channels
- 12- to 16-bit PWM output port
- Main pulse + additional pulse configuration
  - Main pulse 4/5/6/7/8 bits
  - Additional pulse 8 bits
- Repeat frequency: 129 kHz to 2 MHz ( $f_{\text{PWMC}} = 33 \text{ MHz}$ )
- Pulse width overwrite frequency selection: each one pulse/256 pulse
- Active level of the PWM output pulse can be selected.
- Operation clock: Can be selected from  $\phi$ ,  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ , and  $\phi/16$ . ( $\phi$  is the internal system clock)

**Remark** n = 0 to 3

## 11.2 Configuration

Figure 11-1 shows the configuration of the output circuit of PWMn.

### (1) Prescaler

Divides the frequency of  $\phi$  and generates PWM operational clock ( $f_{PWM}$ ). Prescaler output is selected by the PWPn0/PWPn1 bit of the PWPRn register.

### (2) Reload control

Controls the reload of the modulo values of x-bit down counter and the 8-bit counter.

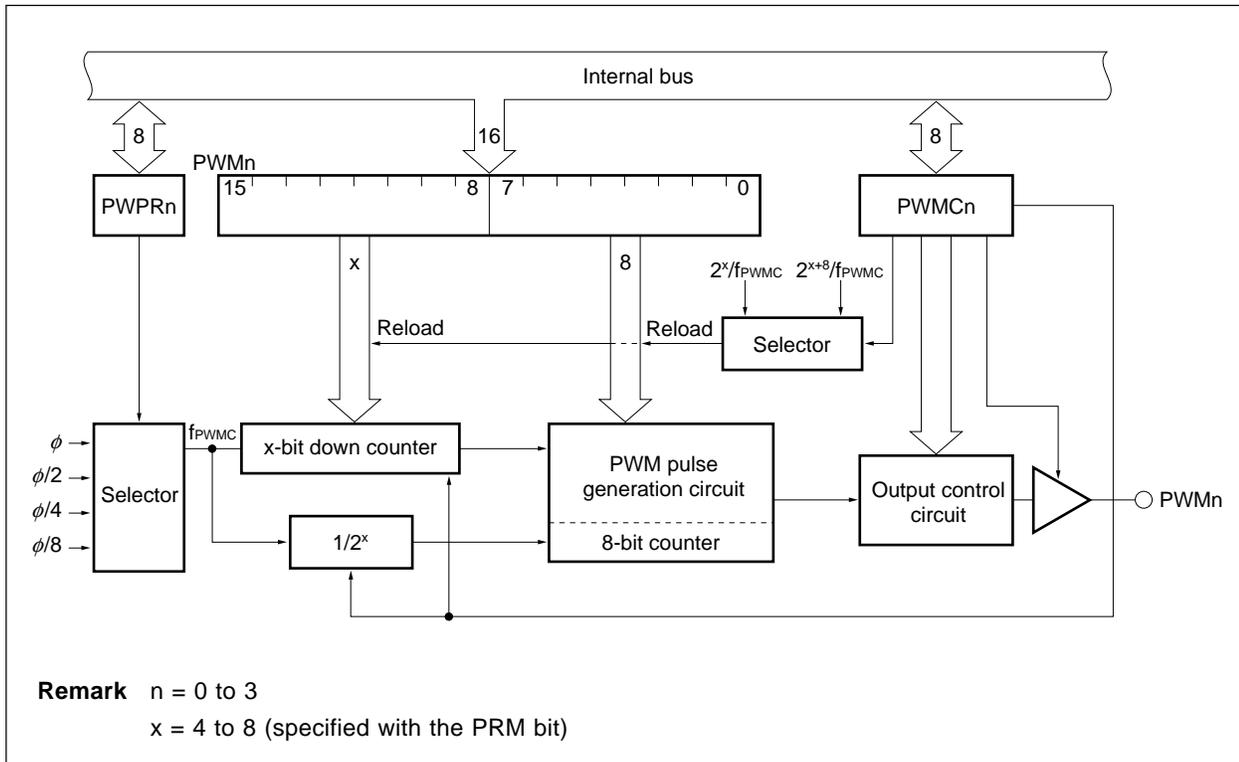
$2^x/f_{PWM}$  or  $2^{x+8}/f_{PWM}$  is selected by the SYNn bit of the PWMCn register for the reload timing (PWM pulse width rewrite cycle).

### (3) x-bit down counter

Controls the output timings of the main pulse.

The value of the modulo H register is loaded to this counter by the reload signal generated in the reload controls and decremented by PWM operational clock ( $f_{PWM}$ ).

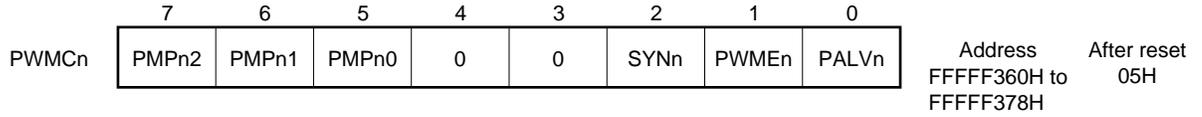
Figure 11-1. Configuration of PWM Unit



### 11.3 Control Register

#### (1) PWM control register 0 to 3 (PWMC0 to PWMC3)

Controls PWMn operation, specifies the output active level, and specifies the bit length of the main pulse. This register can be read/written in 8- or 1-bit units. The contents of this register can also be changed during PWMn operation (PWME = 1).

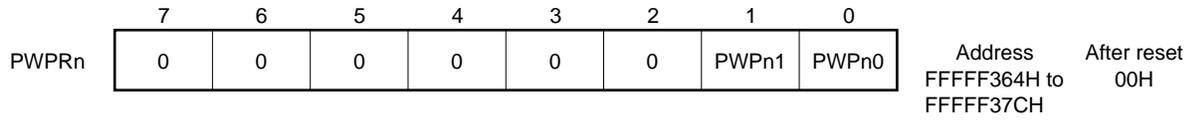


Bit Position	Bit Name	Function																												
7 to 5	PMPn2 to PMPn0	PWM Main Pulse Specifies bit length of x-bit down counter (main pulse) <table border="1" style="border-collapse: collapse; width: 100%; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">PMPn2</th> <th style="width: 10%;">PMPn1</th> <th style="width: 10%;">PMPn0</th> <th style="width: 70%;">Bit Length</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>8 bits</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>7 bits</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>6 bits</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>5 bits</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>4 bits</td> </tr> <tr> <td colspan="3" style="text-align: center;">Others</td> <td style="text-align: center;">Setting prohibited</td> </tr> </tbody> </table>	PMPn2	PMPn1	PMPn0	Bit Length	0	0	0	8 bits	0	0	1	7 bits	0	1	0	6 bits	0	1	1	5 bits	1	0	0	4 bits	Others			Setting prohibited
PMPn2	PMPn1	PMPn0	Bit Length																											
0	0	0	8 bits																											
0	0	1	7 bits																											
0	1	0	6 bits																											
0	1	1	5 bits																											
1	0	0	4 bits																											
Others			Setting prohibited																											
2	SYNn	Rewrite Cycle Specifies PWM pulse width rewrite cycle 0: Large cycle (every PWM $2^{x+8}$ cycles ( $2^{x+8}/f_{PWM}$ )) 1: Small cycle (every PWM 1 cycle ( $2^x/f_{PWM}$ ))																												
1	PWME n	PWM Enable Controls operation/stop of PWMn 0: Operation stops 1: Operation in progress PWM counter is cleared by changing this bit from 0 to 1.																												
0	PALVn	PWM Active Level Specifies PWMn active level 0: Active low 1: Active high																												

**Remark** n = 0 to 3  
 x: number of bits set with PRM

**(2) PWM prescaler register 0 to 3 (PWPR0 to PWPR3)**

This register selects the operation clock ( $f_{PWM}$ ) of PWMn, and can be read/written in 8- or 1-bit units. Change the contents of this register while the bits of the PWMCn register are 0. If the contents of this register are changed when the setting of the PWME<sub>n</sub> bit is 1, the operation cannot be guaranteed.



Bit Position	Bit Name	Function															
1, 0	PWP <sub>n1</sub> , PWP <sub>n0</sub>	PWM Prescaler Clock Mode Selects operation clock of PWMn <table border="1" style="margin: 10px auto; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">PWP<sub>n1</sub></th> <th style="width: 10%;">PWP<sub>n0</sub></th> <th style="width: 80%;">Operation Clock (<math>f_{PWM}</math>)</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>\phi</math></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>\phi/2</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;"><math>\phi/4</math></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;"><math>\phi/8</math></td> </tr> </tbody> </table> <p><math>\phi</math>: Internal system clock</p>	PWP <sub>n1</sub>	PWP <sub>n0</sub>	Operation Clock ( $f_{PWM}$ )	0	0	$\phi$	0	1	$\phi/2$	1	0	$\phi/4$	1	1	$\phi/8$
PWP <sub>n1</sub>	PWP <sub>n0</sub>	Operation Clock ( $f_{PWM}$ )															
0	0	$\phi$															
0	1	$\phi/2$															
1	0	$\phi/4$															
1	1	$\phi/8$															

**Remark** n = 0 to 3

**(3) PWM modulo registers 0 to 3 (PWM0 to PWM3)**

The PWM modulo registers 0 to 3 are 16-bit registers used to determine the pulse width of the PWM<sub>n</sub> pulse. These registers can be read/written in 16-bit units. These registers consist of the following two parts.

**<1> Modulo H register (bit 8 to bit 15)**

Indicates the number of bits specified with the PMP<sub>n</sub> bit of the PWM<sub>Cn</sub> register. This value is the accuracy when generating the main pulse.

The pulse width rewrite timing depends on the number of bits of this register.

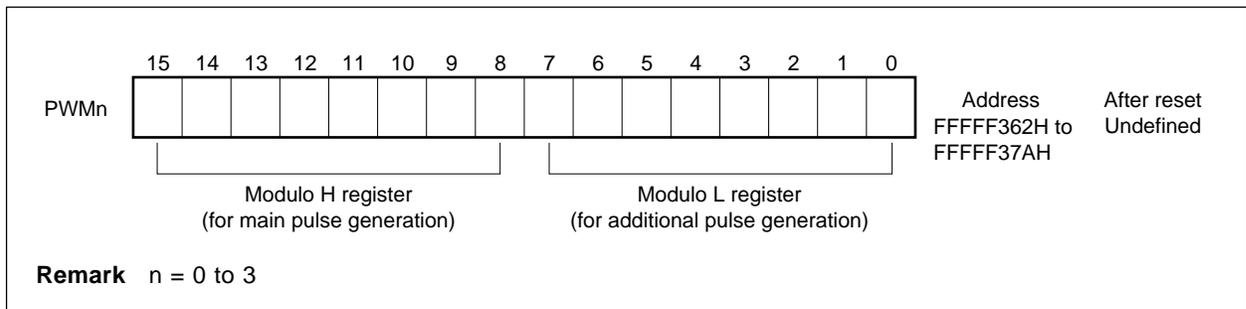
When selecting 4 to 7 bits for the counter with the PMP<sub>n</sub> bit, set 0 to the rest of the higher bits.

**<2> Modulo L register (bit 0 to bit 7)**

The value of this register determines the additional timings of the additional pulse to perform minute adjustment (refer to **Figure 11-3**).

The value of this register becomes undefined by  $\overline{\text{RESET}}$  input. Set the data with initialization program before enabling PWM output.

The value from 0000H to FFFFH can be set to the PWM<sub>n</sub> register, and PWM output also changes linearly. When 0000H is set, inactive level is retained. When FFFFH is set, one additional pulse ( $1/f_{\text{PWMC}}$ ) becomes inactive by one rewrite cycle ( $2^{16}/f_{\text{PWMC}}$ ) (refer to **Figure 11-4**).



## 11.4 PWM Operations

### 11.4.1 Basic operations of PWM

The duty of the PWM pulse output is determined as follows by the value set to the modulo H register of the PWM modulo register (PWMn: n = 0 to 3 ).

$$\text{Duty of PWM pulse output} = \frac{(\text{Value of modulo H register})^{\text{Note 1}} + 1^{\text{Note 2}}}{2^x}$$

- Notes**
1.  $0 \leq (\text{Value of modulo H register}) \leq 2^x - 1$
  2. With additional pulse

**Remark**  $x = 4$  to  $8$

The repeat frequency of the PWM pulse output is the frequency of the  $2^x$  frequency division ( $= f_{\text{PWMC}}/2^x$ ) of the PWM clock ( $f_{\text{PWMC}}$ ) of  $\phi$  to  $\phi/4$  set by the PWM prescaler register (PWPR), and the minimum pulse width is  $1/f_{\text{PWMC}}$ .

The PWM pulse output realizes 12- to 16-bit resolution by repeatedly outputting the PWM signal with 4- to 8-bit resolution and  $f_{\text{PWMC}}/2^x$  repeat frequency for 256 times. The PWM pulse signal with 12- to 16-bit resolution is realized in 256 cycles by controlling the addition of the additional pulse ( $1/f_{\text{PWMC}}$ ) to the PWM pulse with 4- to 8-bit resolution determined by the modulo H register according to the value of the modulo L register in 1-cycle units.

**Figure 11-2. Basic Operations of PWM**

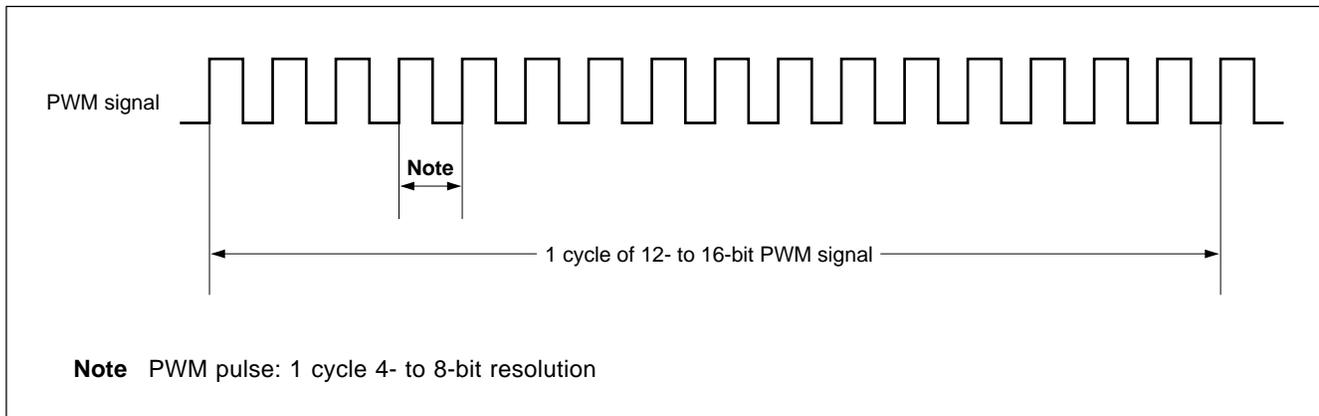


Figure 11-3. Example of PWM Output by Main Pulse and Additional Pulse

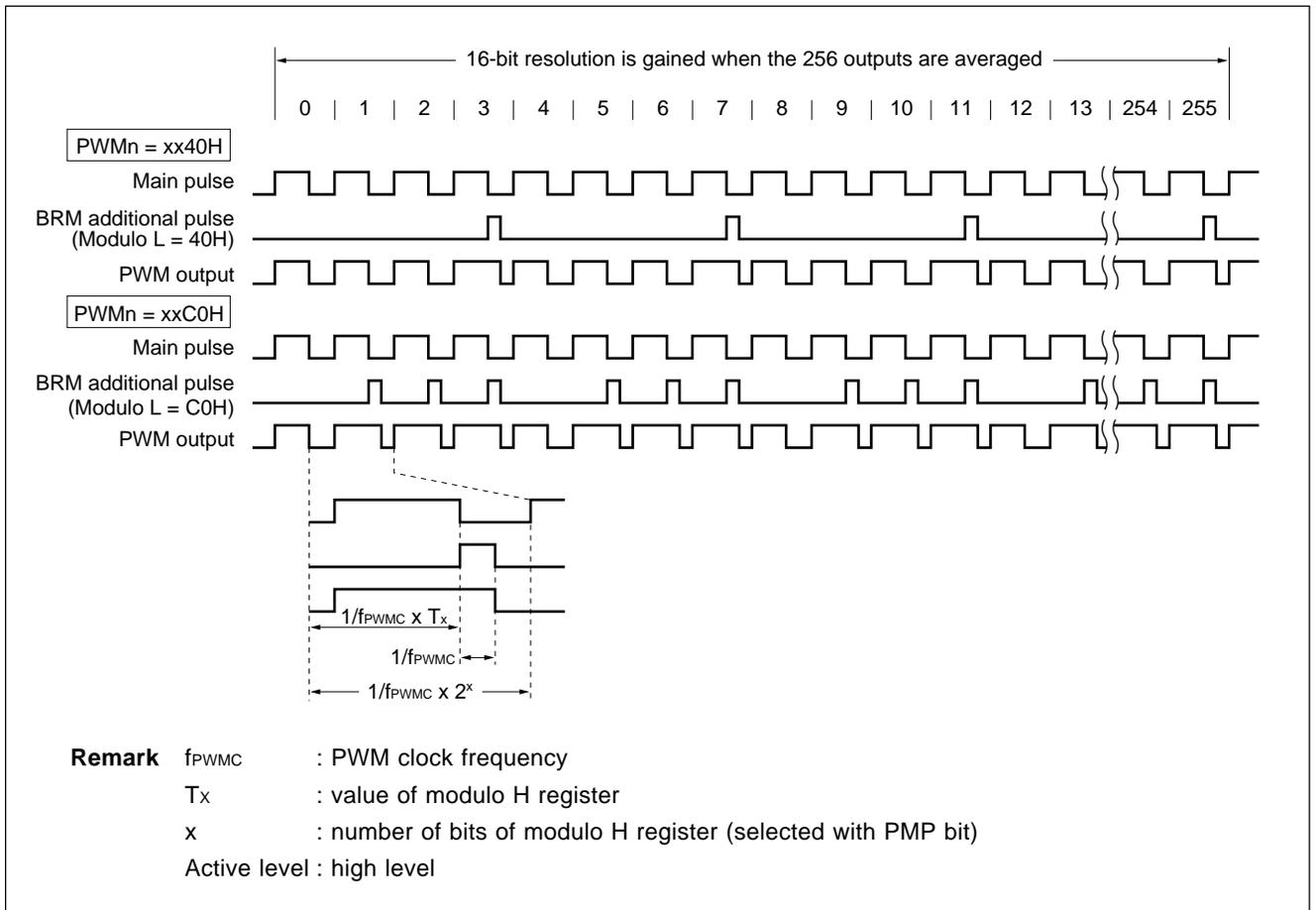
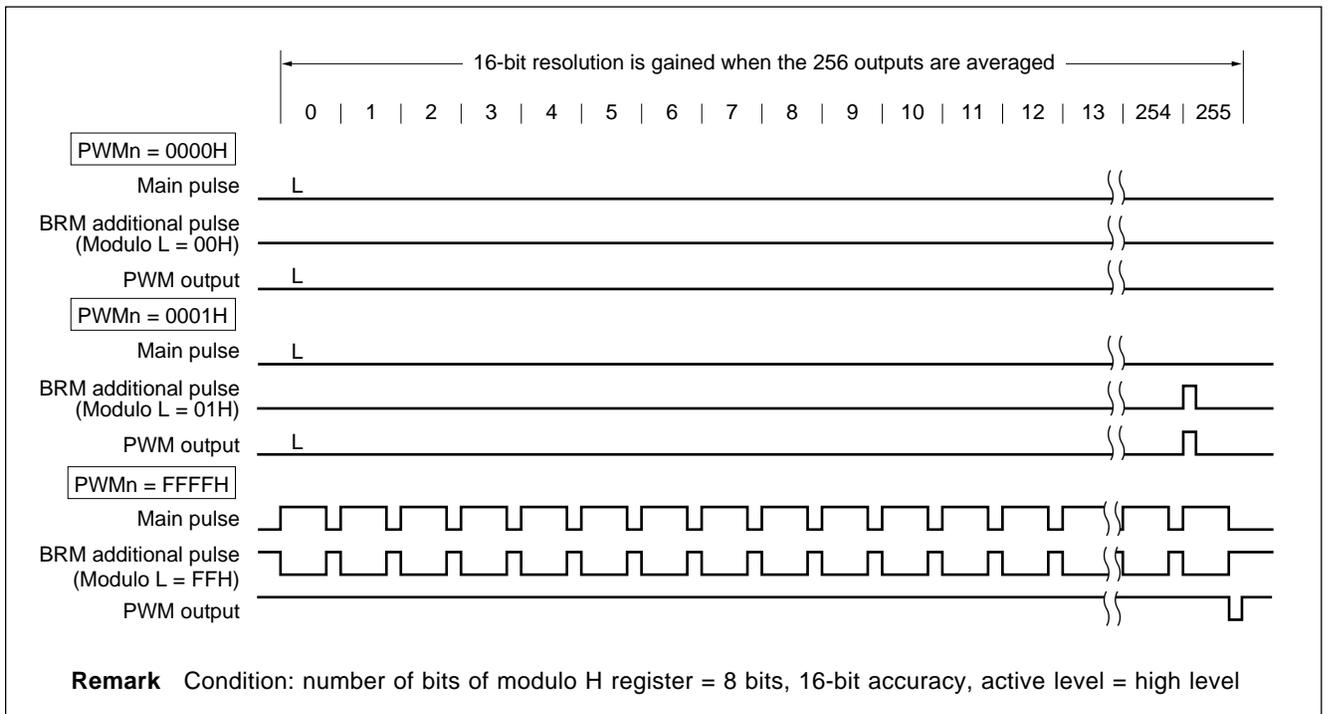


Figure 11-4. Example of PWM Output Operation



### 11.4.2 Enabling/disabling PWM operation

To output the PWM pulse, the PWME bit of the PWM control register (PWMCn) is set (1) after setting data to the PWM prescaler register (PWPRn) and the PWM modulo register (PWMn) (n = 0 to 3).

Thereby, PWM pulse with active level specified by the PALVn bit of the PWMCn register is output from the PWM output pin.

When the PWME bit of the PWMCn register is cleared (0), the PWM output unit immediately stops the PWM output operation, and the PWM output pin becomes inactive.

#### (1) Setting when PWM operation starts

When the PWME bit of the PWMCn register is set, PWMn goes into operation status. However, the PWM pin maintains the port mode status even after the operation status is set until the reload signal of the PWMn register is generated. In addition, the value of the PWMn register is not loaded to the x-bit down counter. Therefore, when the pulse width rewrite timing is set to  $2^{x+8}$  (large cycle: SYN bit = 0), operation starts in  $2^{x+8}/f_{PWM}$  max. after the PWME bit is set. The SYNn bit of the PWMCn register can be rewritten even during PWM output.

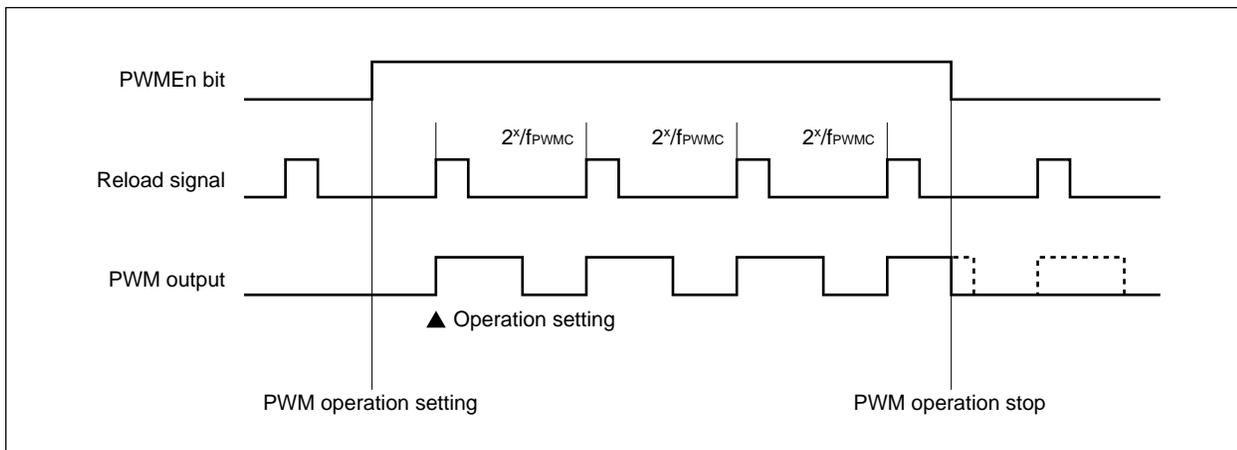
Initialize the following registers before starting PWMn operation.

- PMC10 register : Setting of the control mode
- PWMn register : Setting of the pulse width
- PWPRn register : Specification of the operational clock of the PWM output circuit
- PWMCn register : Specification of the PWM pulse width rewrite cycle, specification of active level of PWM pin, PWM operation control, and selection of the number of bits of the main pulse

#### (2) Setting when PWM operation stops

When the PWME bit of the PWMCn register is reset, PWM operation immediately stops.

Figure 11-5. Operation Timing of PWM



### 11.4.3 Specification of active level of PWM pulse

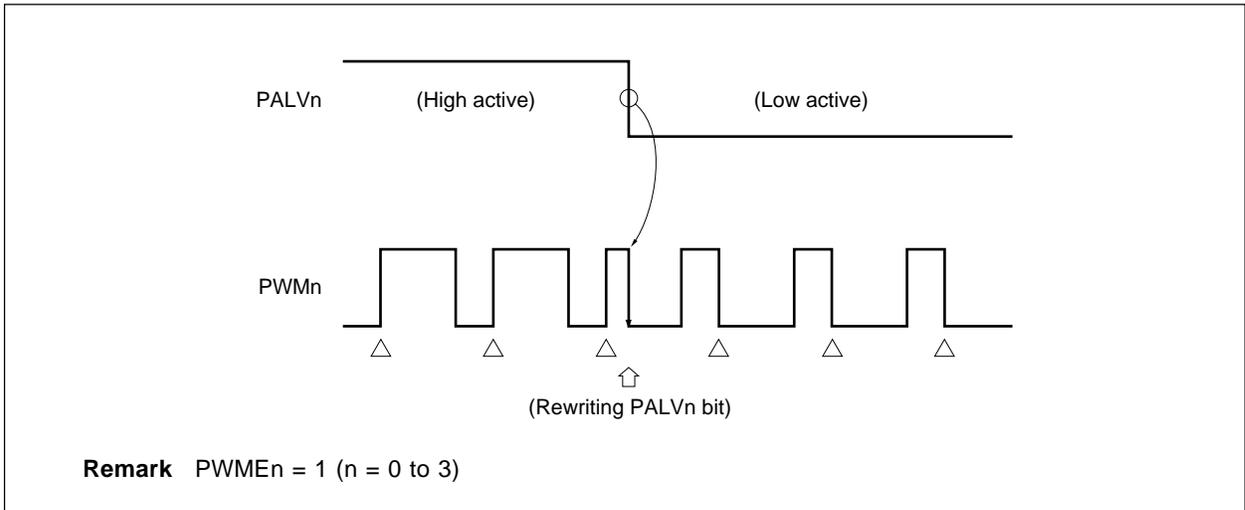
The PALVn bit of the PWM control register (PWMCn) specifies the active level of the PWM pulse output from the PWM output pin (n = 0 to 3).

When the PALVn bit is set (1), a pulse with high active level is output, and when it is cleared (0), a pulse with low active level is output.

When the PALVn bit is rewritten, the active level of the PWM output immediately changes. Figure 11-6 shows the active level setting and the pin status of the PWM output.

The active level of the PWM output can be changed by manipulating the PALVn bit, regardless of the setting of the PWME n bit (enabling/disabling PWM).

**Figure 11-6. Setting of Active Level of PWM Output**



### 11.4.4 Specification of PWM pulse width rewrite cycle

Starting PWM output and changing the PWM pulse width are performed in synchronization either with each  $2^{(x+8)}$  cycles ( $2^{(x+8)}/f_{PWM}$ ) of the PWM pulse or with each 1 cycle ( $2^x/f_{PWM}$ ) of the PWM pulse. The specification of the PWM pulse width rewrite cycle is performed with the SYNn bit of the PWMCn register (n = 0 to 3).

When the SYNn bit is cleared (0), the pulse width is changed at every  $2^{(x+8)}$  cycles ( $2^{(x+8)}/f_{PWM}$ ) of the PWM pulse. Therefore, it will take  $2^{(x+8)}$  clocks max. before the pulse with the width corresponding to the data written to the PWMn register is output.

Figure 11-7 shows an example of the PWM output timing.

On the other hand, when the SYNn bit is set (1), the pulse width is changed at every 1 cycle of the PWM pulse ( $2^x/f_{PWM}$ ). In this case, it will take  $2^x$  clocks max. before the pulse with the width corresponding to the data written to the PWMn register is output.

When the PWM pulse rewriting cycle is specified as every  $2^x/f_{PWM}$  (when the SYNn bit is set (1)), the accuracy of the PWM pulse gained is x bits or more and (x+8) bits or less, which is lower than the accuracy when the rewriting cycle is specified as  $2^{(x+8)}/f_{PWM}$ . However, the response is improved because the repeat frequency is increased.

Figure 11-8 shows an example of the PWM output timing when the rewrite timing is  $2^x/f_{PWM}$ .

Figure 11-7. Example 1 of PWM Output Timing (PWM pulse width rewrite cycle  $2^{(x+8)}/f_{PWM}$ )

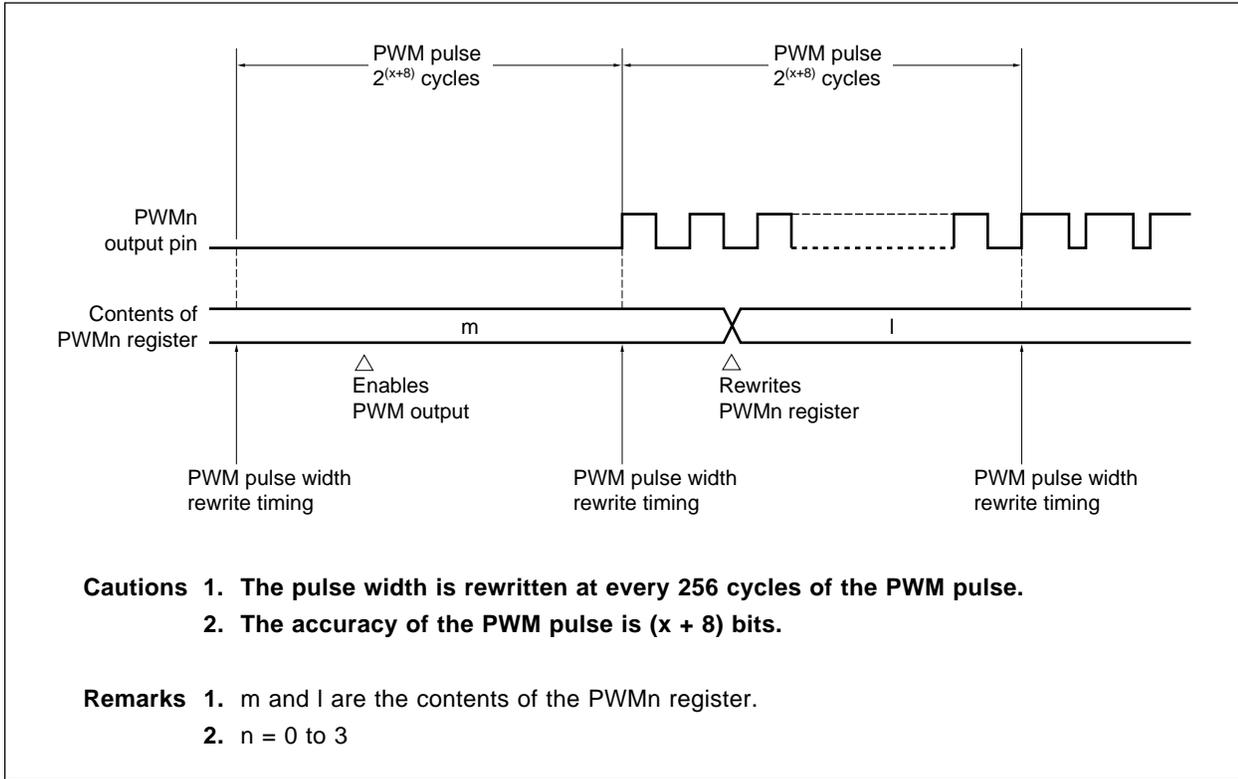
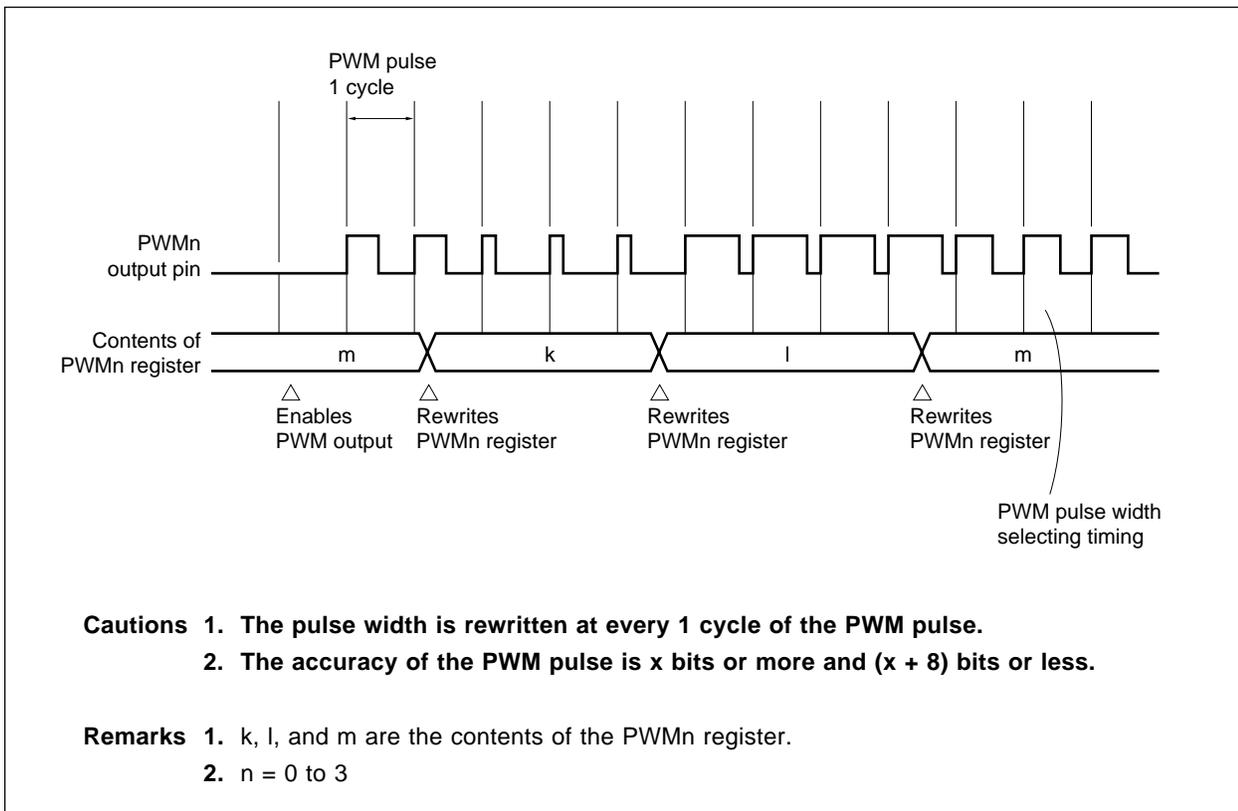


Figure 11-8. Example 2 of PWM Output Timing (PWM pulse width rewrite cycle  $2^x/f_{PWM}$ )



**11.4.5 Repetition frequency**

The repetition frequency of the PWMn is shown below (n = 0 to 3).

Main Pulse	Additional Pulse	Repetition Frequency	Pulse Width Rewrite Cycle	
			Large Cycle (SYNn bit = 0)	Small Cycle (SYNn bit = 1)
4 bits	8 bits	$f_{\text{PWMc}}/16$	$f_{\text{PWMc}}/2^{12}$	$f_{\text{PWMc}}/2^4$
5 bits	8 bits	$f_{\text{PWMc}}/32$	$f_{\text{PWMc}}/2^{13}$	$f_{\text{PWMc}}/2^5$
6 bits	8 bits	$f_{\text{PWMc}}/64$	$f_{\text{PWMc}}/2^{14}$	$f_{\text{PWMc}}/2^6$
7 bits	8 bits	$f_{\text{PWMc}}/128$	$f_{\text{PWMc}}/2^{15}$	$f_{\text{PWMc}}/2^7$
8 bits	8 bits	$f_{\text{PWMc}}/256$	$f_{\text{PWMc}}/2^{16}$	$f_{\text{PWMc}}/2^8$

$f_{\text{PWMc}}$ : Select from  $\phi$ ,  $\phi/2$ ,  $\phi/4$ , and  $\phi/8$  by the PWPRn register.

[MEMO]

## CHAPTER 12 PORT FUNCTION

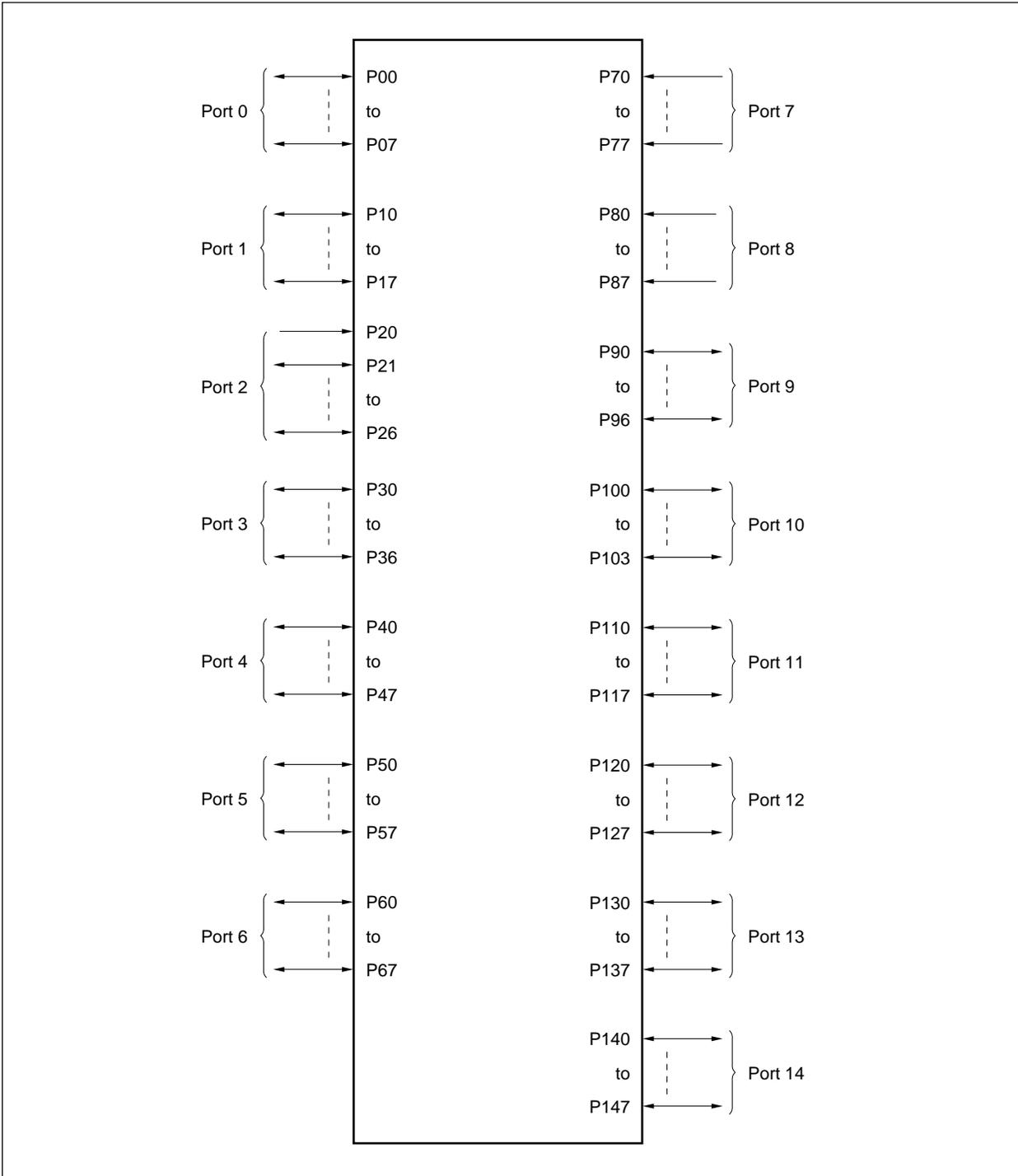
### 12.1 Features

The ports of the V854 have the following features:

- Number of pins: input: 16  
I/O: 96
- Also function as I/O pins of other peripheral functions
- Can be set in input/output mode in 1-bit units

## 12.2 Basic Configuration of Ports

The V854 is provided with a total of 112 input/output port pins (of which 16 are input-only port pins) that make up ports 0 to 14. The configuration of the V854's ports is shown below. (P20 cannot be used for a port function.)



**(1) Function of each port**

The ports of the V854 have the functions shown in the table below.

Each port can be manipulated in 8- or 1-bit units and perform various types of control operations. In addition to port functions, the ports also have functions as internal hardware input/output pins, when placed in the control mode.

For the details, refer to the description of each port. Figure 12-1 to 12-11 show the block diagram of the ports.

Port	Pin	Port Function	Function in Control Mode
Port 0	P00 to P07	8-bit I/O	Real time pulse unit (RPU) input/output External interrupt request input (capture trigger input)
Port 1	P10 to P17	8-bit I/O	RPU input/output External interrupt request input (capture trigger input) P17 is only for port.
Port 2	P20 to P26 <sup>Note</sup>	6-bit I/O	Non-maskable interrupt request input External interrupt request input (capture trigger input) Analog trigger input
Port 3	P30 to P36	7-bit I/O	Serial interface input/output (UART, CSI, I <sup>2</sup> C) P36 is only for port.
Port 4	P40 to P47	8-bit I/O	Address/data bus for external memory
Port 5	P50 to P57		
Port 6	P60 to P67	8-bit I/O	Address bus for external memory
Port 7	P70 to P77	8-bit input	Analog input to A/D converter (only for input in the port mode)
Port 8	P80 to P87		
Port 9	P90 to P96	7-bit I/O	Control signal input/output for external memory
Port 10	P100 to P103	4-bit I/O	PWM control signal output
Port 11	P110 to P117	8-bit I/O	RPU input/output External interrupt request input
Port 12	P120 to P127	8-bit I/O	Serial interface input/output Clock output P126 is only for port.
Port 13	P130 to P137	8-bit I/O	Real-time output port
Port 14	P140 to P147	8-bit I/O	Only for port

**Note** P20 cannot be used for a port function.

**Caution** when outputting in the control mode or switching a port that operates as an input/output pin to the control mode, take the following steps.

- (1) Set the corresponding bit of port n (Pn)(n = 0, 1, 3 to 6, 9 to 13) to the inactive level of the output signal in the control mode.
- (2) Switch to the control mode with a port n mode control register (PMCn).

If (1) above is omitted, the contents of port n (Pn) may momentarily be output when switching from the port mode to control mode.

(2) Register for setting function at reset and port/control mode of each port pin

(1/2)

Port Name	Pin Name	Function after Reset (Input/output is shown in parentheses)				Register for Setting Mode
		In Single-Chip Mode 1	In Single-Chip Mode 2	In ROM-less Mode 1	In ROM-less Mode 2	
Port 0	P00/TO00	P00 (input)				PMC0
	P01/TO01	P01 (input)				
	P02/INTP00	P02 (input)				
	P03/INTP01	P03 (input)				
	P04/INTP02	P04 (input)				
	P05/INTP03	P05 (input)				
	P06/INTP04/TCLR0	P06 (input)				
	P07/INTP05/TI0	P07 (input)				
Port 1	P10/INTP10	P10 (input)				PMC1
	P11/INTP11	P11 (input)				
	P12/INTP12	P12 (input)				
	P13/INTP13	P13 (input)				
	P14/INTP14/TI1	P14 (input)				
	P15/TO20	P15 (input)				
	P16/INTP20/TI20	P16 (input)				
	P17	P17 (input)				
Port 2	P20/NMI	NMI (input)				PMC2
	P21/INTP30	P21 (input)				
	P22/ADTRG	P22 (input)				
	P23/INTP50	P23 (input)				
	P24/INTP51	P24 (input)				
	P25/INTP52	P25 (input)				
	P26/INTP53	P26 (input)				
Port 3	P30/SO0/TDX	P30 (input)				PMC3
	P31/SI0/RXD	P31 (input)				
	P32/SCK0	P32 (input)				
	P33/SO1/SDA	P33 (input)				
	P34/SI1	P34 (input)				
	P35/SCK1/SCL	P35 (input)				
	P36	P36 (input)				
Port 4	P40/AD0 to P47/AD7	P40 to P47 (all input)		AD0 to AD7 (all input/output)		MM
Port 5	P50/AD8 to P57/AD15	P50 to P57 (all input)		AD8 to AD15 (all input/output)		MM
Port 6	P60/A16 to P67/A23	P60 to P67 (all input)		AD16 to A23 (all output)		MM
Port 7	P70/ANI0 to P77/ANI7	P70/ANI0 to P77/ANI7 (all input)				–
Port 8	P80/ANI8 to P87/ANI15	P80ANI8 to P87ANI15 (all input)				–

Port Name	Pin Name	Function after Reset (Input/output is shown in parentheses)				Register for Setting Mode
		In Single-Chip Mode 1	In Single-Chip Mode 2	In ROM-less Mode 1	In ROM-less Mode 2	
Port 9	P90/LBEN/WRL	P90 (input)		LBEN (output)	WRL (output)	MM
	P91/UBEN	P91 (input)		UBEN (output)		
	P92/R/W/WRH	P92 (input)		R/W (output)	WRH (output)	
	P93/DSTB/RD	P93 (input)		DSTB (output)	RD (output)	
	P94/ASTB	P94 (input)		ASTB (output)		
	P95/HLDAK	P95 (input)				
	P96/HLDRQ	P96 (input)				
Port 10	P100/PWM0	P100 (input)				PMC10
	P101/PWM1	P101 (input)				
	P102/PWM2	P102 (input)				
	P103/PWM3	P103 (input)				
Port 11	P110/TO21	P110 (input)				PMC11
	P111/INTP21/TI21	P111 (input)				
	P112/TO22	P112 (input)				
	P113/INTP22/TI22	P113 (input)				
	P114/TO23	P114 (input)				
	P115/INTP23/TI23	P115 (input)				
	P116/TO24	P116 (input)				
	P117/INTP24/TI24	P117 (input)				
Port 12	P120/SO2	P120 (input)				PMC12
	P121/SI2	P121 (input)				
	P122/SCK2	P122 (input)				
	P123/SO3	P123 (input)				
	P124/SI3	P124 (input)				
	P125/SCK3	P125 (input)				
	P126	P126 (input)				
	P127/CLO	P127 (input)				
Port 13	P130/RTP0 to P137/RTP7	P130 to P137 (all input)				PMC13
Port 14	P140 to P147	P140 to P147 (all input)				–

Figure 12-1. Block Diagram of Type A

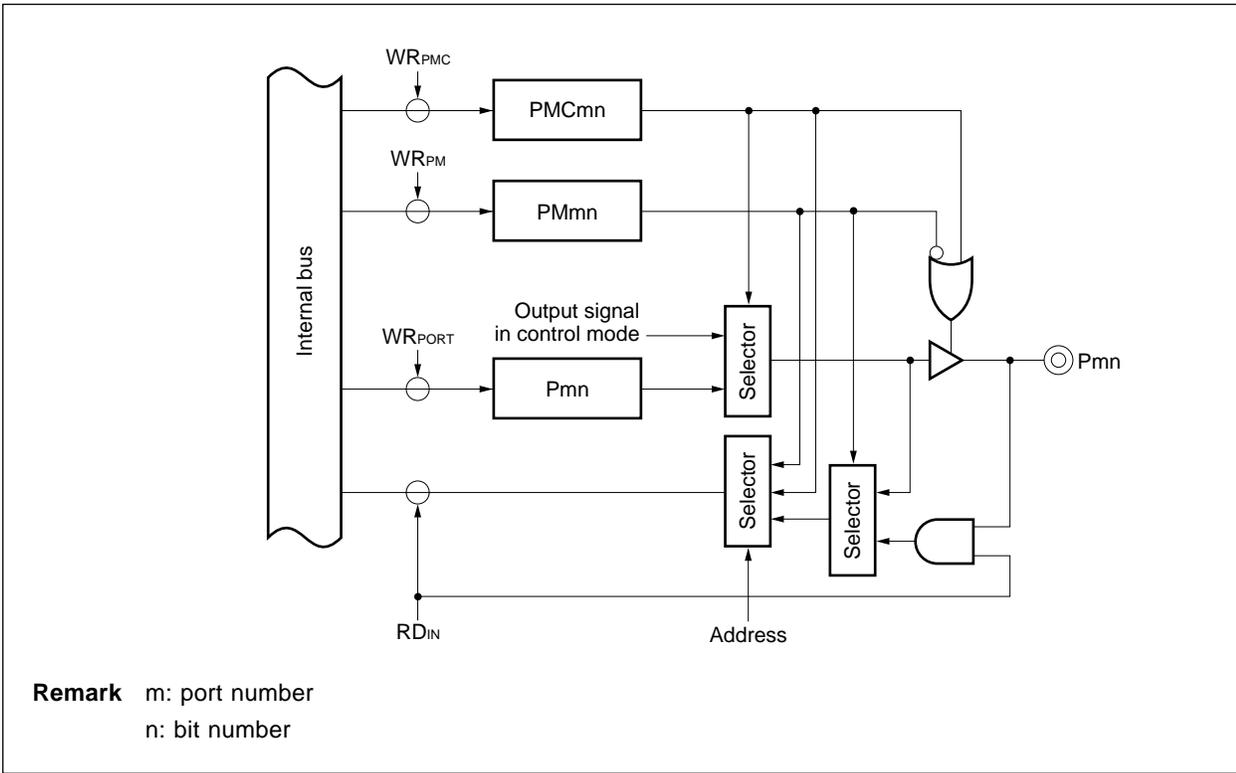


Figure 12-2. Block Diagram of Type B

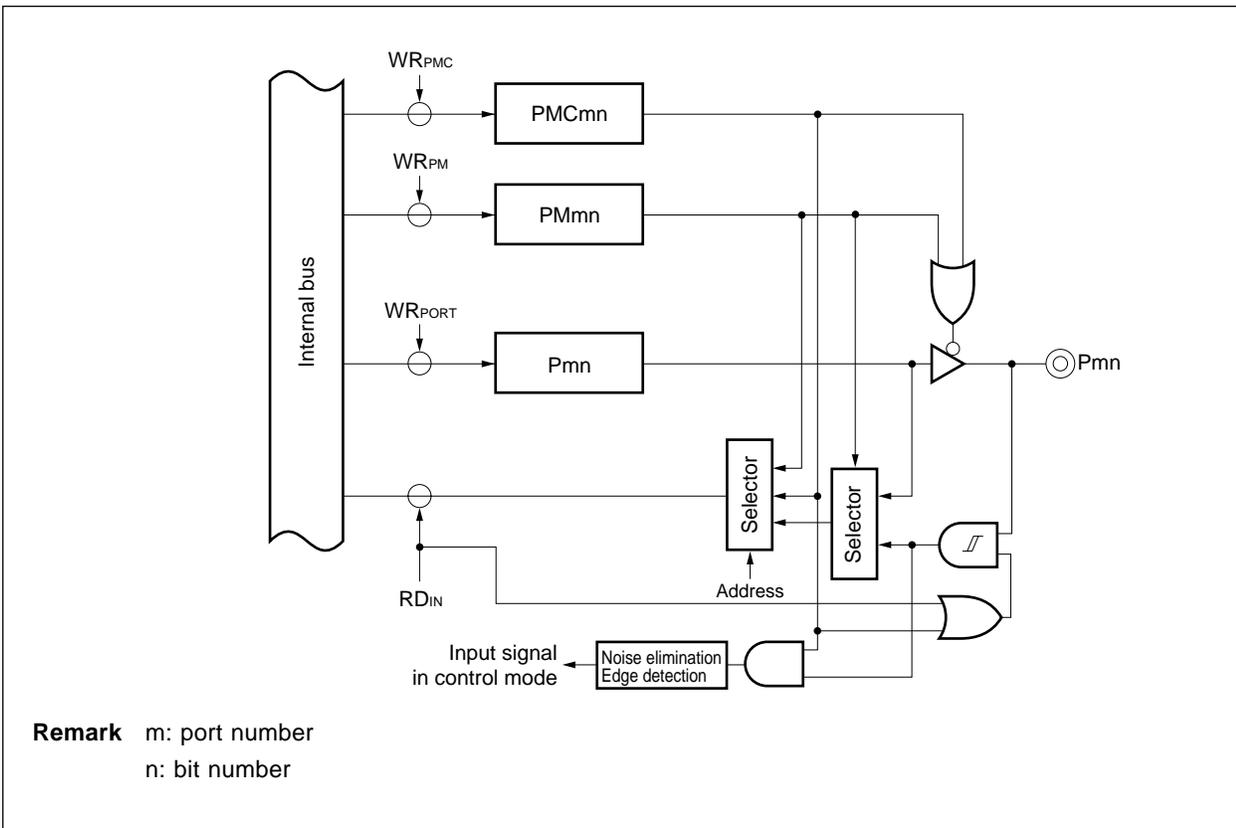




Figure 12-5. Block Diagram of Type E

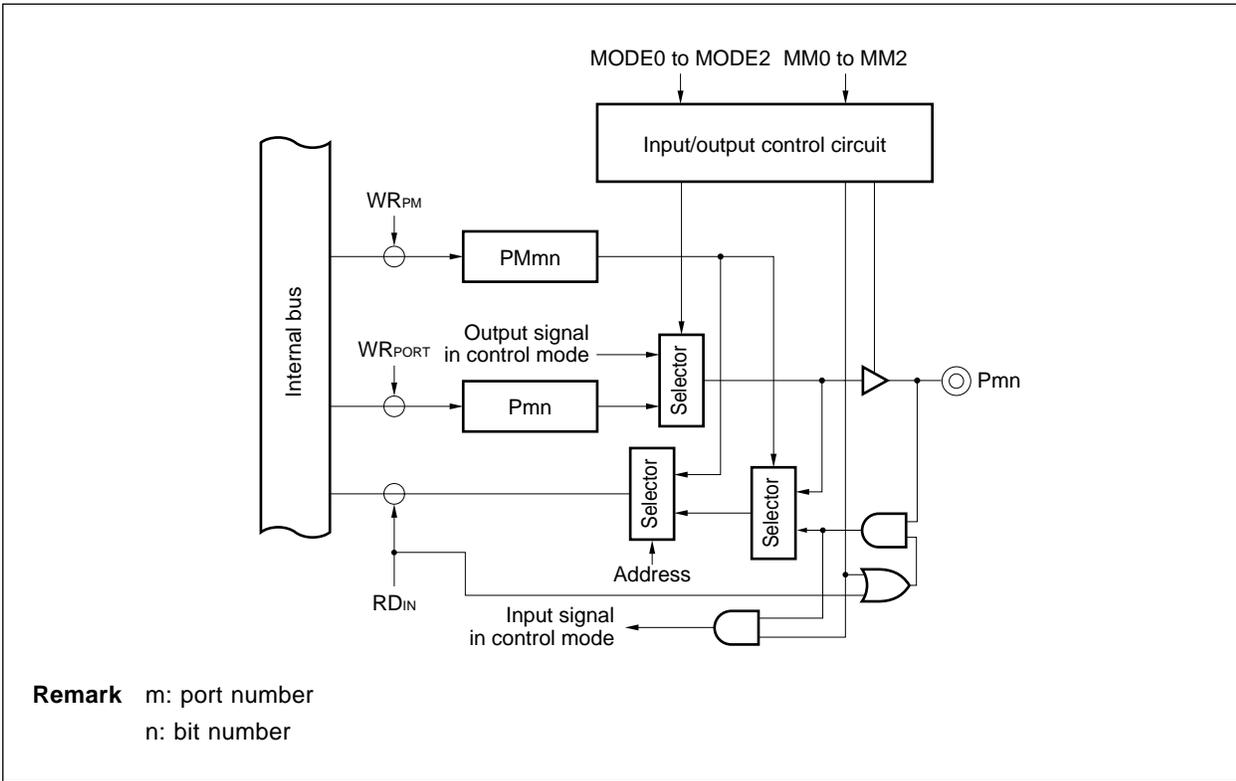


Figure 12-6. Block Diagram of Type F

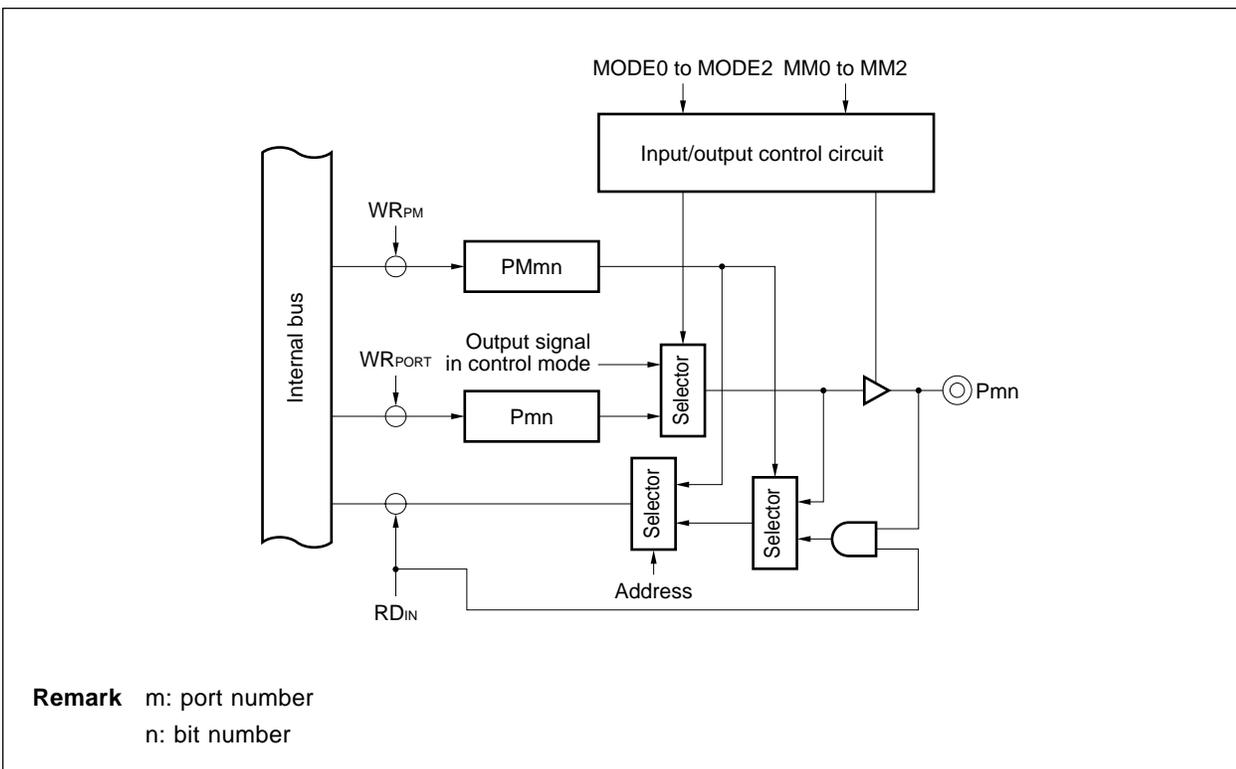


Figure 12-7. Block Diagram of Type G

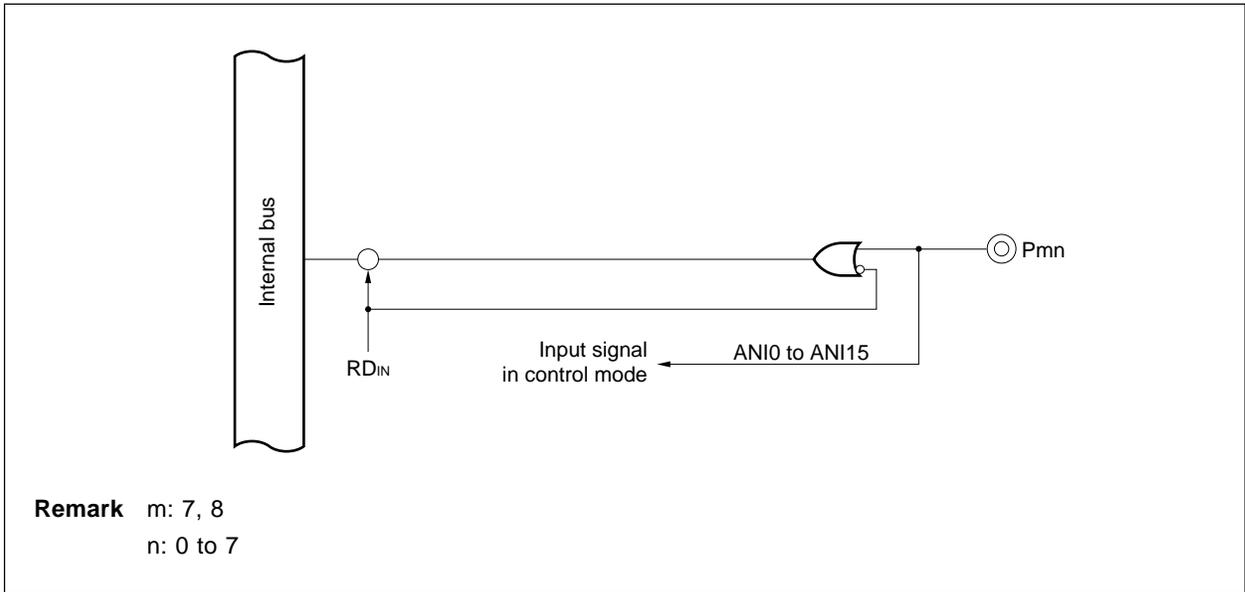


Figure 12-8. Block Diagram of Type H

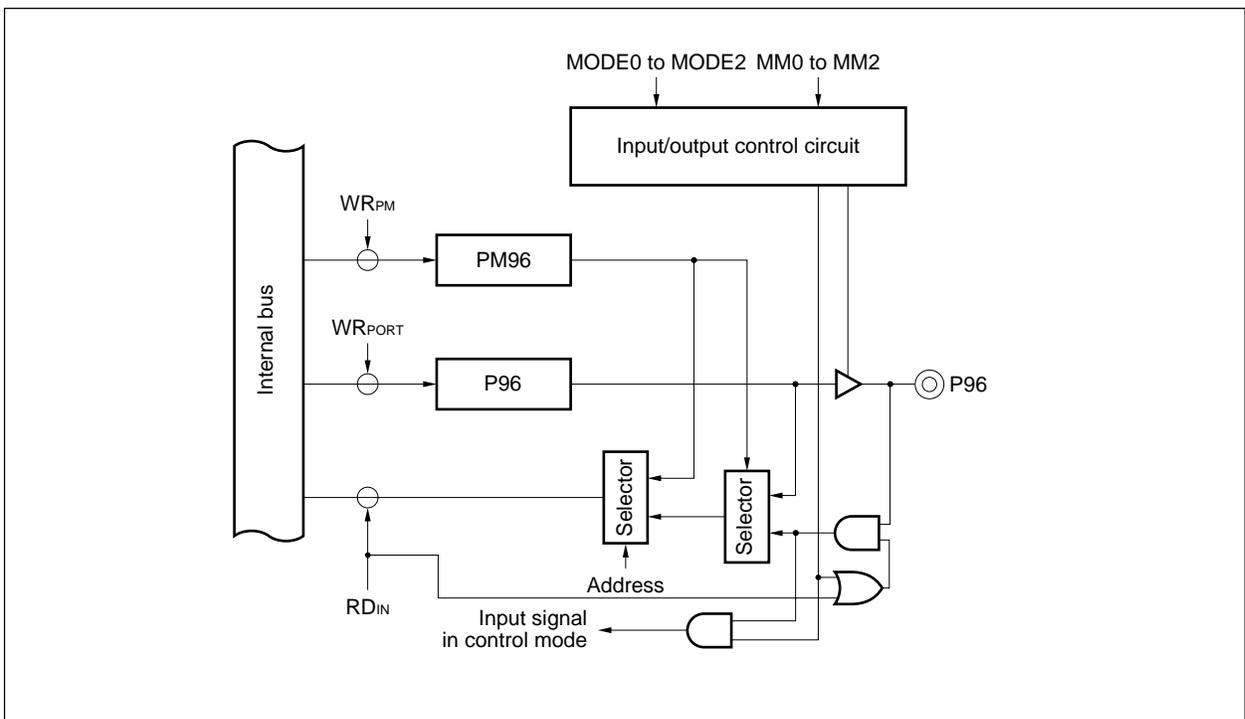


Figure 12-9. Block Diagram of Type I

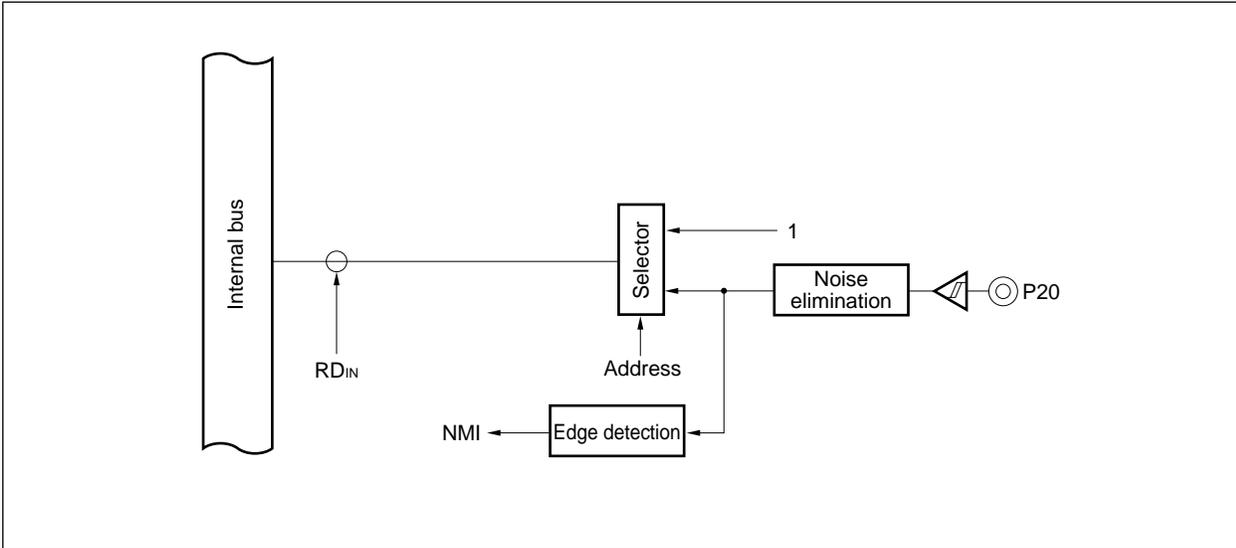
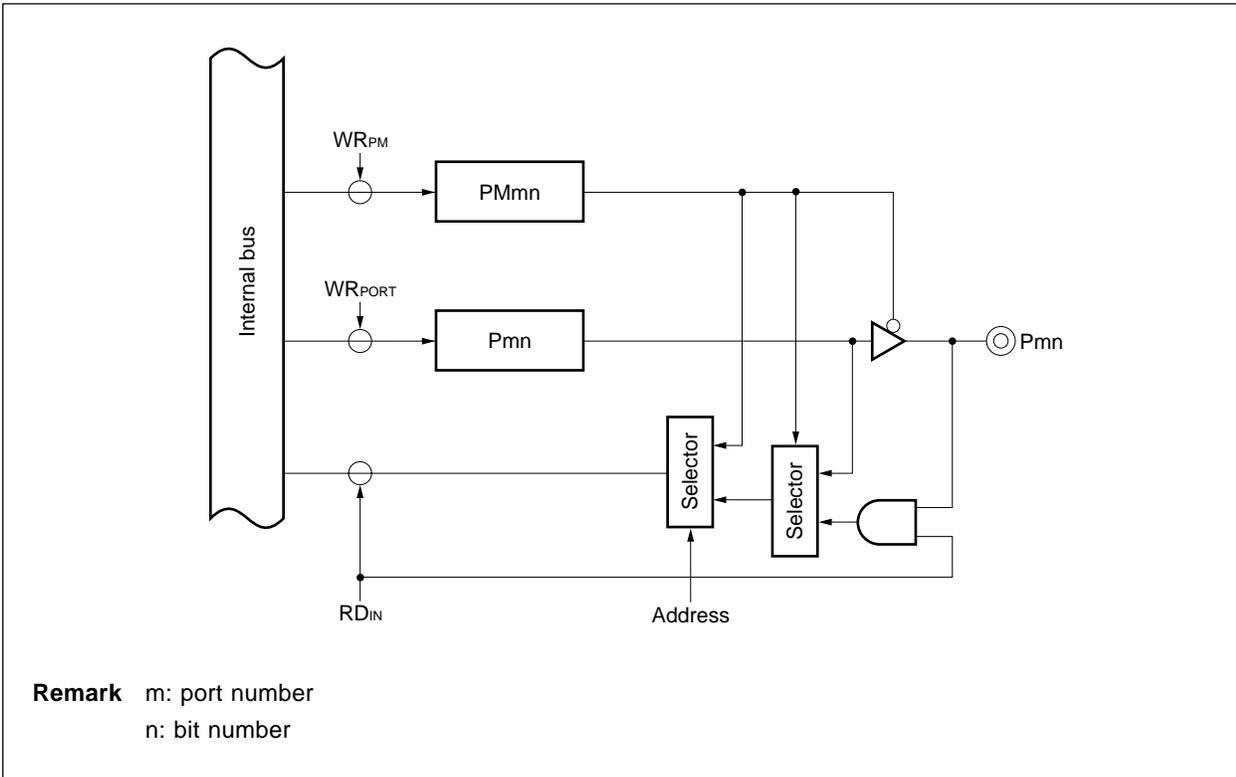
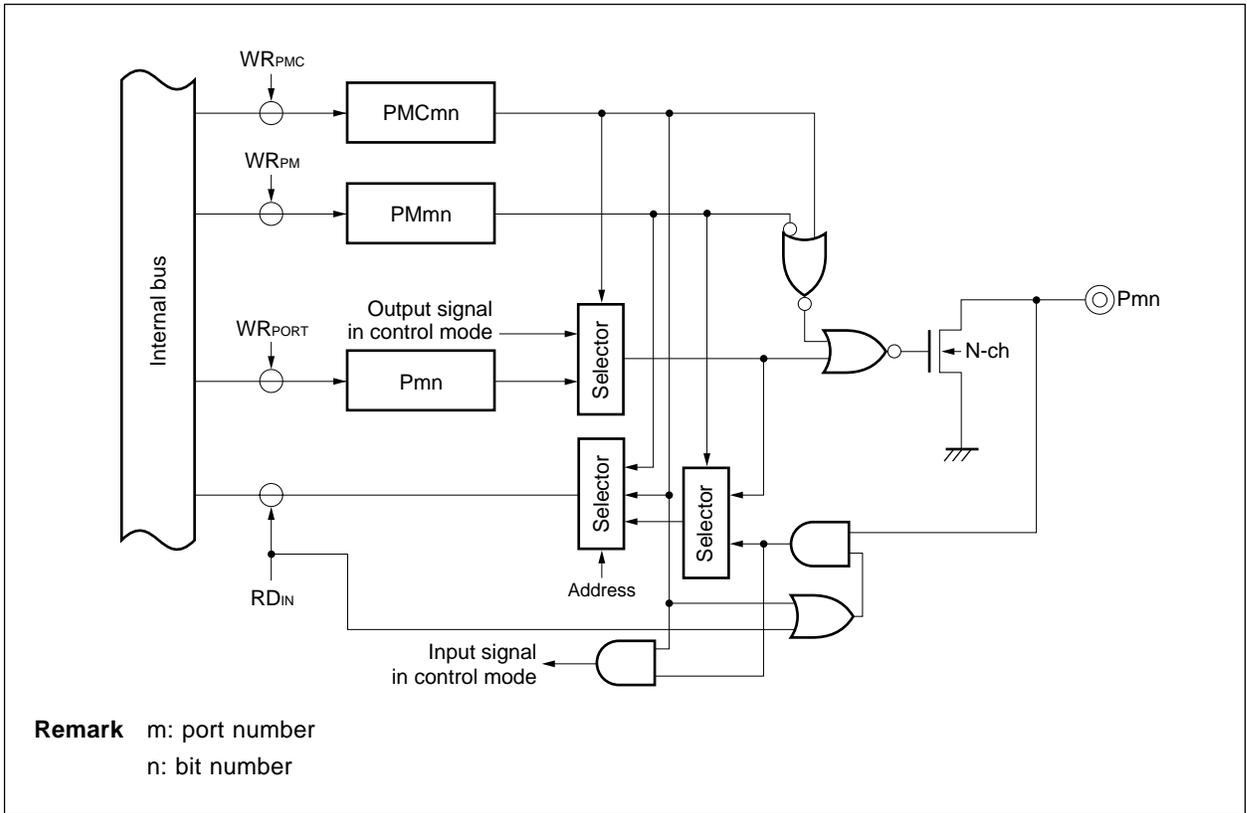


Figure 12-10. Block Diagram of Type J



**Remark** m: port number  
n: bit number

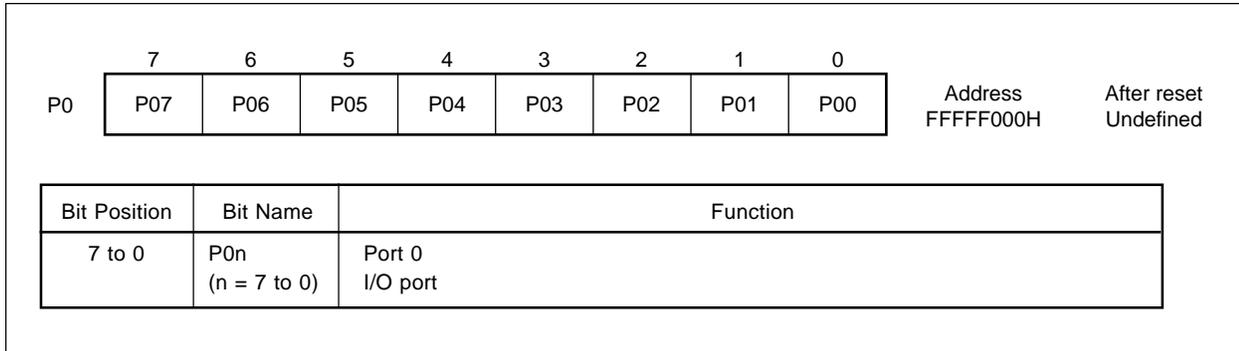
Figure 12-11. Block Diagram of Type K



## 12.3 Port Pin Function

### 12.3.1 Port 0

Port 0 is an 8-bit input/output port that can be set to the input or output mode in 1-bit units.



In addition to the function as a general I/O port, this port can also be used to input/output signals of the real-time pulse unit (RPU) and input external interrupt requests, when placed in the control mode.

#### (1) Operations in control mode

Port	Control Mode	Function in Control Mode	Block Type
Port 0	P00	TO00	RPU output
	P01	TO01	
Port 0	P02	INTP00	External interrupt request input/RPU capture trigger input
	P03	INTP01	
	P04	INTP02	
	P05	INTP03	
Port 0	P06	TCLR0/INTP04	External interrupt request input/RPU input
	P07	TI0/INTP05	

**(2) Setting input/output mode and control mode**

The input/output mode of port 0 is set by port mode register 0 (PM0). The control mode is set by port mode control register 0 (PMC0).

**Port 0 mode register (PM0)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	Address FFFFFF020H	After reset FFH

Bit Position	Bit Name	Function
7 to 0	PM00 to PM07	Port Mode Sets P00 to P07 pins in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Port 0 mode control register (PMC0)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PMC0	PMC07	PMC06	PMC05	PMC04	PMC03	PMC02	PMC01	PMC00	Address FFFFFF040H	After reset 00H

Bit Position	Bit Name	Function
7	PMC07	Port mode Control Specifies operation mode of P07 pin. 0: I/O port mode 1: External interrupt request input (INTP05)/TI0 input mode
6	PMC06	Port Mode Control Specifies operation mode of P06 pin. 0: I/O port mode 1: External interrupt request input (INTP04)/TCLR0 input mode
5 to 2	PMC05 to PMC02	Port Mode Control Specifies operation mode of P05 to P02 pins. 0: I/O port mode 1: External interrupt request input (INTP03 to INTP00)
1	PMC01	Port Mode Control Specifies operation mode of P01 pin. 0: I/O port mode 1: TO01 output mode
0	PMC00	Port Mode Control Specifies operation mode of P00 pin. 0: I/O port mode 1: TO00 output mode

**12.3.2 Port 1**

Port 1 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

P1	7	6	5	4	3	2	1	0	Address FFFFF002H	After reset Undefined
	P17	P16	P15	P14	P13	P12	P11	P10		
Bit Position		Bit Name		Function						
7 to 0		P1n (n = 7 to 0)		Port 1 I/O port						

In addition to the function as a general I/O port, this port can also be used to input/output signals of the real-time pulse unit (RPU), and to input external interrupts when placed in the control mode.

**(1) Operations in control mode**

Port		Control Mode	Function in Control Mode	Block Type
Port 1	P10	INTP10	External interrupt request input/RPU capture trigger input	B
	P11	INTP11		
	P12	INTP12		
	P13	INTP13		
	P14	TI1/INTP14	External interrupt request input	
	P15	TO20	RPU output	A
	P16	TI20/INTP20	External interrupt request input/RPU input	B
	P17	–	Port only	J

**(2) Setting input/output mode**

The input/output mode of port 1 is set by port mode register 1 (PM1). The control mode is set by port mode control register 1 (PMC1).

**Port 1 mode register (PM1)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	Address FFFFF022H	After reset FFH

Bit Position	Bit Name	Function
7 to 0	PM1n (n = 7 to 0)	Port Mode Sets P1n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Port 1 mode control register (PMC1)**

This register can be read/written in 8- or 1-bit units. However, bit 7 is fixed to "0" and ignored when "1" is written.

	7	6	5	4	3	2	1	0		
PMC1	0	PMC16	PMC15	PMC14	PMC13	PMC12	PMC11	PMC10	Address FFFFF042H	After reset 00H

Bit Position	Bit Name	Function
6	PMC16	Port Mode Control Specifies operation mode of P16 pin. 0: I/O port mode 1: External interrupt request input (INTP20)/TI20 input mode
5	PMC15	Port Mode Control Specifies operation mode of P15 pin. 0: I/O port mode 1: TO20 output mode
4	PMC14	Port Mode Control Specifies operation mode of P14 pin. 0: I/O port mode 1: External interrupt request input (INTP14)/TI1 input mode
3 to 0	PMC13 to PMC10	Port Mode Control Specifies operation mode of P13 to P10 pins. 0: I/O port mode 1: External interrupt request input (INTP13 to INTP10)

**12.3.3 Port 2**

Port 2 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units. P20, however, always operates as the NMI input when an edge is input.

P2	7	6	5	4	3	2	1	0		
	-	P26	P25	P24	P23	P22	P21	P20	Address FFFFF004H	After reset Undefined

Bit Position	Bit Name	Function
6 to 1	P2n (n = 6 to 1)	Port 2 I/O port
0	P20	NMI input level

In addition to the function as a port, this port can also be used as the external interrupt request input, when placed in the control mode.

When port 2 is accessed in 8-bit units for write, the data in the higher 1 bit is ignored. When it is accessed in 8-bit units for read, undefined data is read.

**(1) Operations in control mode**

Port	Control Mode	Function in Control Mode	Block Type
Port 2	P20	NMI	Non-maskable interrupt request input
	P21	INTP30	External interrupt request input (capture trigger input)
	P22	ADTRG	
	P23	INTP50	External interrupt request input
	P24	INTP51	
	P25	INTP52	
	P26	INTP53	
			B

**(2) Setting input/output mode and control mode**

The input/output mode of port 2 is set by port mode register 2 (PM2). The control mode is set by port mode control register 2 (PMC2). P20 is fixed to NMI input.

**Port 2 mode register (PM2)**

This register can be read/written in 8- or 1-bit units. However, bit 0 and bit 7 are fixed to “1” by hardware and ignored when 0 is written in.

	7	6	5	4	3	2	1	0		
PM2	1	PM26	PM25	PM24	PM23	PM22	PM21	1	Address FFFFF024H	After reset FFH

Bit Position	Bit Name	Function
6 to 1	PM2n (n = 6 to 1)	Port Mode Sets P2n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Port 2 mode control register (PMC2)**

This register can be read/written in 8- or 1-bit units. However, bit 0 is fixed to “1” by hardware, and ignored when “0” is written. Bit 7 is fixed to “0” by hardware, and ignored when “1” is written.

	7	6	5	4	3	2	1	0		
PMC2	0	PMC26	PMC25	PMC24	PMC23	PMC22	PMC21	1	Address FFFFF044H	After reset 01H

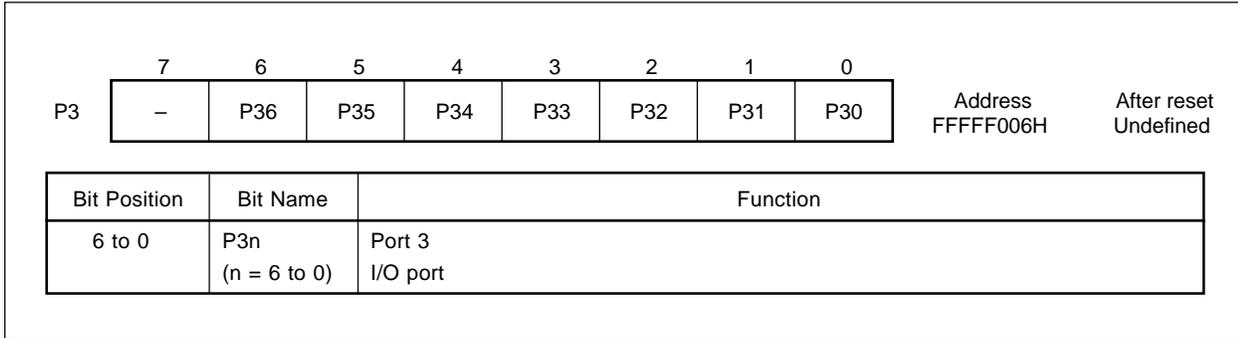
Bit Position	Bit Name	Function
6 to 3	PMC26 to PMC23	Port Mode Control Sets operation mode of P26 to P23 pins. 0: I/O port mode 1: External interrupt request input (INTP53 to INTP50)
2	PMC22	Port Mode Control Sets operation mode of P22 pin. 0: I/O port mode 1: AD converter trigger input (ADTRG)
1	PMC21	Port Mode Control Sets operation mode of P21 pin. 0: I/O port mode 1: External interrupt request input (INTP30)

**Remark** Bit 0 is fixed to NMI input mode.

**12.3.4 Port 3**

Port 3 is a 7-bit input/output port that can be set in the input or output mode in 1-bit units.



In addition to the function as a port, this port can also be used as the input/output lines of the serial interface (UART, CSI, I<sup>2</sup>C), when placed in the control mode. P33 and P35 are multiplexed with SDA and SCL pin of I<sup>2</sup>C bus, respectively, and output is N-ch open drain.

When port 3 is accessed in 8-bit units for write, the data in the higher 1 bit is ignored. When it is accessed in 8-bit units for read, undefined data is read.

**(1) Operations in control mode**

Port	Control Mode	Function in Control Mode	Block Type	
Port 3	P30	SO0/TXD	Serial interface I/O (UART, CSI, I <sup>2</sup> C)	A
	P31	SI0/RXD		D
	P32	$\overline{\text{SCK0}}$		C
	P33	SO1/SDA		K
	P34	SI1		D
	P35	$\overline{\text{SCK1/SCL}}$		K
	P36	–	Port only	J

**(2) Setting input/output mode and control mode**

The input/output mode of port 3 is set by port mode register 3 (PM3). The control mode is set by port mode control register 3 (PMC3).

**Port 3 mode register (PM3)**

This register can be read/written in 8- or 1-bit units.

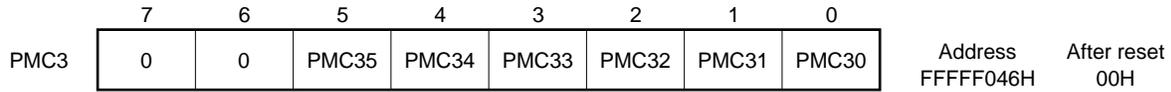
	7	6	5	4	3	2	1	0		
PM3	1	PM36	PM35	PM34	PM33	PM32	PM31	PM30	Address FFFFFF026H	After reset FFH

Bit Position	Bit Name	Function
6 to 0	PM3n (n = 6 to 1)	Port Mode Sets P3n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Port 3 mode control register (PMC3)**

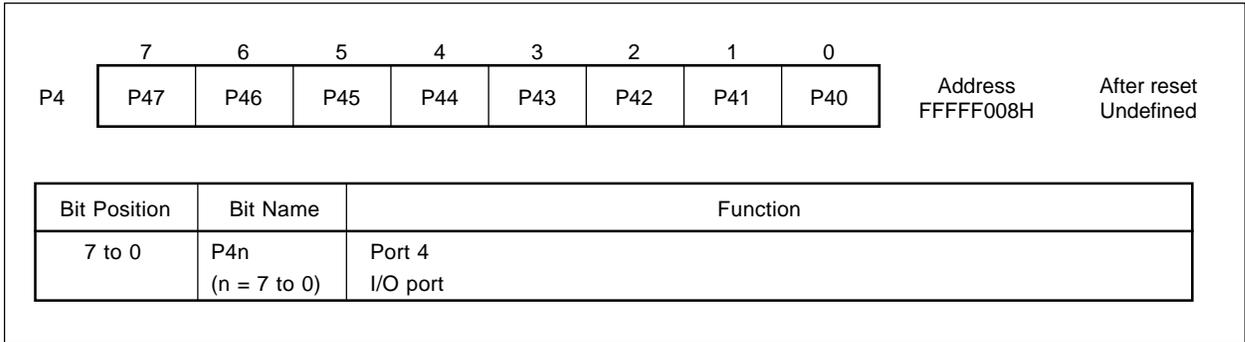
This register can be read/written in 8- or 1-bit units. However, the higher 2 bits are fixed to “0” by hardware, and ignored when “1 “ is written in.



Bit Position	Bit Name	Function
5	PMC35	Port Mode Control Sets operation mode of P35 pin. 0: I/O port mode 1: SCK1/SCL output mode
4	PMC34	Port Mode Control Sets operation mode of P34 pin. 0: I/O port mode 1: SI1 input mode
3	PMC33	Port Mode Control Sets operation mode of P33 pin. 0: I/O port mode 1: SO1/SDA output mode
2	PMC32	Port Mode Control Sets operation mode of P32 pin. 0: I/O port mode 1: SCK0 I/O mode
1	PMC31	Port Mode Control Sets operation mode of P31 pin. 0: I/O port mode 1: SI0/RXD input mode
0	PMC30	Port Mode Control Sets operation mode of P30 pin. 0: I/O port mode 1: SO0/TXD output mode

**12.3.5 Port 4**

Port 4 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.



In addition to the function as a general I/O port, this port also serves as an external address/data bus for external memory expansion, when placed in the control mode (external expansion mode).

**(1) Operation in control mode**

Port		Control Mode	Function in Control Mode	Block Type
Port 4	P40 to P47	AD0 to AD7	Address/data bus for external memory	E

**(2) Setting input/output mode and control mode**

The input/output mode of port 4 is set by port mode register 4 (PM4). The control mode (external expansion mode) is set by mode specification pins MODE and memory expansion mode register (MM: refer to **3.4.6 (1)**) (n = 0 to 2).

**Port 4 mode register (PM4)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	Address	After reset
									FFFFFF028H	FFH

Bit Position	Bit Name	Function
7 to 0	PM4n (n = 7 to 0)	Port Mode Sets P4n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Operation mode of port 4**

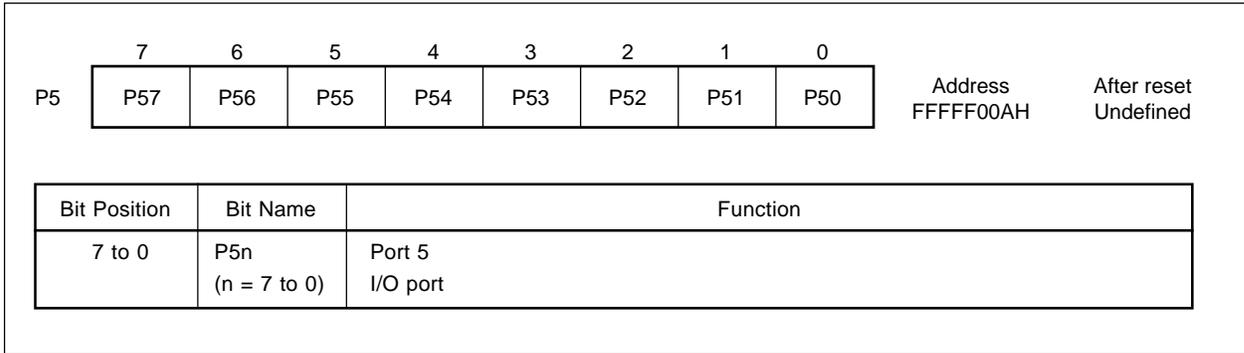
Bit of MM Register			Operation Mode								
MM2	MM1	MM0	P40	P41	P42	P43	P44	P45	P46	P47	
0	0	0	Port								
0	1	1	Address/data bus (AD0 to AD7)								
1	0	0									
1	0	1									
1	1	1	RFU (reserved)								
Others			RFU (reserved)								

For the details of mode selection by the MODE pins, refer to **3.3.2 Specifying operation mode**.

In the ROM-less mode, MM0 to MM2 bits are initialized to 111 at system reset, enabling the external expansion mode. External expansion can be disabled by programming the MM0 to MM2 bits and setting the port mode. If MM0 to MM2 are cleared to 000, the subsequent external instruction cannot be fetched.

**12.3.6 Port 5**

Port 5 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.



In addition to the function as a general I/O port, this port also serves as an external address/data bus for external memory expansion, when placed in the control mode (external expansion mode).

**(1) Operation in control mode**

	Port	Control Mode	Function in Control Mode	Block Type	
	Port 5	P50 to P57	AD8 to AD15	Address/data bus for external memory	E

**(2) Setting input/output mode and control mode**

The input/output mode of port 5 is set by port mode register 5 (PM5). The control mode (external expansion mode) is set by mode specification pins MODEn and memory expansion mode register (MM: refer to 3.4.6 (1)) (n = 0 to 2).

**Port 5 mode register (PM5)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	Address FFFFFF02AH	After reset FFH

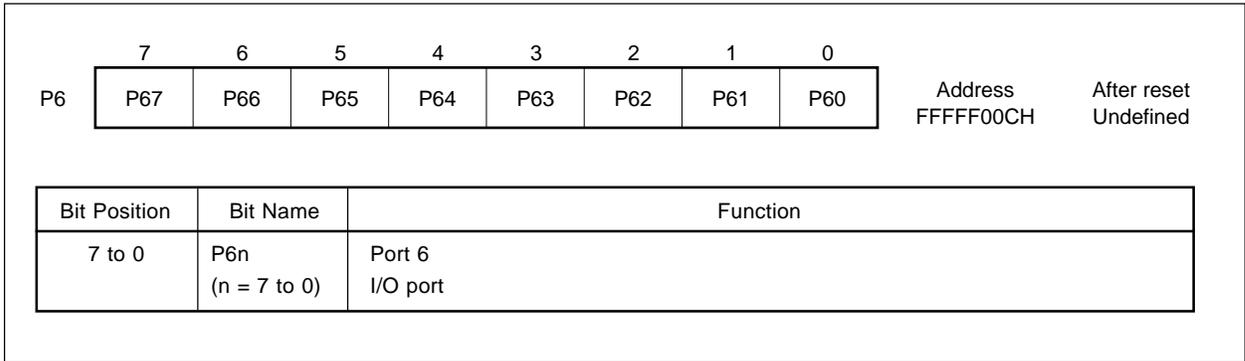
Bit Position	Bit Name	Function
7 to 0	PM5n (n = 7 to 0)	Port Mode Sets P5n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Operation mode of port 5**

Bit of MM Register			Operation Mode							
MM2	MM1	MM0	P50	P51	P52	P53	P54	P55	P56	P57
0	0	0	Port							
0	1	1	Address/data bus (AD8 to AD15)							
1	0	0								
1	0	1								
Others			RFU (reserved)							

**12.3.7 Port 6**

Port 6 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.



In addition to the function as a general I/O port, this port also serves as an external address bus for external memory expansion, when placed in the control mode (external expansion mode).

**(1) Operation in control mode**

	Port	Control Mode	Function in Control Mode	Block Type
Port 6	P60 to P67	A16 to A23	Address bus for external memory expansion	F

**(2) Setting input/output mode and control mode**

The input/output mode of port 6 is set by port mode register 6 (PM6). The control mode (external expansion mode) is set by mode specification pins MODE<sub>n</sub> and memory expansion mode register (MM: refer to 3.4.6 (1)) (n = 0 to 2).

**Port 6 mode register (PM6)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PM6	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60	Address FFFFFF02CH	After reset FFH

Bit Position	Bit Name	Function
7 to 0	PM6 <sub>n</sub> (n = 7 to 0)	Port Mode Sets P6 <sub>n</sub> pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Operation mode of port 6**

Bit of MM Register			Operation Mode							
MM2	MM1	MM0	P60	P61	P62	P63	P64	P65	P66	P67
0	0	0	Port							
0	1	1								
1	0	0	A16	A17						
1	0	1	A18	A19						
1	1	0			A20	A21				
1	1	1			A22	A23				
Others			RFU (reserved)							

**12.3.8 Port 7, port 8**

Port 7 and port 8 are 8-bit input only ports and all pins of port 7 and port 8 are fixed in the input mode.

	7	6	5	4	3	2	1	0		
P7	P77	P76	P75	P74	P73	P72	P71	P70	Address FFFFFF0EH	After reset Undefined
	7	6	5	4	3	2	1	0		
P8	P87	P86	P85	P84	P83	P82	P81	P80	Address FFFFFF010H	After reset Undefined

In addition to the function as input ports, these ports can also always be used as analog input for the A/D converter.

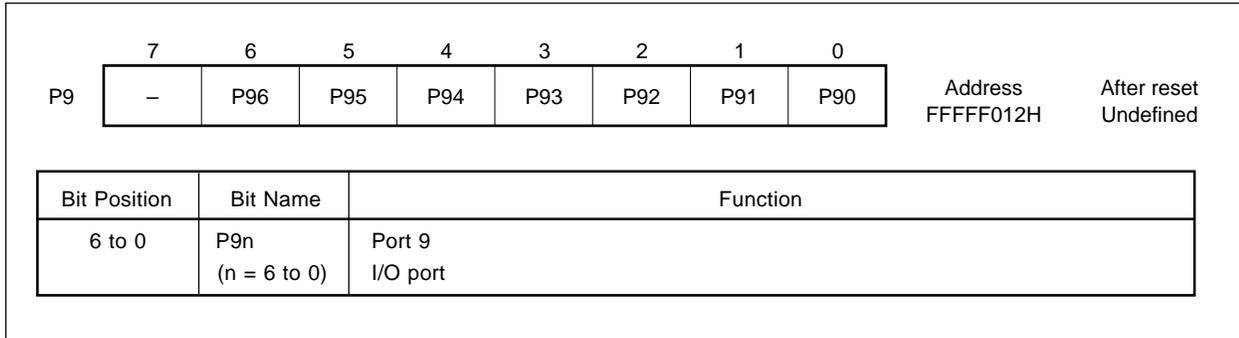
These ports are used also as the analog input pins (ANI0 to ANI7 and ANI8 to ANI15), but the input port and analog input pin cannot be switched. The status of each pin can be read by reading out ports.

**(1) Normal operation**

Port		Control Mode	Function in Control Mode	Block Type
Port 7	P70 to P77	ANI0 to ANI7	Analog input to A/D converter (only for input in port mode)	G
Port 8	P80 to P87	ANI8 to ANI15		

**12.3.9 Port 9**

Port 9 is a 7-bit input/output port that can be set in the input or output mode in 1-bit units.



In addition to the function as a general I/O port, this port can also be used to output external bus control signals and output bus hold control signals, when placed in the control mode (external expansion mode). When port 9 is accessed in 8-bit units for write, the higher 1 bit is ignored. When it is accessed in 8-bit units for read, undefined data is read.

**(1) Operations in control mode**

Port	Control Mode	Function in Control Mode	Block Type
Port 9	P90	$\overline{\text{LBEN}}/\overline{\text{WRL}}$	Control signal output for external memory
	P91	$\overline{\text{UBEN}}$	
	P92	$\overline{\text{R}}/\overline{\text{W}}/\overline{\text{WRH}}$	
	P93	$\overline{\text{DSTB}}/\overline{\text{RD}}$	
	P94	ASTB	
	P95	$\overline{\text{HLDAK}}$	Bus hold acknowledge signal output
P96	$\overline{\text{HLDRQ}}$	Bus hold request signal input	

**(2) Setting input/output mode and control mode**

The input/output mode of port 9 is set by port mode register 9 (PM9). The control mode (external expansion mode) is set by the mode specification pin MODEn and the memory expansion mode register (MM: refer to 3.4.6 (1)) (n = 0 to 2).

**Port 9 mode register (PM9)**

This register can be read/written in 8- or 1-bit units. However, bit 7 is ignored during write access, and undefined during read access.

	7	6	5	4	3	2	1	0		
PM9	-	PM96	PM95	PM94	PM93	PM92	PM91	PM90	Address FFFFF032H	After reset FFH

Bit Position	Bit Name	Function
6 to 0	PM9n (n = 6 to 0)	Port Mode Sets P9n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Operation mode of port 9**

P90 to P94

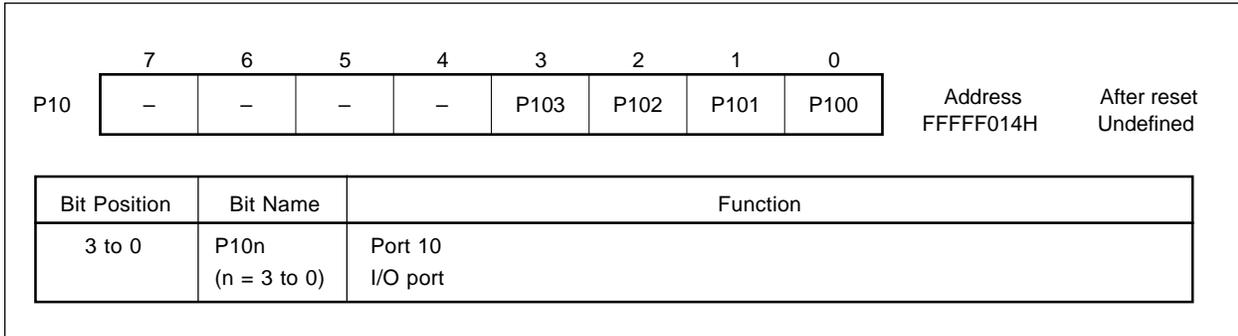
Bit of MM Register			Operation Mode				
MM2	MM1	MM0	P90	P91	P92	P93	P94
0	0	0	Port				
0	1	1	$\overline{\text{LBEN}}$ , $\overline{\text{WRL}}$	$\overline{\text{UBEN}}$	$\text{R}/\overline{\text{W}}$ , $\overline{\text{WRH}}$	$\overline{\text{DSTB}}$ , $\overline{\text{RD}}$	$\text{ASTB}$
1	0	0					
1	0	1					
1	1	1					
Others			RFU (reserved)				

P95, P96

MM3	Operation Mode	P95	P96
0	Port mode	Port	
1	External expansion mode	$\overline{\text{HLDAK}}$	$\overline{\text{HLDRQ}}$

**12.3.10 Port 10**

Port 10 is a 4-bit input/output port that can be set in the input or output mode in 1-bit units.



When port 10 is accessed in 8-bit units for write, the data in the higher 4 bits is ignored. When it is accessed in 8-bit units for read, undefined data is read.

In addition to the function as a general I/O port, this port can also be used to output the PWMn.

**(1) Operations in control mode**

Port	Control Mode	Function in Control Mode	Block Type
Port 10	P100	PWM0	A
	P101	PWM1	
	P102	PWM2	
	P103	PWM3	

**(2) Setting input/output mode and control mode**

The input/output mode of port 10 is set by port mode register 10 (PM10). The control mode is set by port mode control register 10 (PMC10).

**Port 10 mode register (PM10)**

This register can be read/written in 8- or 1-bit units. However, the higher 4 bits are fixed to “0” by hardware and ignored when 0 is written in.

	7	6	5	4	3	2	1	0		
PM10	1	1	1	1	PM103	PM102	PM101	PM100	Address FFFFFF034H	After reset FFH

Bit Position	Bit Name	Function
3 to 0	PM10n (n = 3 to 0)	Port Mode Sets P10n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Port 10 mode control register (PMC10)**

This port can be read/written in 8- or 1-bit units. However, the higher 4 bits are fixed to “0” and ignored if “1” is written in.

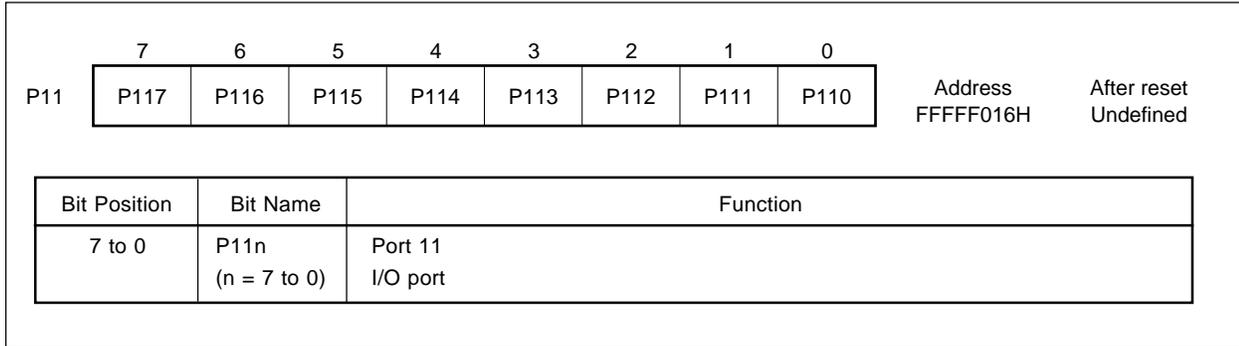
	7	6	5	4	3	2	1	0		
PMC10	0	0	0	0	PMC103	PMC102	PMC101	PMC100	Address FFFFFF054H	After reset 00H

Bit Position	Bit Name	Function
3 to 0	PMC10n (n = 3 to 0)	Port Mode Control Sets operation mode of P10n pin. 0: I/O port mode 1: PWMn output mode

**12.3.11 Port 11**

Port 11 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.



In addition to the function as a port, this port can also be used to input/output signals of the real-time pulse unit (RPU) and to input external interrupt requests, when placed in the control mode.

**(1) Operations in control mode**

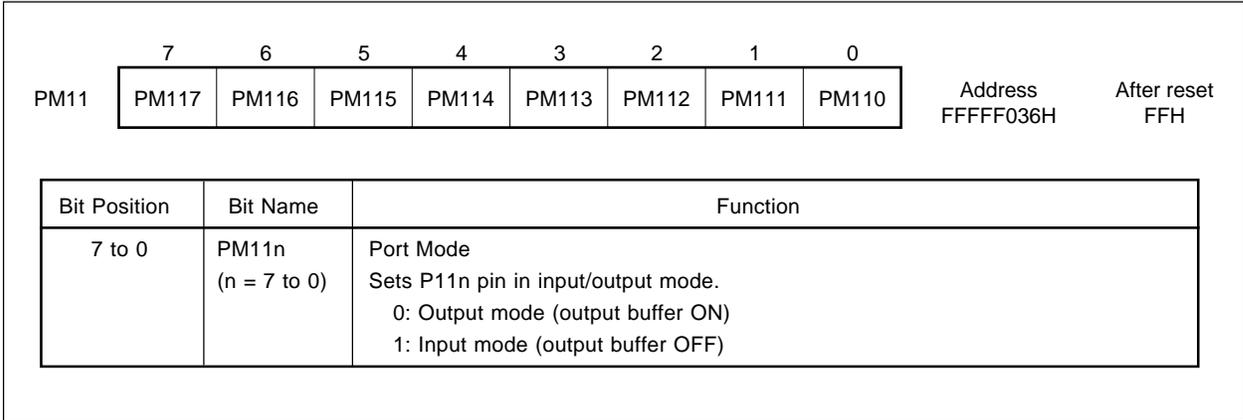
Port	Alternate Function	Function in Control Mode	Block Type	
Port 11	P110	TO21	RPU output	A
	P111	TI21/INTP21	RPU input/external interrupt request input	B
	P112	TO22	RPU output	A
	P113	TI22/INTP22	RPU input/external interrupt request input	B
	P114	TO23	RPU output	A
	P115	TI23/INTP23	RPU input/external interrupt request input	B
	P116	TO24	RPU output	A
	P117	TI24/INTP24	RPU input/external interrupt request input	B

**(2) Setting input/output mode and control mode**

The input/output mode of port 11 is set by port mode register 11 (PM11). The control mode is set by port mode control register 11 (PMC11).

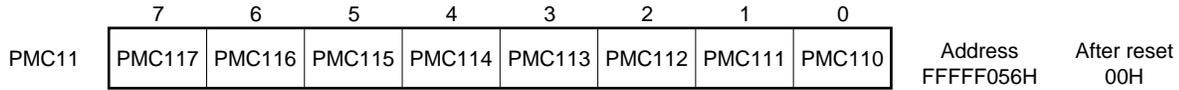
**Port 11 mode register (PM11)**

This register can be read/written in 8- or 1-bit units.



**Port 11 mode control register (PMC11)**

This register can be read/written in 8- or 1-bit units.



Bit Position	Bit Name	Function
7	PMC117	Port Mode Control Sets P117 pin in input/output mode. 0: I/O port mode 1: External interrupt request input (INTP24)/TI24 input mode
6	PMC116	Port Mode Control Sets P116 pin in input/output mode. 0: I/O port mode 1: TO24 output mode
5	PMC115	Port Mode Control Sets P115 pin in input/output mode. 0: I/O port mode 1: External interrupt request input (INTP23)/TI23 input mode
4	PMC114	Port Mode Control Sets P114 pin in input/output mode. 0: I/O port mode 1: TO23 output mode
3	PMC113	Port Mode Control Sets P113 pin in input/output mode. 0: I/O port mode 1: External interrupt request input (INTP22)/TI22 input mode
2	PMC112	Port Mode Control Sets P112 pin in input/output mode. 0: I/O port mode 1: TO22 output mode
1	PMC111	Port Mode Control Sets P111 pin in input/output mode. 0: I/O port mode 1: External interrupt request input (INTP21)/TI21 input mode
0	PMC110	Port Mode Control Sets P110 pin in input/output mode. 0: I/O port mode 1: TO21 output mode

**12.3.12 Port 12**

Port 12 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.

	7	6	5	4	3	2	1	0		
P12	P127	P126	P125	P124	P123	P122	P121	P120	Address FFFFF018H	After reset Undefined

Bit Position	Bit Name	Function
7 to 0	P12n (n = 7 to 0)	Port 12 I/O port

In addition to the function as a port, this port can also be used as the input/output lines of the serial interface when placed in the control mode.

**(1) Operation in control mode**

Port	Alternate-Function Pin	Function in Control Mode	Block Type
Port 12	P120	SO2	Serial interface output
	P121	SI2	Serial interface input
	P122	$\overline{\text{SCK2}}$	Serial interface output
	P123	SO3	Serial interface output
	P124	SI3	Serial interface input
	P125	$\overline{\text{SCK3}}$	Serial interface output
	P126	–	Only for port
	P127	CLO	Clock output

**(2) Setting input/output mode and control mode**

The input/output mode of port 12 is set by the port mode register 12 (PM12). The control mode is set by the port mode control register 12 (PMC12).

**Port 12 mode register (PM12)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PM12	PM127	PM126	PM125	PM124	PM123	PM122	PM121	PM120	Address FFFFF038H	After reset FFH

Bit Position	Bit Name	Function
7 to 0	PM12n (n = 7 to 0)	Port Mode Set P12n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer ON)

**Port 12 mode control register (PMC12)**

Port 12 can be read/written in 8- or 1-bit units. However, bit 6 is fixed to “0” by hardware and ignored if “1” is written in.

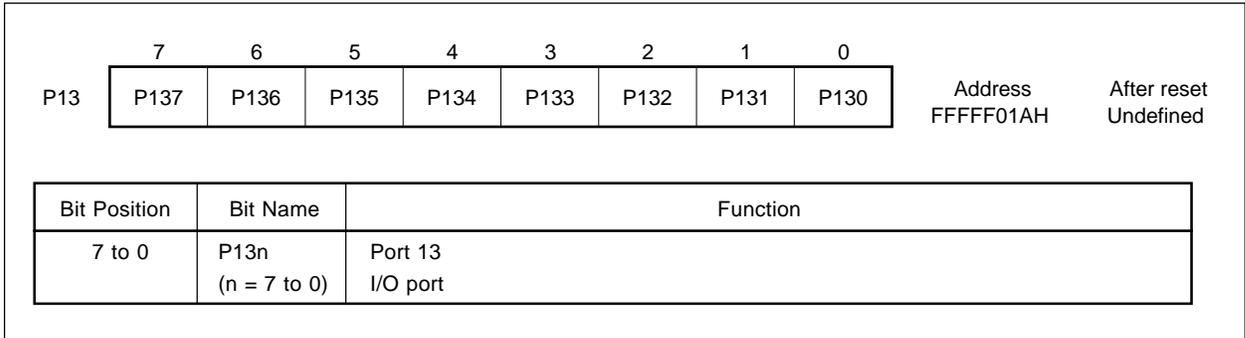
	7	6	5	4	3	2	1	0		
PMC12	PMC127	0	PMC125	PMC124	PMC123	PMC122	PMC121	PMC120	Address FFFFF058H	After reset 00H

Bit Position	Bit Name	Function
7	PMC127	Port Mode Control Sets P127 pin in input/output mode. 0: I/O port mode 1: CLO output mode
5	PMC125	Port Mode Control Sets P125 pin in input/output mode. 0: I/O port mode 1: SCK3 input/output mode
4	PMC124	Port Mode Control Sets P124 pin in input/output mode. 0: I/O port mode 1: SI3 input mode
3	PMC123	Port Mode Control Sets P123 pin in input/output mode. 0: I/O port mode 1: SO3 output mode
2	PMC122	Port Mode Control Sets P122 pin in input/output mode. 0: I/O port mode 1: SCK2 input/output mode
1	PMC121	Port Mode Control Sets P121 pin in input/output mode. 0: I/O port mode 1: SI2 input mode
0	PMC120	Port Mode Control Sets P120 pin in input/output mode. 0: I/O port mode 1: SO2 output mode

**12.3.13 Port 13**

Port 13 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.



In addition to the function as a port, this port can also be used as the output of real time output port.

**(1) Operation in control mode**

Port	Alternate-Function Pin	Function in Control Mode	Block Type
Port 13	P130 to P137	RTP0 to RTP7	A

**(2) Setting input/output mode and control mode**

The input/output mode of port 13 is set by port mode register 13 (PM13). The control mode is set by port mode control register 13 (PMC13).

**Port 13 mode register (PM13)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PM13	PM137	PM136	PM135	PM134	PM133	PM132	PM131	PM130	Address FFFFFF03AH	After reset FFH

Bit Position	Bit Name	Function
7 to 0	PM13n (n = 7 to 0)	Port Mode Sets P13n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Port 13 mode control register (PMC13)**

This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
PMC13	PMC137	PMC136	PMC135	PMC134	PMC133	PMC132	PMC131	PMC130	Address FFFFF05AH	After reset 00H

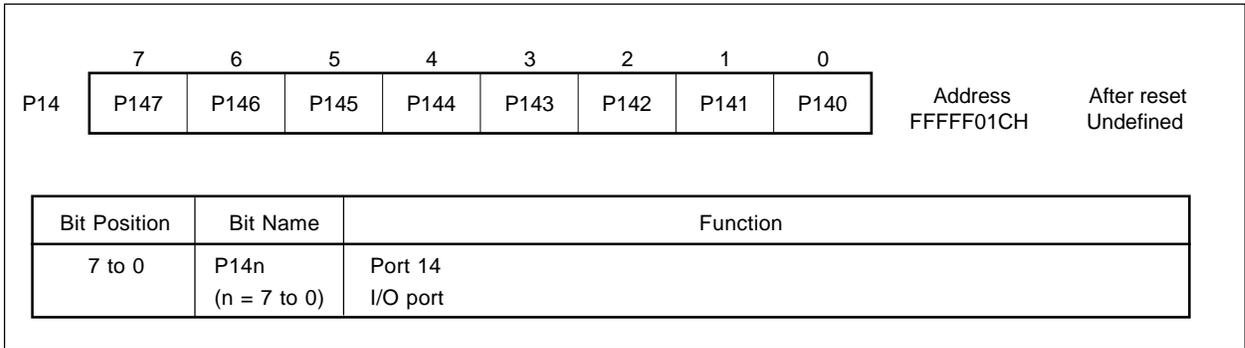
  

Bit Position	Bit Name	Function
7 to 0	PMC13n (n = 7 to 0)	Port Mode Control Sets P13n pin in input/output mode. 0: I/O port mode 1: RTPn output mode

**Caution** When each bit is set to 1, the output buffer of the corresponding port is turned on, regardless of the contents of the PM13 register, and the contents of the RTP register are output. Therefore, initialize the contents of the RTP register before setting each bit to 1.

**12.3.14 Port 14**

Port 14 is an 8-bit input/output port that can be set in the input or output mode in 1-bit units.



Port 14 can be used only as a port.

**(1) Operation in control mode**

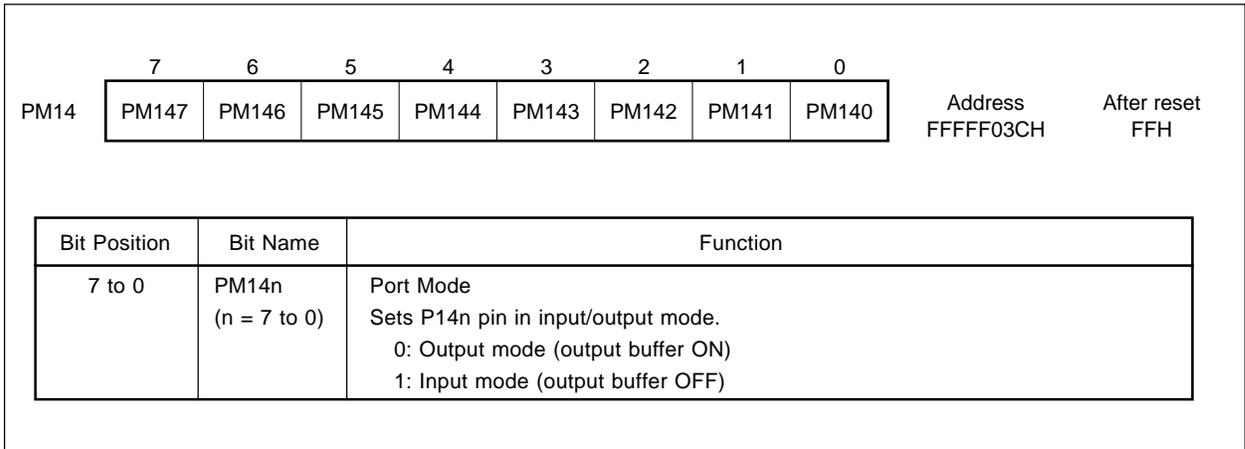
Port	Alternate-Function Pin	Function in Control Mode	Block Type
Port 14	P140 to P147	-	Port only

**(2) Setting input/output mode and control mode**

The input/output mode of port 14 is set by port mode register 14 (PM14). Port 14 does not have port mode control register 14 (PMC14) because it is not provided with the control mode.

**Port 14 mode register (PM14)**

This register can be read/written in 8- or 1-bit units.



**[MEMO]**

## CHAPTER 13 RESET FUNCTION

When the low-level is input to the  $\overline{\text{RESET}}$  pin, the system is reset and each on-chip hardware is initialized to the initial state.

When the  $\overline{\text{RESET}}$  pin changes from low-level to high-level, the reset state is released and the CPU starts executing the program. Initialize the contents of each register in the program as necessary.

### 13.1 Features

- Analog noise elimination circuit (delay of approx. 60 ns) provided on reset pin

### 13.2 Pin Function

- ★ During the reset state, all the pins (except CLKOUT,  $\overline{\text{RESET}}$ , X2, V<sub>DD</sub>, V<sub>SS</sub>, CV<sub>DD</sub>, CV<sub>SS</sub>, AV<sub>DD</sub>, AV<sub>SS</sub> and AV<sub>REF</sub> pins) are in the high-impedance state.

When an external memory is connected, a pull-up (or pull-down) resistor must be connected to each pin of ports 4, 5, 6, and 9. Otherwise, the memory contents may be lost if these pins go into a high-impedance state.

Also treat signal outputs of the on-chip peripheral I/O function and the output port so that they will not be affected.

In the ROM-less mode and single-chip mode 2, the CLKOUT signal is output even during the reset period (low level output). In single-chip mode 1, the CLKOUT signal is not output until the PSC register is set. In the flash memory programming mode, the CLKOUT signal is not output (low level output).

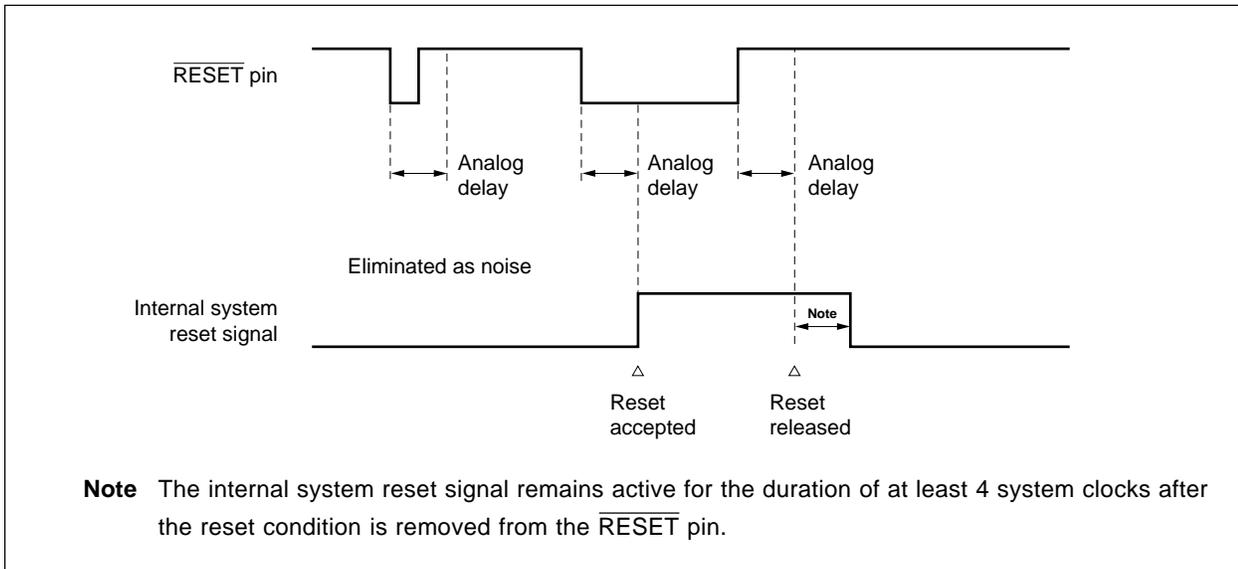
Table 13-1 shows the operating status of each pin during the reset period.

**Table 13-1. Operating Status of I/O and Output Pins During Reset Period**

I/O or Output Pin	Pin Status				
	In Single-Chip Mode 1	In Single-Chip Mode 2	In ROM-less Mode 1	In ROM-less Mode 2	In Flash Memory Programming Mode
P00 to P07, P10 to P17, P21 to P26, P30 to P37, P95, P96, P100 to P103, P110 to P117, P120 to P127, P130 to P137, P140 to P147	(Input)				Hi-Z
P40 to P47, P50 to P57, P69 to P67, P90 to P94	(Input)		(Control Mode)		
AD0 to AD15, A16 to A23	(Port mode)		Hi-Z		
$\overline{\text{LBEN}}$	(P90)		Hi-Z	$\overline{\text{WRL}}$	
$\overline{\text{WRL}}$	(P90)		$\overline{\text{LBEN}}$	Hi-Z	
$\overline{\text{UBEN}}$	(P91)		Hi-Z		
$\overline{\text{R/W}}$	(P92)		Hi-Z	$\overline{\text{WRH}}$	
$\overline{\text{WRH}}$	(P92)		$\overline{\text{R/W}}$	Hi-Z	
$\overline{\text{DSTB}}$	(P93)		Hi-Z	$\overline{\text{RD}}$	
$\overline{\text{RD}}$	(P93)		$\overline{\text{DSTB}}$	Hi-Z	
ASTB	(P94)		Hi-Z		
$\overline{\text{HLD\!AK}}$	(Port mode)				
CLKOUT	L	Operates			L
SO0, SO2, TXD	(Port mode)				Operation
TO00, TO01, TO20 to TO24, RTP0 to RTP7, SDA, SDL, SO1, SO3, PWM0 to PWM3	(Port mode)				Hi-Z

**Remark** Hi-Z : High impedance  
 L : Low-level output

**(1) Accepting reset signal**



**(2) Power-ON reset**

For the reset operations at power-on it is necessary to secure an oscillation stabilization time of 10 ms or more from when the power supply starts until reset is accepted by the low-level width of the  $\overline{\text{RESET}}$  pin. Furthermore,

★

it is also necessary to secure oscillation stabilization time when an external clock is used in the direct mode.

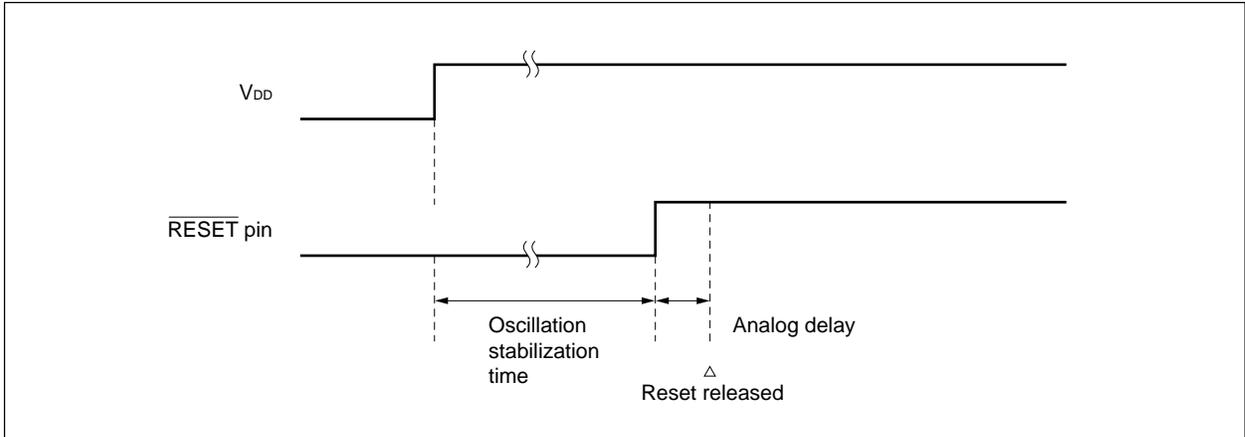
**13.3 Initialize**

Table 13-2 shows the initial value of each register after reset.

The contents of the registers must be initialized in the program as necessary. Especially, set the following registers as necessary because they are related to system setting:

- Power save control register (PSC) ... X1 and X2 pin function, CLKOUT pin operation, etc.
- Data wait control register (DWC) ... Number of data wait states

Table 13-2. Initial Values after Reset of Each Register (1/2)

Register		Initial Value after Reset
r0		00000000H
r1 to r31		Undefined
PC		00000000H
PSW		00000020H
EIPC		Undefined
EIPSW		Undefined
FEPC		Undefined
FEPSW		Undefined
ECR		00000000H
Internal RAM		Undefined
Port	Output latch (P0 to P6, P9 to P14)	Undefined
	Mode register (PM0 to PM6, PM9 to PM14)	FFH
	Mode control register (PMC0, PMC1, PMC3, PMC10 to PMC13) (PMC2)	00H 01H
	Memory expansion mode register (MM)	00H/07H
Clock generator	System status register (SYS)	0000000xB
Clock output	Clock output mode register (CLOM)	00H
Real-time pulse unit	Timer control register (TMC01, TMC02) (TMC00, TMC1, TMC20 to TMC24, TMC3)	00H 01H
	Timer output control register (TOC0, TOC1)	00H
	Capture/compare register (CC00 to CC03, CC00L to CC03L, CC3)	Undefined
	Compare register (CM10, CM11, CM10L, CM11L, CM20 to CM24)	Undefined
	Capture register (CP10 to CP13, CP10L to CP13L, CP3)	Undefined
	Timer register (TM0, TM1, TM0L, TM1L, TM20 to TM24, TM3)	0000H
	Timer overflow status register (TOVS)	00H
A/D converter	A/D converter mode register (ADM0, ADM1)	07H
	A/D conversion result register (ADCR0 to ADCR7)	Undefined

★

**Caution** “Undefined” means an undefined value due to power-on reset or data corruption when a falling edge of  $\overline{\text{RESET}}$  coincides with a data write operation. The previous status of data is retained by a falling edge of  $\overline{\text{RESET}}$  in cases other than the above.

Table 13-2. Initial Values after Reset of Each Register (2/2)

	Register	Initial Value after Reset
Real-time output function	Port 13 buffer register (PB)	Undefined
	Output latch register (RTP)	Undefined
Serial interface	Asynchronous serial interface mode register (ASIM0)	80H
	Asynchronous serial interface mode register (ASIM1)	00H
	Asynchronous serial interface status register (ASIS)	00H
	Receive buffer (RXB, RXBL)	Undefined
	Transmit shift register (TXS, TXSL)	Undefined
	Clocked serial interface mode register (CSIM0 to CSIM3)	00H
	Serial I/O shift register (SIO0 to SIO3)	Undefined
	Baud rate generator compare register (BRGC0 to BRGC3)	Undefined
	Baud rate generator prescaler mode register (BPRM0 to BPRM3)	00H
	IIC control register (IICC)	00H
	IIC status register (IICS)	00H
	IIC clock selection register (IICCL)	00H
	IIC shift register (IIC)	00H
	Slave address register (SVA)	00H
PWM	PWM control register (PWMC0 to PWMC3)	05H
	PWM prescaler register (PWPR0 to PWPR3)	00H
	PWM modulo register (PWM0 to PWM3)	Undefined
Interrupt/exception processing function	Interrupt control register (xxICn)	47H
	In-service priority register (ISPR)	00H
	External interrupt mode register (INTM0 to INTM7)	00H
	Event divide counter (EDV0 to EDV2)	00H
	Event divide control register (EDVC0 to EDVC2)	01H
	Event selection register (EVS)	00H
Memory management function	Data wait control register (DWC)	FFFFH
	Bus cycle control register (BCC)	AAAAH
	System control register (SYC)	00H/01H
Power save control	Command register (PRCMD)	Undefined
	Power save control register (PSC)	00H/C0H
	Clock control register (CKC)	00H

**Caution** “Undefined” means an undefined value due to power-on reset or data corruption when a falling edge of  $\overline{\text{RESET}}$  coincides with a data write operation. The previous status of data is retained by a falling edge of  $\overline{\text{RESET}}$  in cases other than the above.

**[MEMO]**

## CHAPTER 14 FLASH MEMORY ( $\mu$ PD70F3008 AND 70F3008Y ONLY)

The  $\mu$ PD70F3008 and 70F3008Y of the V854 are provided with a 128-Kbyte flash memory. In the instruction fetch to this flash memory, 4 bytes can be accessed by a single clock as well as the mask ROM version.

Writing to a flash memory can be performed with memory mounted on the target system (on board). The dedicated flash writer is connected to the target system to perform writing.

The followings can be considered as the development environment and the applications using a flash memory.

- Software can be altered after the V854 is solder mounted on the target system.
- Small scale production of various models is made easier by differentiating software.
- Data adjustment in starting mass production is made easier.

### 14.1 Features

- 4-byte/1-clock access (in instruction fetch access)
- All area one-shot erase
- Communication through serial interface from the dedicated flash writer
- ★ • Erase/writing voltage:  $V_{PP} = 7.8 \text{ V}$
- On-board programming
- Number of rewrite: 100 times (target)

### 14.2 Writing by Flash Writer

Writing can be performed either on-board or off-board by the dedicated flash writer.

#### (1) On-board programming

The contents of the flash memory is rewritten after the V854 is mounted on the target system. Mount connectors, etc., on the target system to connect the dedicated flash writer.

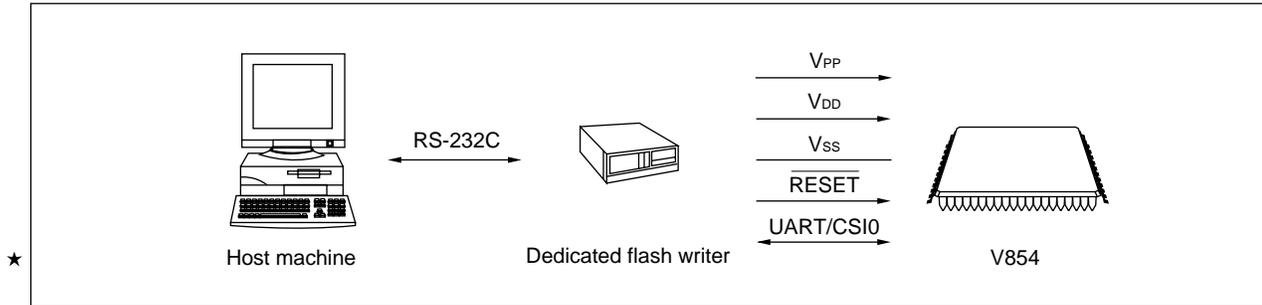
#### (2) Off-board programming

Writing to a flash memory is performed by the dedicated program adapter (FA Series **Note**), etc., before mounting the V854 on the target system.

**Note** FA Series program adapters are made by Naito Densai Machidaseisakusho Co., Ltd.

### 14.3 Programming Environment

The following shows the environment required for writing programs to the flash memory of the V854.



A host machine is required for controlling the dedicated flash writer.

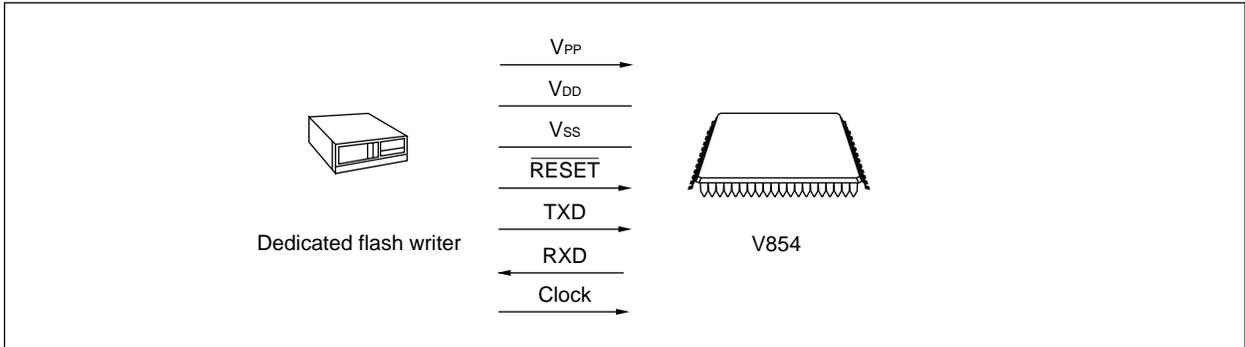
UART or CSI is used as the interface between the dedicated flash writer and the V854 to perform writing, erasing, etc. A dedicated program adapter (FA Series) is required for off-board writing.

## 14.4 Communication System

- ★ The communication between the dedicated flash writer and the V854 is performed by serial communication using UART or CSIO.

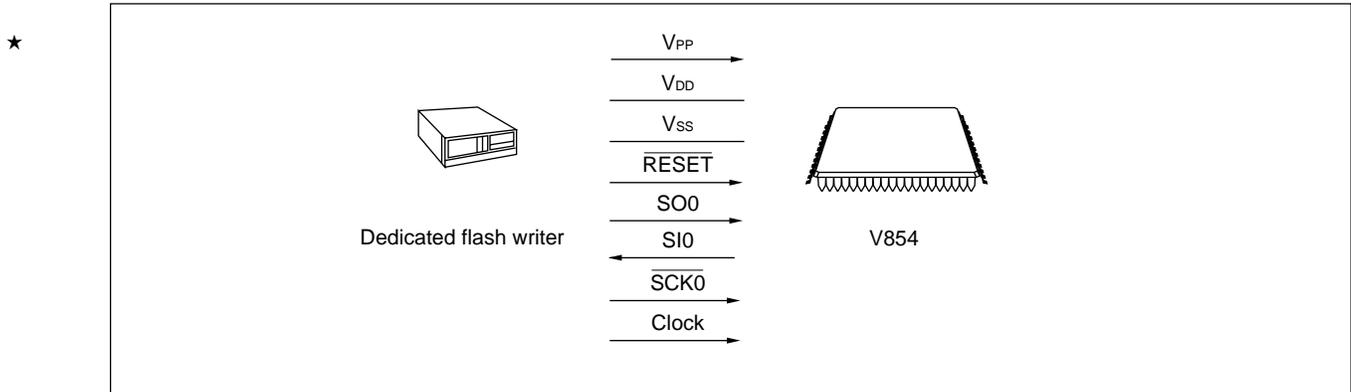
### (1) UART

Transfer rate: 1200 to 76800 bps (LSB first)



### (2) CSIO

Transfer rate: up to 8.25 Mbps (MSB first)



The dedicated flash writer outputs the transfer clock, and the V854 operates as a slave.

When Flashpro II is used as the dedicated flash writer, it generates the following signals to the V854. For the details, refer to the Flashpro II manual.

**Remark** Flashpro II is a product of Naitou Densai Machidaseisakusho Co., Ltd.

Flashproll			V854	Measurement when connected	
Signal Name	I/O	Pin Function	Pin Name	CSIn	UART
V <sub>PP</sub>	Output	Writing voltage	V <sub>PP</sub>	◎	◎
V <sub>DD</sub>	I/O	V <sub>DD</sub> voltage generation/ voltage monitoring	V <sub>DD</sub>	◎	◎
GND	—	Ground	V <sub>SS</sub>	◎	◎
CLK	Output	Clock output to V854	X1 <sup>Note</sup>	○	○
$\overline{\text{RESET}}$	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎
★ SI/RxD	Input	Receive signal	SO0/TxD	◎	◎
★ SO/TxD	Output	Transmit signal	SI0/RxD	◎	◎
★ SCK	Output	Transfer clock	$\overline{\text{SCK0}}$	◎	x

**Note** Only for off-board writing

**Remark** ◎ : Always connected

○ : Does not need to connect, if generated on the target board

x : Does not need to connect

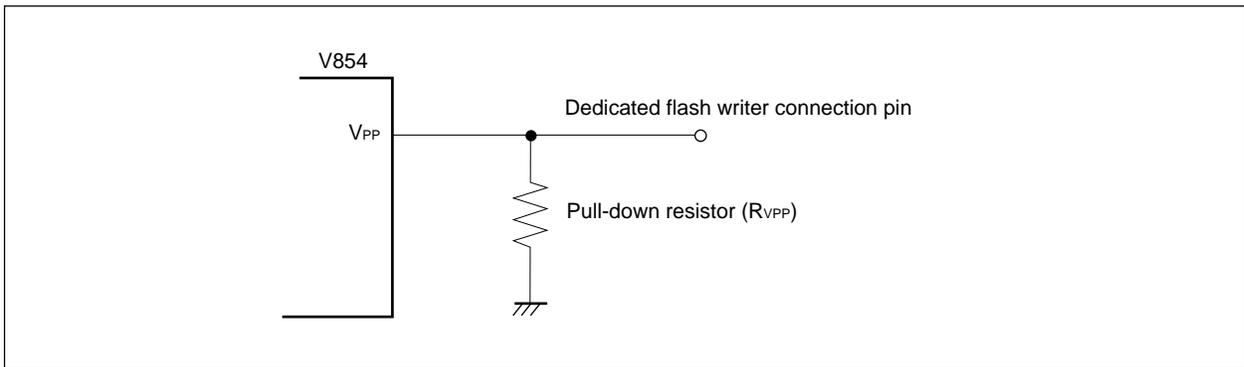
## 14.5 Pin Handling

When performing on-board writing, install a connector on the target system to connect to the dedicated flash writer. Also, install the function on-board to switch from the normal operation mode to the flash memory programming mode.

When switched to the flash memory programming mode, all the pins not used for the flash memory programming become the same status as that immediately after reset of single-chip mode 1. Therefore, all the ports become high-impedance status, so that pin handling is required when the external device does not acknowledge the high-impedance status.

### 14.5.1 $V_{PP}$ pin

- ★ In the normal operation mode, 0 V is input to  $V_{PP}$  pin. In the flash memory programming mode, 7.8-V writing voltage is supplied to  $V_{PP}$  pin. The following shows an example of the connection of  $V_{PP}$  pin.



★ 14.5.2 Serial interface pin

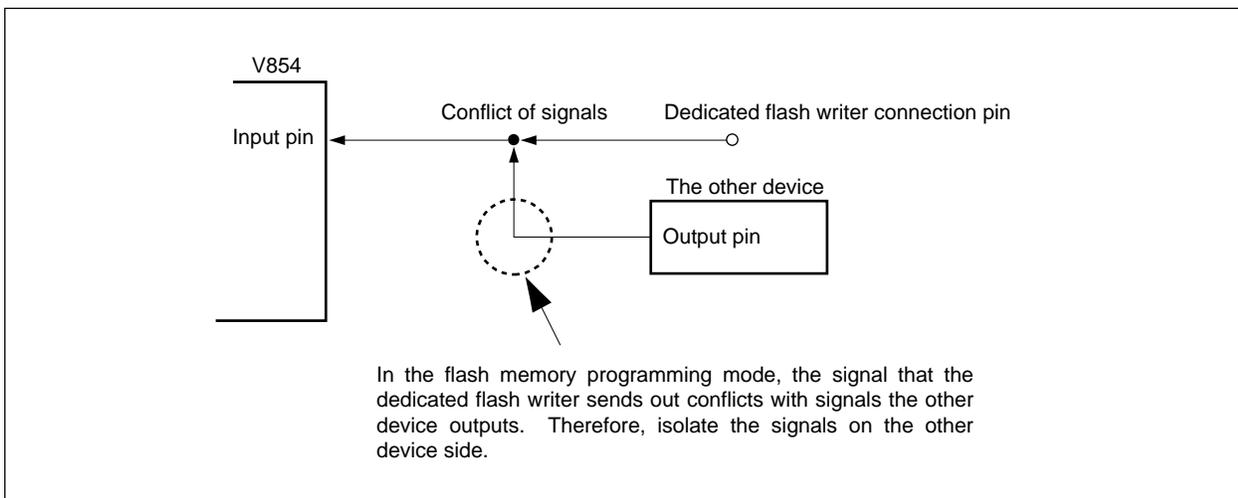
The following shows the pins used by each serial interface.

Serial Interface	Pins Used
CSI0	SO0, SI0, $\overline{\text{SCK0}}$
UART	TXD, RXD

When connecting a dedicated flash writer to a serial interface pin which is connected to other devices on-board, care should be taken to the conflict of signals and the malfunction of other devices, etc.

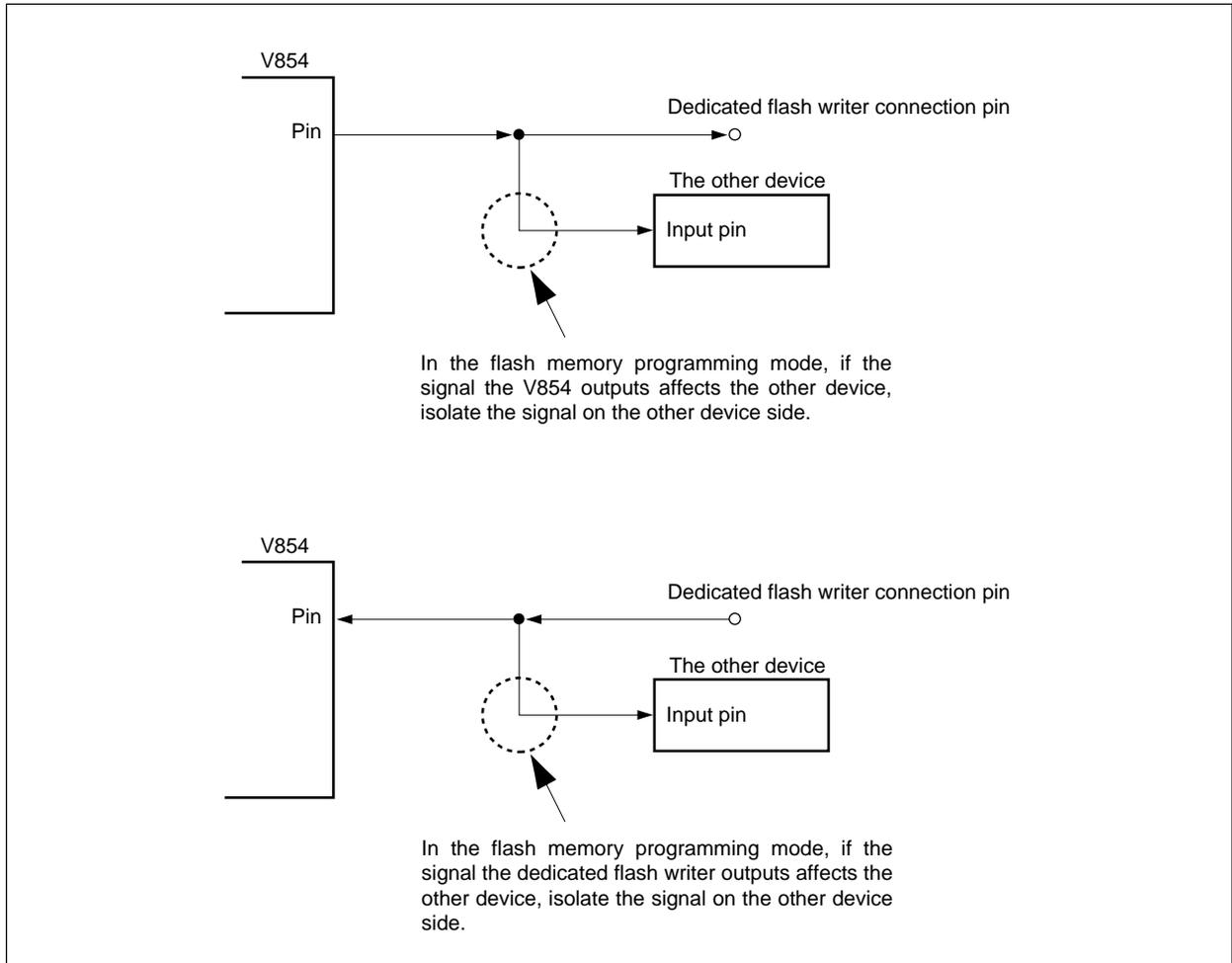
(1) Conflict of signals

When connecting a dedicated flash writer (output) to a serial interface pin (input) which is connected to another device (output), conflict of signals occurs. To avoid the conflict of signals, isolate the connection to the other device or set the other device to the high-impedance status.



**(2) Malfunction of the other device**

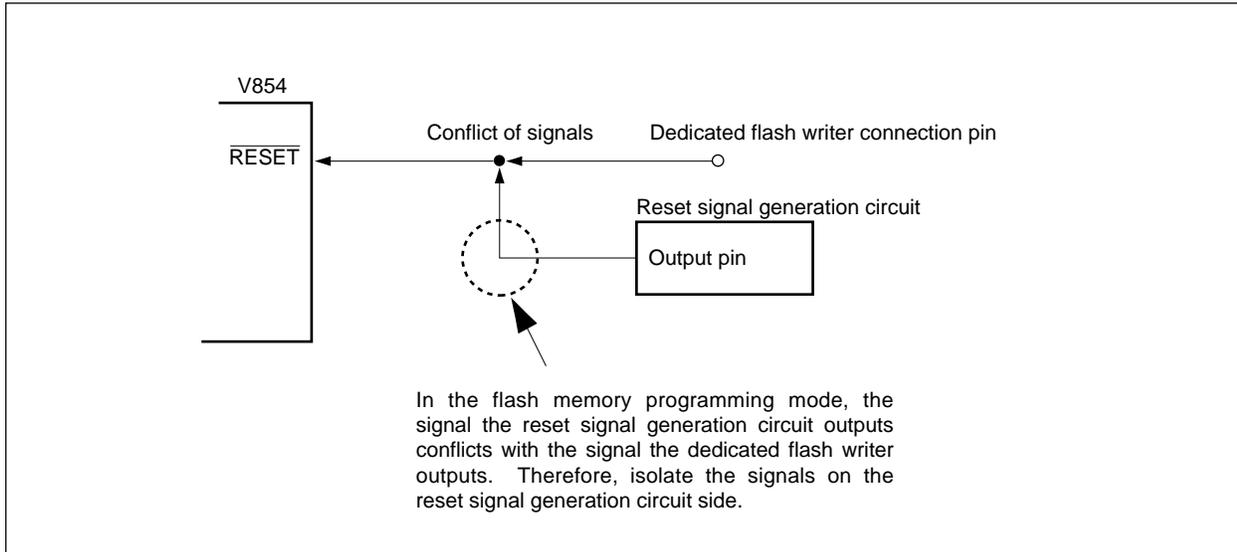
When connecting a dedicated flash writer (output or input) to a serial interface pin (input or output) connected to another device (input), the signal output to the other device may cause the device to malfunction. To avoid this, isolate the connection to the other device or make the setting so that the input signal to the other device is ignored.



### 14.5.3 Reset pin

When connecting the reset signals of the dedicated flash writer to the  $\overline{\text{RESET}}$  pin which is connected to the reset signal generation circuit on-board, conflict of signals occurs. To avoid the conflict of signals, isolate the connection to the reset signal generation circuit.

When reset signal is input from the user system during the flash memory programming mode, programming operation will not be performed correctly. Therefore, do not input signals other than the reset signals from the dedicated flash writer.



### 14.5.4 NMI pin

Do not change the input signal to the NMI pin during the flash memory programming mode. If the NMI pin is changed during the flash memory programming mode, the programming may not be performed correctly.

### 14.5.5 Mode pin

To switch to the flash memory programming mode, change MODE0 through MODE2 to "111" with a jumper, etc., applies writing voltage to  $V_{PP}$  pin, and release the reset.

### 14.5.6 Port pin

When the flash memory programming mode is set, all the port pins except the pins which communicate with the dedicated flash writer become output high-impedance status. The treatment of these port pins are not necessary. If problems such as disabling output high-impedance status should occurs to the external devices connected to the port, connect them to  $V_{DD}$  or  $V_{SS}$  through resistors.

### 14.5.7 Other signal pin

Connect X1, X2, CKSEL, PLLSEL, and  $AV_{REF}$  to the same status as that in the normal operation mode.

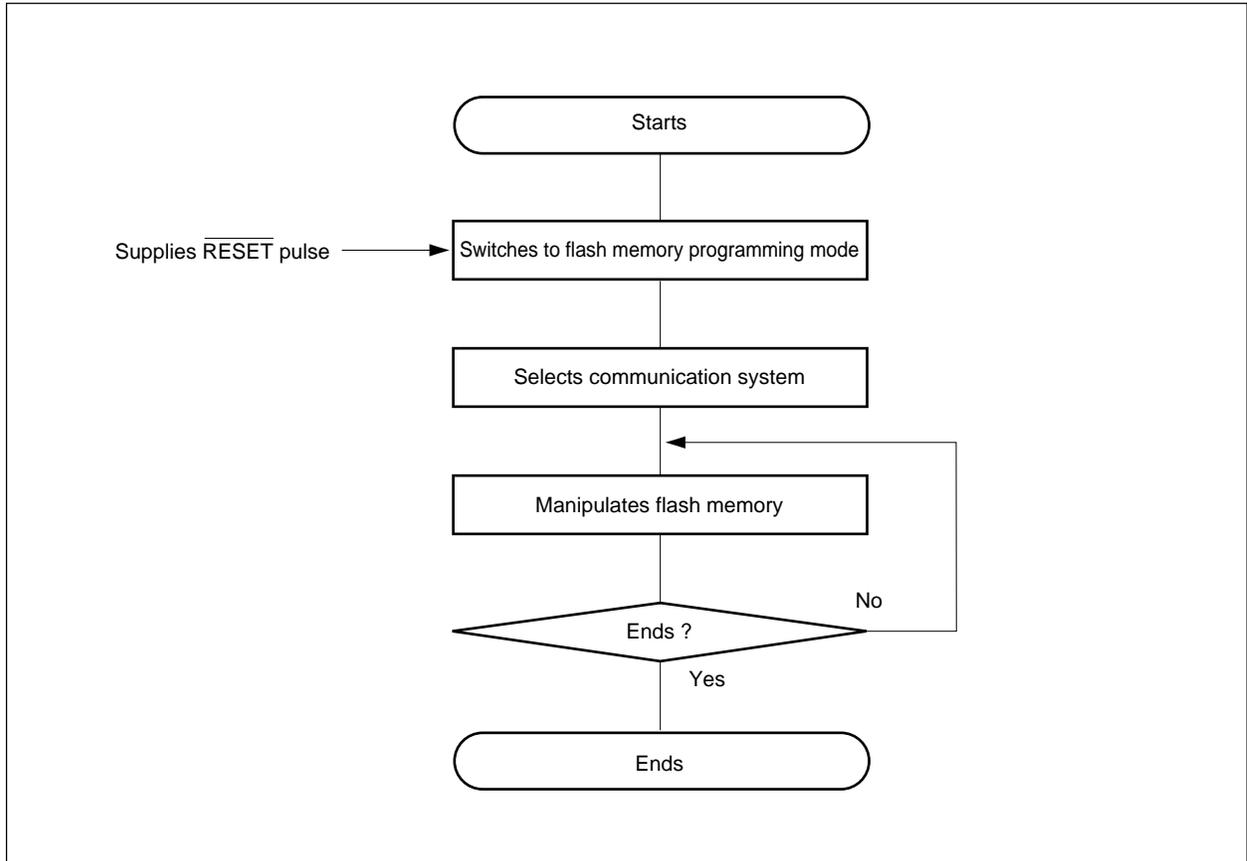
### 14.5.8 Power supply

- ★ Supply the same power supplies ( $V_{DD}$ ,  $V_{SS}$ ,  $AV_{DD}$ ,  $AV_{SS}$ ,  $CV_{DD}$ ,  $CV_{SS}$ ) as those in the normal operation mode. Connect  $V_{DD}$  and GND of the dedicated flash writer to  $V_{DD}$  and  $V_{SS}$ . (The  $V_{DD}$  dedicated flash writer is provided with a power supply monitoring function.)

## 14.6 Programming Method

### 14.6.1 Flash memory control

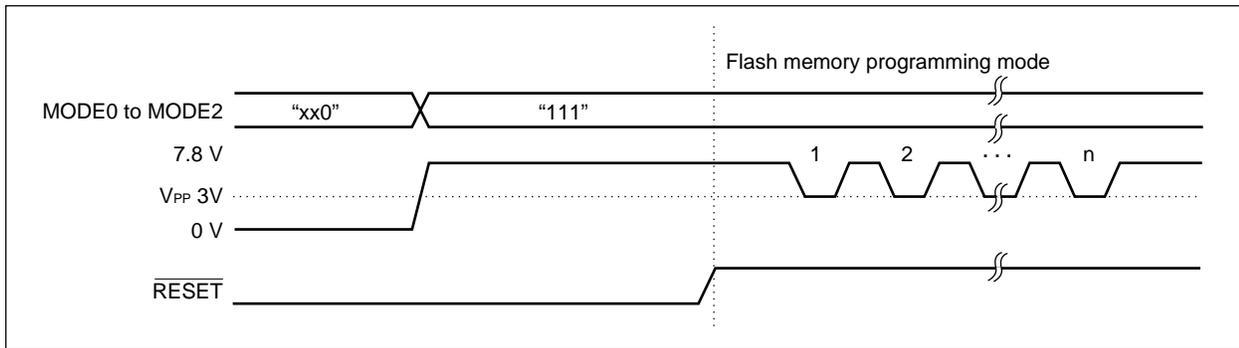
The following shows the procedure that this firmware manipulates the flash memory.



**14.6.2 Flash memory programming mode**

When rewriting the contents of a flash memory using the dedicated flash writer, set the V854 in the flash memory programming mode. When switching to modes, set MODE0 through MODE2 and V<sub>PP</sub> pin before releasing reset. When performing on-board writing, change modes using a jumper, etc.

V <sub>PP</sub>	MODE2	MODE1	MODE0	Operation Mode	
0 V	0	0	0	Normal operation mode	ROM-less mode 1
0 V	0	0	1		ROM-less mode 2
0 V	0	1	0		Single-chip mode 1
0 V	0	1	1		Single-chip mode 2
★ 7.8 V	1	1	1	Flash memory programming mode	
Other than the above				Setting prohibited	



**14.6.3 Selection of communication mode**

In the V854, a communication system is selected by inputting pulse (16 pulses max.) to V<sub>PP</sub> pin after switching to the flash memory programming mode. The V<sub>PP</sub> pulse is generated by the dedicated flash writer. The following shows the relation between the number of pulses and the communication systems.

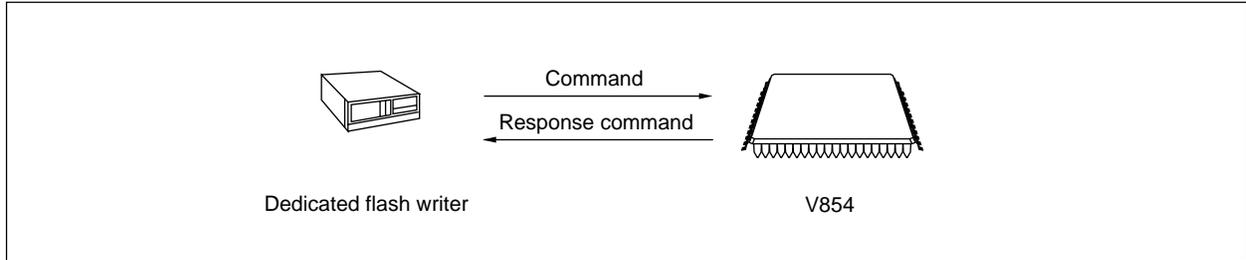
**Table 14-1. List of Communication Systems**

V <sub>PP</sub> pulse	Communication System	Remarks
0	CSI0	V854 performs slave operation, MSB first
8	UART	Communication rate: 9600 bps (at reset), LSB first
Others	RFU	Setting prohibited

**Caution** When UART is selected, the receive clock is calculated based on the reset command sent from the dedicated flash writer after receiving V<sub>PP</sub> pulse.

### 14.6.4 Communication command

The V854 communicates with the dedicated flash writer by means of commands. The command sent from the dedicated flash writer to the V854 is called a “command”. The response signal sent from the V854 to the dedicated flash writer is called a “response command”.



The following shows the command for flash memory control of the V854. All of these commands are issued from the dedicated flash writer, and the V854 performs the various processings corresponding to the commands.

Category	Command Name	Function
★ Verify	One-shot verify command	Compares the contents of the entire memory and the input data.
★ Erase	One-shot erase command	Erases the contents of the entire memory.
★ Blank check	One-shot blank check command	Checks the erase state of the entire memory.
Data write	High-speed write command	Writes data by the specification of the write address and the number of bytes to be written, and executes verify check.
	Continuous write command	Writes data from the address following the high-speed write command executed immediately before, and executes verify check.
★ System setting and control	Status read out command	Acquires the status of operations.
	Oscillating frequency setting command	Sets the oscillating frequency.
	Erasing time setting command	Sets the erasing time of one-shot erase.
	Writing time setting command	Sets the writing time of data write.
	Baud rate setting command	Sets the baud rate when using UART.
	Silicon signature command	Reads out the silicon signature information.
	Reset command	Escapes from each state.

The V854 sends back response commands to the commands issued from the dedicated flash writer. The following shows the response commands the V854 sends out.

Response Command Name	Function
ACK (acknowledge)	Acknowledges command/data, etc.
NAK (not acknowledge)	Acknowledges illegal command/data, etc.

### 14.6.5 Resources used

According to the flash memory programming mode setting, the resources used consist of the area from FFE000H to FFE7FFH of the internal RAM and all the registers. Area FFE800H to FFEFFFH of the internal RAM retains data as long as the power is on. The contents of the registers that are initialized by reset change to the default value.

[MEMO]

## CHAPTER 15 DIFFERENCES BETWEEN VERSIONS

### 15.1 Differences between Versions with I<sup>2</sup>C Function and Versions without I<sup>2</sup>C Function

The product names of the versions with the I<sup>2</sup>C function are  $\mu$ PD703008Y and 70F3008Y. They are identified by the letter “Y” in the product names.

Product Name		$\mu$ PD703008Y, 70F3008Y	$\mu$ PD703006, 703008, 70F3008
Function			
I <sup>2</sup> C function		Available	Not available
	Pin	P33/SO1/SDA P35/SCK1/SCL	P33/SO1 P35/SCK1
	Interrupt	INTIIC	Not available
	Register	Available	Not available (undefined)
	IIC0 IICC IICS IICCL IIC SAV		

### 15.2 Differences between On-chip Flash Memory Versions, On-chip Mask ROM Versions, and ROM-less Versions

The product names of the versions with the on-chip flash memory are  $\mu$ PD70F3008 and 70F3008Y. They are identified by the letter “F” in the product names.

Part Number	$\mu$ PD703006	$\mu$ PD703008, 703008Y	$\mu$ PD70F3008, 70F3008Y
Parameter			
On-chip ROM	None	Mask ROM	Flash memory
Flash memory programming pin	None		Provided ( $V_{PP}$ )
Flash memory programming mode	None		Provided ( $V_{PP} = 7.8$ V, MODE0 to MODE2 = High-level)
Electrical specifications	Current consumption, etc. differ. (Refer to each product data sheet.)		
Others	Noise immunity and noise radiation differ for each product depending on the circuit size and mask layout.		

- ★
- Cautions**
1. There are differences in noise immunity and noise radiation between the flash memory version, mask ROM version, and ROM-less version. 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 set using consumer samples (not engineering samples) of the mask ROM version.
  2. When replacing the flash memory version with the mask ROM version, be sure to write the same code into the internal ROM's reserved area.

[MEMO]

## APPENDIX A REGISTER INDEX

(1/6)

Symbol	Name	Unit	Page
ADCR0	A/D conversion result register 0	ADC	299
ADCR1	A/D conversion result register 1	ADC	299
ADCR2	A/D conversion result register 2	ADC	299
ADCR3	A/D conversion result register 3	ADC	299
ADCR4	A/D conversion result register 4	ADC	299
ADCR5	A/D conversion result register 5	ADC	299
ADCR6	A/D conversion result register 6	ADC	299
ADCR7	A/D conversion result register 7	ADC	299
ADIC0	Interrupt control register	INTC	116
ADM0	A/D converter mode register 0	ADC	297
ADM1	A/D converter mode register 1	ADC	299
ASIM0	Asynchronous serial interface mode register 0	UART	209
ASIM1	Asynchronous serial interface mode register 1	UART	211
ASIS	Asynchronous serial interface status register 1	UART	212
BCC	Bus cycle control register	BCU	88
BPRM0	Baud rate generator prescaler mode register 0	BRG	292
BPRM1	Baud rate generator prescaler mode register 1	BRG	292
BPRM2	Baud rate generator prescaler mode register 2	BRG	292
BPRM3	Baud rate generator prescaler mode register 3	BRG	292
BRGC0	Baud rate generator compare register 0	BRG	291
BRGC1	Baud rate generator compare register 1	BRG	291
BRGC2	Baud rate generator compare register 2	BRG	291
BRGC3	Baud rate generator compare register 3	BRG	291
CC00	Capture/compare register 00	RPU	161
CC00L	Capture/compare register 00L	RPU	161
CC01	Capture/compare register 01	RPU	161
CC01L	Capture/compare register 01L	RPU	161
CC02	Capture/compare register 02	RPU	161
CC02L	Capture/compare register 02L	RPU	161
CC03	Capture/compare register 03	RPU	161
CC03L	Capture/compare register 03L	RPU	161
CC0IC0	Interrupt control register	INTC	116
CC0IC1	Interrupt control register	INTC	116
CC0IC2	Interrupt control register	INTC	116
CC0IC3	Interrupt control register	INTC	116
CC3	Capture/compare register 3	RPU	165
CKC	Clock control register	CG	137
CLOM	Clock output mode register	CG	154

Symbol	Name	Unit	Page
CM10	Compare register 10	RPU	163
CM10L	Compare register 10L	RPU	163
CM11	Compare register 11	RPU	163
CM11L	Compare register 11L	RPU	163
CM20	Compare register 20	RPU	164
CM21	Compare register 21	RPU	164
CM22	Compare register 22	RPU	164
CM23	Compare register 23	RPU	164
CM24	Compare register 24	RPU	164
CM1IC0	Interrupt control register	INTC	116
CM1IC1	Interrupt control register	INTC	116
CM2IC0	Interrupt control register	INTC	116
CM2IC1	Interrupt control register	INTC	116
CM2IC2	Interrupt control register	INTC	116
CM2IC3	Interrupt control register	INTC	116
CM2IC4	Interrupt control register	INTC	116
CC3IC0	Interrupt control register	INTC	116
CP10	Capture register 10	RPU	162
CP10L	Capture register 10L	RPU	162
CP11	Capture register 11	RPU	162
CP11L	Capture register 11L	RPU	162
CP12	Capture register 12	RPU	162
CP12L	Capture register 12L	RPU	162
CP13	Capture register 13	RPU	162
CP13L	Capture register 13L	RPU	162
CP3	Capture register 3	RPU	165
CSIC0	Interrupt control register	INTC	116
CSIC1	Interrupt control register	INTC	116
CSIC2	Interrupt control register	INTC	116
CSIC3	Interrupt control register	INTC	116
CSIM0	Clocked serial interface mode register 0	CSI	222
CSIM1	Clocked serial interface mode register 1	CSI	222
CSIM2	Clocked serial interface mode register 2	CSI	222
CSIM3	Clocked serial interface mode register 3	CSI	222
DWC	Data wait control register	BCU	86
ECR	Interrupt source register	CPU	52
EDVC0	Event divide control register 0	INTC	125
EDVC1	Event divide control register 1	INTC	125
EDVC2	Event divide control register 2	INTC	125
EDV0	Event divide counter 0	INTC	125

Symbol	Name	Unit	Page
EDV1	Event divide counter 1	INTC	125
EDV2	Event divide counter 2	INTC	125
EIPC	Interrupt status save register	CPU	52
EIPSW	Interrupt status save register	CPU	52
EVS	Event selection register	INTC	125
FEPC	NMI status save register	CPU	52
FEPSW	NMI status save register	CPU	52
IICC	IIC control register	I <sup>2</sup> C	237
IICCL	IIC clock selection register	I <sup>2</sup> C	241
IICS	IIC status register	I <sup>2</sup> C	239
IIC	IIC shift register	I <sup>2</sup> C	241
SVA	Slave address register	I <sup>2</sup> C	242
IIIC0	Interrupt control register	INTC	116
INTM0	External interrupt mode register 0	INTC	106
INTM1	External interrupt mode register 1	INTC	121
INTM2	External interrupt mode register 2	INTC	121
INTM3	External interrupt mode register 3	INTC	121
INTM4	External interrupt mode register 4	INTC	121
INTM5	External interrupt mode register 5	INTC	121
INTM6	External interrupt mode register 6	INTC	121
INTM7	External interrupt mode register 7	INTC	123
ISPR	In-service priority register	INTC	118
MM	Memory expansion mode register	Port	68
OVIC0	Interrupt control register	INTC	116
OVIC1	Interrupt control register	INTC	116
P0	Port 0	Port	348
P1	Port 1	Port	350
P2	Port 2	Port	352
P3	Port 3	Port	354
P4	Port 4	Port	357
P5	Port 5	Port	359
P6	Port 6	Port	361
P7	Port 7	Port	363
P8	Port 8	Port	363
P9	Port 9	Port	364
P10	Port 10	Port	366
P11	Port 11	Port	368
P12	Port 12	Port	371
P13	Port 13	Port	373
P14	Port 14	Port	375

Symbol	Name	Unit	Page
P1IC0	Interrupt control register	INTC	116
P1IC1	Interrupt control register	INTC	116
P1IC2	Interrupt control register	INTC	116
P1IC3	Interrupt control register	INTC	116
P5IC0	Interrupt control register	INTC	116
P5IC1	Interrupt control register	INTC	116
P5IC2	Interrupt control register	INTC	116
P5IC3	Interrupt control register	INTC	116
PB	Port 13 buffer register	Port	322
PM0	Port 0 mode register	Port	349
PM1	Port 1 mode register	Port	351
PM2	Port 2 mode register	Port	353
PM3	Port 3 mode register	Port	355
PM4	Port 4 mode register	Port	358
PM5	Port 5 mode register	Port	360
PM6	Port 6 mode register	Port	362
PM9	Port 9 mode register	Port	365
PM10	Port 10 mode register	Port	367
PM11	Port 11 mode register	Port	369
PM12	Port 12 mode register	Port	371
PM13	Port 13 mode register	Port	374
PM14	Port 14 mode register	Port	375
PMC0	Port 0 mode control register	Port	349
PMC1	Port 1 mode control register	Port	351
PMC2	Port 2 mode control register	Port	353
PMC3	Port 3 mode control register	Port	356
PMC10	Port 10 mode control register	Port	367
PMC11	Port 11 mode control register	Port	370
PMC12	Port 12 mode control register	Port	372
PMC13	Port 13 mode control register	Port	374
PRCMD	Command register	CG	78
PSC	Power save control register	CG	142
PSW	Program status word	CPU	52, 106, 118, 128
PWM0	PWM modulo register 0 (12 bits)	PWM	329
PWM1	PWM modulo register 1 (12 bits)	PWM	329
PWM2	PWM modulo register 2	PWM	329
PWM3	PWM modulo register 3	PWM	329
PWMC0	PWM control register 0	PWM	327
PWMC1	PWM control register 1	PWM	327
PWMC2	PWM control register 2	PWM	327

Symbol	Name	Unit	Page
PWMC3	PWM control register 3	PWM	327
PWPR0	PWM prescaler register 0	PWM	328
PWPR1	PWM prescaler register 1	PWM	328
PWPR2	PWM prescaler register 2	PWM	328
PWPR3	PWM prescaler register 3	PWM	328
RTP	Output latch register	RPU	322
RXB	Receive buffer (9 bits)	UART	213
RXBL	Receive buffer L (lower 8 bits)	UART	213
SEIC0	Interrupt control register	INTC	116
SIO0	Serial I/O shift register 0	CSI	223
SIO1	Serial I/O shift register 1	CSI	223
SIO2	Serial I/O shift register 2	CSI	223
SIO3	Serial I/O shift register 3	CSI	223
SRIC0	Interrupt control register	INTC	116
STIC0	Interrupt control register	INTC	116
SVA	I <sup>2</sup> C slave address register	I <sup>2</sup> C	242
SYC	System control register	BCU	82
SYS	System status register	CG	79, 139
TM0	Timer 0	RPU	161
TM0L	Timer 0L	RPU	161
TM1	Timer 1	RPU	162
TM1L	Timer 1L	RPU	162
TM20	Timer 20	RPU	164
TM21	Timer 21	RPU	164
TM22	Timer 22	RPU	164
TM23	Timer 23	RPU	164
TM24	Timer 24	RPU	164
TM3	Timer 3	RPU	165
TMC00	Timer control register 00	RPU	166
TMC01	Timer control register 01	RPU	167
TMC02	Timer control register 02	RPU	168
TMC1	Timer control register 1	RPU	169
TMC20	Timer control register 20	RPU	170
TMC21	Timer control register 21	RPU	170
TMC22	Timer control register 22	RPU	170
TMC23	Timer control register 23	RPU	170
TMC24	Timer control register 24	RPU	170
TMC3	Timer control register 3	RPU	171
TOC0	Timer output control register 0	RPU	172
TOC1	Timer output control register 1	RPU	172

Symbol	Name	Unit	Page
TOVS	Timer overflow status register	RPU	173
TXS	Transmit shift register (9 bits)	UART	214
TXSL	Transmit shift register L (lower 8 bits)	UART	214

## APPENDIX B INSTRUCTION SET LIST

### Legend

#### (1) Symbols used for operand description

Symbol	Description
reg1	General register (r0 to r31): Used as source register
reg2	General register (r0 to r31): Mainly used as destination register
immx	x-bit immediate
dispx	x-bit displacement
regID	System register number
bit#3	3-bit data for bit number specification
ep	Element pointer (r30)
cccc	4-bit data for condition code
vector	5-bit data for trap vector number (00H to 1FH)

#### (2) Symbols used for operation description

Symbol	Description
←	Assignment
GR[ ]	General register
SR[ ]	System register
zero-extend(n)	Zero-extends n to word length
sign-extend(n)	Sign-extends n to word length
load-memory(a,b)	Reads data of size b from address a
store-memory(a,b,c)	Writes data b of size c to address a
load-memory-bit(a,b)	Reads bit b of address a
store-memory-bit(a,b,c)	Writes c to bit b of address a
saturated(n)	Performs saturated processing of n (n is 2's complement). If n is $n \geq 7FFFFFFH$ as result of calculation, $7FFFFFFH$ . If n is $n \leq 80000000H$ as result of calculation, $80000000H$ .
result	Reflects result on flag
Byte	Byte (8 bits)
Halfword	Half-word (16 bits)
Word	Word (32 bits)
+	Add
-	Subtract
	Bit concatenation
x	Multiply
÷	Divide
AND	Logical product
OR	Logical sum

Symbol	Description
XOR	Exclusive logical sum
NOT	Logical negate
logically shift left by	Logical left shift
logically shift right by	Logical right shift
arithmetically shift right by	Arithmetic right shift

**(3) Symbols used for execution clock description**

Symbol	Description
i: issue	To execute another instruction immediately after instruction execution
r: repeat	To execute same instruction immediately after instruction execution
l: latency	To reference result of instruction execution by the next instruction

**(4) Flag operation**

Identifier	Description
(Blank)	Not affected
0	Cleared to 0
1	Set to 1
x	Set or cleared according to result
R	Previously saved value is restored

**Condition code**

Condition Name (cond)	Condition Code (cccc)	Conditional Expression	Description
V	0 0 0 0	OV = 1	Overflow
NV	1 0 0 0	OV = 0	No overflow
C/L	0 0 0 1	CY = 1	Carry Lower (Less than)
NC/NL	1 0 0 1	CY = 0	No carry No lower (Greater than or equal)
Z/E	0 0 1 0	Z = 1	Zero Equal
NZ/NE	1 0 1 0	Z = 0	Not zero Not equal
NH	0 0 1 1	(CY or Z) = 1	Not higher (Less than or equal)
H	1 0 1 1	(CY or Z) = 0	Higher (Greater than)
N	0 1 0 0	S = 1	Negative
P	1 1 0 0	S = 0	Positive
T	0 1 0 1	—	Always (unconditional)
SA	1 1 0 1	SAT = 1	Saturated
LT	0 1 1 0	(S xor OV) = 1	Less than signed
GE	1 1 1 0	(S xor OV) = 0	Greater than or equal signed
LE	0 1 1 1	((S xor OV) or Z) = 1	Less than or equal signed
GT	1 1 1 1	((S xor OV) or Z) = 0	Greater than signed

Instruction Set (alphabetical order) (1/4)

Mnemonic	Operand	Code	Operation	Execution Clock			Flag					
				i	r	l	CY	OV	S	Z	SAT	
ADD	reg1, reg2	r r r r r 001110RRRRR	GR[reg2]←GR[reg2]+GR[reg1]	1	1	1	x	x	x	x		
	imm5, reg2	r r r r r 010010 i i i i i	GR[reg2]←GR[reg2]+sign-extend(imm5)	1	1	1	x	x	x	x		
ADDI	imm16, reg1, reg2	r r r r r 110000RRRRR i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1]+sign-extend(imm16)	1	1	1	x	x	x	x		
AND	reg1, reg2	r r r r r 001010RRRRR	GR[reg2]←GR[reg2]AND GR[reg1]	1	1	1		0	x	x		
ANDI	imm16, reg1, reg2	r r r r r 110110RRRRR i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1]AND zero-extend(imm16)	1	1	1		0	0	x		
Bcond	disp9	d d d d d 1011 d d c c c c c  <b>Note 1</b>	if conditions are satisfied	When condition satisfied	3	3	3					
			then PC←PC+sign-extned(disp9)	When condition not satisfied	1	1	1					
CLR1	bit#3, disp16[reg1]	10 b b b 111110RRRRR d d d d d d d d d d d d d d d d	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr, bit#3)) Store-memory-bit(adr, bit#3.0)	4	4	4					x	
CMP	reg1, reg2	r r r r r 001111RRRRR	result←GR[reg2]−GR[reg1]	1	1	1	x	x	x	x		
	imm5, reg2	r r r r r 010011 i i i i i	result←GR[reg2]−sign-extend(imm5)	1	1	1	x	x	x	x		
DI		0000011111100000 0000000101100000	PSW.ID←1 (Maskable interrupt disabled)	1	1	1						
DIVH	reg1, reg2	r r r r r 000010RRRRR	GR [reg2]←GR [reg2]+GR [reg1] <sup>Note 2</sup> (signed division)	36	36	36		x	x	x		
EI		1000011111100000 0000000101100000	PSW.ID←0 (Maskable interrupt enabled)	1	1	1						
HALT		0000011111100000 0000000100100000	Stops	1	1	1						
JARL	disp22, reg2	r r r r r 11110 d d d d d d d d d d d d d d d d d d d d d d 0 <b>Note 3</b>	GR[reg2]←PC+4 PC←PC+sign-extend(disp22)	3	3	3						
JMP	[reg1]	0000000011RRRRR	PC←GR[reg1]	3	3	3						
JR	disp22	0000011110 d d d d d d d d d d d d d d d d d d d d d d 0 <b>Note 3</b>	PC←PC+sign-extend(disp22)	3	3	3						
LD.B	disp16[reg1], reg2	r r r r r 111000RRRRR d d d d d d d d d d d d d d d d	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr, Byte))	1	1	2						
LD.H	disp16[reg1], reg2	r r r r r 111001RRRRR d d d d d d d d d d d d d d d 0 <b>Note 4</b>	adr←GR[reg1]+sign-extend(disp16) GR[reg2]sign-extend(Load-memory(adr, Halfword))	1	1	2						
LD.W	disp16[reg1], reg2	r r r r r 111001RRRRR d d d d d d d d d d d d d d d 1 <b>Note 4</b>	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←Load-memory(adr, Word)	1	1	2						

- Notes**
1. d d d d d d d d is the higher 8 bits of disp9.
  2. Only the lower half-word is valid.
  3. d d d d d d d d d d d d d d d d d d d d d d is the higher 21 bits of disp22.
  4. d d d d d d d d d d d d d d d is the higher 15 bits of disp16.

Instruction Set (alphabetical order) (2/4)

Mnemonic	Operand	Code	Operation	Execution Clock			Flag					
				i	r	l	CY	OV	S	Z	SAT	
LDSR	reg2, regID	r r r r r 1 1 1 1 1 1 R R R R R	SR[regID]←GR[reg2]	1	1	3						
		0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0	regID = EIPC, FEPC									
		<b>Note 1</b>	regID = EIPSW, FEPSW									
			regID = PSW				1	x	x	x	x	x
MOV	reg1, reg2	r r r r r 0 0 0 0 0 0 R R R R R	GR[reg2]←GR[reg1]	1	1	1						
	imm5, reg2	r r r r r 0 1 0 0 0 0 i i i i i	GR[reg2]←sign-extend(imm5)	1	1	1						
MOVEA	imm16, reg1, reg2	r r r r r 1 1 0 0 0 1 R R R R R i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1]+sign-extend(imm16)	1	1	1						
MOVHI	imm16, reg1, reg2	r r r r r 1 1 0 0 1 0 R R R R R i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1]+(imm16    0 <sup>16</sup> )	1	1	1						
MULH	reg1, reg2	r r r r r 0 0 0 1 1 1 R R R R R	GR[reg2]←GR[reg2] <sup>Note 2</sup> xGR[reg1] <sup>Note 2</sup> (Signed multiplication)	1	1	2						
	imm5, reg2	r r r r r 0 1 0 1 1 1 i i i i i	GR[reg2]←GR[reg2] <sup>Note 2</sup> xsign-extend(imm5) (Signed multiplication)	1	1	2						
MULHI	imm16, reg1, reg2	r r r r r 1 1 0 1 1 1 R R R R R i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1] <sup>Note 2</sup> ximm16 (Signed multiplication)	1	1	2						
NOP		0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Uses 1 clock cycle without doing anything	1	1	1						
NOT	reg1, reg2	r r r r r 0 0 0 0 1 R R R R R	GR[reg2]←NOT(GR[reg1])	1	1	1		0	x	x		
NOT1	bit#3, disp16[reg1]	0 1 b b b 1 1 1 1 1 0 R R R R R d d d d d d d d d d d d d d d d	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr, bit#3)) Store-memory-bit(adr, bit#3, Z flag)	4	4	4					x	
OR	reg1, reg2	r r r r r 0 0 1 0 0 0 R R R R R	GR[reg2]←GR[reg2]OR GR[reg1]	1	1	1		0	x	x		
ORI	imm16, reg1, reg2	r r r r r 1 1 0 1 0 0 R R R R R i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1]OR zero-extend(imm16)	1	1	1		0	x	x		
RETI		0 0 0 0 0 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0	if PSW.EP=1 then PC ←EIPC PSW ←EIPSW else if PSW.NP=1 then PC ←FEPC PSW ←FEPSW else PC ←EIPC PSW ←EIPSW	4	4	4	R	R	R	R	R	
SAR	reg1, reg2	r r r r r 1 1 1 1 1 1 R R R R R 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0	GR[reg2]←GR[reg2]arithmetically shift right by GR[reg1]	1	1	1	x	0	x	x		
	imm5, reg2	r r r r r 0 1 0 1 0 1 i i i i i	GR[reg2]←GR[reg2]arithmetically shift right by zero-extend(imm5)	1	1	1	x	0	x	x		

**Notes 1.** The op code of this instruction uses the field of reg1 though the source register is shown as reg2 in the above table. Therefore, the meaning of register specification for mnemonic description and op code is different from that of the other instructions.

rrrrr = regID specification  
RRRRR = reg2 specification

**2.** Only the lower half-word data is valid.

Instruction Set (alphabetical order) (3/4)

Mnemonic	Operand	Code	Operation	Execution Clock			Flag				
				i	r	l	CY	OV	S	Z	SAT
SATADD	reg1, reg2	r r r r r 000110RRRRR	GR[reg2]←saturated(GR[reg2]+GR[reg1])	1	1	1	x	x	x	x	x
	imm5, reg2	r r r r r 010001 i i i i i	GR[reg2]←saturated(GR[reg2]+sign-extend(imm5))	1	1	1	x	x	x	x	x
SATSUB	reg1, reg2	r r r r r 000101RRRRR	GR[reg2]←saturated(GR[reg2]-GR[reg1])	1	1	1	x	x	x	x	x
SATSUBI	imm16, reg1, reg2	r r r r r 110011RRRRR i i i i i i i i i i i i i i i i	GR[reg2]←saturated(GR[reg1]-sign-extend(imm16))	1	1	1	x	x	x	x	x
SATSUBR	reg1, reg2	r r r r r 000100RRRRR	GR[reg2]←saturated(GR[reg1]-GR[reg2])	1	1	1	x	x	x	x	x
SETF	cccc, reg2	r r r r r 1111110cccc 0000000000000000	if conditions are satisfied then GR[reg2]←0000001H else GR[reg2]←0000000H	1	1	1					
SET1	bit#3, disp16[reg1]	00bbb111110RRRRR dddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr, bit#3)) Store-memory-bit(adr, bit#3, 1)	4	4	4				x	
SHL	reg1, reg2	r r r r r 111111RRRRR 0000000011000000	GR[reg2]←GR[reg2] logically shift left by GR[reg1]	1	1	1	x	0	x	x	
	imm5, reg2	r r r r r 010110 i i i i i	GR[reg2]←GR[reg1] logically shift left by zero-extend(imm5)	1	1	1	x	0	x	x	
SHR	reg1, reg2	r r r r r 111111RRRRR 0000000010000000	GR[reg2]←GR[reg2] logically shift right by GR[reg1]	1	1	1	x	0	x	x	
	imm5, reg2	r r r r r 010100 i i i i i	GR[reg2]←GR[reg2] logically shift right by zero-extend(imm5)	1	1	1	x	0	x	x	
SLD.B	disp7[ep], reg2	r r r r r 0110dddddd	adr←ep+zero-extend(disp7) GR[reg2]←sign-extend(Load-memory(adr, Byte))	1	1	2					
SLD.H	disp8[ep], reg2	r r r r r 1000dddddd <b>Note 1</b>	adr←ep+zero-extend(disp8) GR[reg2]←sign-extend(Load-memory(adr, Halfword))	1	1	2					
SLD.W	disp8[ep], reg2	r r r r r 1010dddddd0 <b>Note 2</b>	adr←ep+zero-extend(disp8) GR[reg2]←Load-memory(adr, Word)	1	1	2					
SST.B	reg2, disp7[ep]	r r r r r 0111dddddd	adr←ep+zero-extend(disp7) Store-memory(adr, GR[reg2], Byte)	1	1	1					
SST.H	reg2, disp8[ep]	r r r r r 1001dddddd <b>Note 1</b>	adr←ep+zero-extend(disp8) Store-memory(adr, GR[reg2], Halfword)	1	1	1					
SST.W	reg2, disp8[ep]	r r r r r 1010dddddd1 <b>Note 2</b>	adr←ep+zero-extend(disp8) Store-memory(adr, GR[reg2], Word)	1	1	1					
ST.B	reg2, disp16[reg1]	r r r r r 111010RRRRR dddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Store-memory(adr, GR[reg2], Byte)	1	1	1					

- Notes**
1. ddddddd is the higher 7 bits of disp8.
  2. ddddddd is the higher 6 bits of disp8.

Instruction Set (alphabetical order) (4/4)

Mnemonic	Operand	Code	Operation	Execution Clock			Flag					
				i	r	l	CY	OV	S	Z	SAT	
ST.H	reg2, disp16[reg1]	r r r r r 1 1 1 0 1 1 R R R R R d d d d d d d d d d d d d d 0 <b>Note</b>	adr←GR[reg1]+sign-extend(disp16) Store-memory(adr, GR[reg2], Halfword)	1	1	1						
ST.W	reg2, disp16[reg1]	r r r r r 1 1 1 0 1 1 R R R R R d d d d d d d d d d d d d d 1 <b>Note</b>	adr←GR[reg1]+sign-extend(disp16) Store-memory(adr, GR[reg2], Word)	1	1	1						
STSR	regID, reg2	r r r r r 1 1 1 1 1 1 R R R R R 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0	GR[reg2]←SR[regID]	1	1	1						
SUB	reg1, reg2	r r r r r 0 0 1 1 0 1 R R R R R	GR[reg2]←GR[reg2]-GR[reg1]	1	1	1	x	x	x	x		
SUBR	reg1, reg2	r r r r r 0 0 1 1 0 0 R R R R R	GR[reg2]←GR[reg1]-GR[reg2]	1	1	1	x	x	x	x		
TRAP	vector	0 0 0 0 0 1 1 1 1 1 1 i i i i i 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0	EIPC ←PC+4(Restored PC) EIPSW ←PSW ECR.EICC ←Interrupt code PSW.EP ←1 PSW.ID ←1 PC ←00000040H(vector = 00H to 0FH) 00000050H(vector = 10H to 1FH)	4	4	4						
TST	reg1, reg2	r r r r r 0 0 1 0 1 1 R R R R R	result←GR[reg2] AND GR[reg1]	1	1	1		0	x	x		
TST1	bit#3, disp16[reg1]	1 1 b b b 1 1 1 1 1 0 R R R R R d d d d d d d d d d d d d d d	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr, bit#3))	3	3	3					x	
XOR	reg1, reg2	r r r r r 0 0 1 0 0 1 R R R R R	GR[reg2]←GR[reg2] XOR GR[reg1]	1	1	1		0	x	x		
XORI	imm16, reg1, reg2	r r r r r 1 1 0 1 0 1 R R R R R i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1] XOR zero-extend(imm16)	1	1	1		0	x	x		

**Note** ddddddddddddddd is the higher 15 bits of disp16.

## APPENDIX C INDEX

<b>[0]</b>	<ul style="list-style-type: none"> <li>1-bit output port ..... 154</li> <li>144-pin plastic LQFP ..... 25</li> </ul>		<ul style="list-style-type: none"> <li>baud rate generator</li> <li>compare register 0 to 3 ..... 291</li> <li>baud rate generator</li> <li>prescaler mode register 0 to 3 ..... 292</li> <li>baud rate generators 0 to 3 set-up values ..... 289</li> </ul>
<b>[A]</b>	<ul style="list-style-type: none"> <li>A/D conversion result register ..... 299</li> <li>A/D converter ..... 295</li> <li>A/D converter mode register 0 ..... 297</li> <li>A/D converter mode register 1 ..... 299</li> <li>A/D trigger mode ..... 302, 308</li> <li>A16 to A23 ..... 40</li> <li>ACKD ..... 240</li> <li>ACKE ..... 238</li> <li>AD0 to AD7 ..... 39</li> <li>AD8 to AD15 ..... 40</li> <li>ADCR0 to ADCR7 ..... 299</li> <li>address space ..... 56, 69</li> <li>ADIC0 ..... 116</li> <li>ADIF0 ..... 116</li> <li>ADM0 ..... 297</li> <li>ADM1 ..... 299</li> <li>ADMK0 ..... 116</li> <li>ADPR00 to ADPR02 ..... 116</li> <li>ADTRG ..... 38</li> <li>ALD ..... 239</li> <li>ALVn (n = 00, 01, 20 to 24) ..... 172</li> <li>ANI0 to ANI15 ..... 40</li> <li>ANIS0 to ANIS2 ..... 297</li> <li>application fields ..... 25</li> <li>arbitration ..... 239</li> <li>ASIM0, ASIM1 ..... 209</li> <li>ASIS ..... 212</li> <li>assembler reservation register ..... 51</li> <li>ASTB ..... 41</li> <li>asynchronous serial interface ..... 206</li> <li>asynchronous serial interface</li> <li>mode register 0, 1 ..... 209</li> <li>asynchronous serial interface</li> <li>status register ..... 212</li> <li>AV<sub>DD</sub> ..... 45</li> <li>AV<sub>REF</sub> ..... 46</li> <li>AV<sub>SS</sub> ..... 45</li> </ul>	<ul style="list-style-type: none"> <li>BCC ..... 88</li> <li>BCn1 (n = 0 to 7) ..... 88</li> <li>BCU ..... 29</li> <li>BIC ..... 82</li> <li>block diagram of ports ..... 342</li> <li>BPRM0 to BPRM3 ..... 292</li> <li>BPRn0 to BPRn3 (n = 0 to 3) ..... 292</li> <li>BRCE0 to BRCE3 ..... 292</li> <li>BRGC0 to BRGC3 ..... 291</li> <li>BRGn0 to BRGn7 (n = 0 to 3) ..... 291</li> <li>BS ..... 297</li> <li>buffer register ..... 322</li> <li>bus access ..... 83</li> <li>bus control function ..... 81</li> <li>bus control pin ..... 82</li> <li>bus control unit ..... 29</li> <li>bus cycle control register ..... 88</li> <li>bus hold ..... 89, 96</li> <li>bus priority ..... 97</li> <li>bus timing ..... 90</li> <li>bus width ..... 84</li> <li>byte access ..... 84</li> </ul>	
<b>[B]</b>	<ul style="list-style-type: none"> <li>baud rate generator 0 to 3 ..... 286</li> </ul>	<b>[C]</b>	<ul style="list-style-type: none"> <li>capture operation (timer 0) ..... 179</li> <li>capture operation (timer 1) ..... 186</li> <li>capture operation (timer 3) ..... 193</li> <li>capture register 10 to 13 ..... 162</li> <li>capture register 3 ..... 165</li> <li>capture/compare register 00 to 03 ..... 161</li> <li>capture/compare register 3 ..... 165</li> <li>CC00 to CC03, CC00L to CC03L ..... 161</li> <li>CC0IC0 to CC0IC3 ..... 116</li> <li>CC0IF0 to CC0IF3 ..... 116</li> <li>CC0MK0 to CC0MK3 ..... 116</li> <li>CC0PRn0 to CC0PRn2 (n = 0 to 3) ..... 116</li> <li>CC3 ..... 165</li> <li>CC3IC0 ..... 116</li> <li>CC3IF0 ..... 116</li> <li>CC3MK0 ..... 116</li> </ul>

CC3PR00 to CC3PR02 .....	116	communication command .....	393
CCLR0 .....	168	communication reservation .....	273
CE .....	297	compare operation (timer 0) .....	181
CE0 .....	166	compare operation (timer 1) .....	187
CE1 .....	169	compare operation (timer 2) .....	189
CE20 to CE24 .....	170	compare operation (timer 3) .....	194
CE3 .....	171	compare register 10, 11 .....	163
CESEL .....	142	compare register 20 to 24 .....	164
CG .....	29	conflict of signals .....	388
CKC .....	137	count clock selection (timer 0) .....	175
CKDIV0 to CKDIV1 .....	137	count clock selection (timer 1) .....	183
CKSEL .....	44	count clock selection (timer 2) .....	188
CL .....	210	count clock selection (timer 3) .....	192
CL0, CL1 .....	241	count operation (timer 0) .....	174
CLD .....	241	count operation (timer 1) .....	183
CLE .....	154	count operation (timer 2) .....	188
clear/start of timer (timer 0) .....	177	count operation (timer 3) .....	192
clear/start of timer (timer 1) .....	185	CP10 to CP13, CP10L to CP13L .....	162
clear/start of timer (timer 2) .....	189	CP3 .....	165
clear/start of timer (timer 3) .....	193	CPU .....	29
CLKOUT .....	44	CPU address space .....	56, 58
CLKOUT signal output control .....	152	CPU function .....	49
CLO .....	43	CPU register set .....	50
CLO signal output control .....	153	CRXE0 to CRXE3 .....	222
clock control register .....	137	CS .....	297
clock generation function .....	135	CS1 .....	169
clock generator .....	29	CS20 to CS24 .....	170
clock output control .....	165	CS3 .....	171
clock output inhibit .....	140, 152	CSI0 to CSI3 .....	220
clock output mode register .....	154	CSIC0 to CSIC3 .....	116
clock serial interface 0 to 3 .....	220	CSIF0 to CSIF3 .....	116
clock serial interface mode register 0 to 3 .....	222	CSIM0 to CSIM3 .....	222
CLOM .....	154	CSMK0 to CSMK3 .....	116
CLS <sub>n</sub> 0, CLS <sub>n</sub> 1 (n = 0 to 3) .....	223	CSOT0 to CSOT3 .....	222
CM10, CM11, CM10L, CM11L .....	163	CSPR <sub>n</sub> 0 to CSPR <sub>n</sub> 2 (n = 0 to 3) .....	116
CM1IC0, CM1IC1 .....	116	CTXE0 to CTXE3 .....	222
CM1IF0, CM1IF1 .....	116	CV <sub>DD</sub> .....	45
CM1MK0, CM1MK1 .....	116	CV <sub>SS</sub> .....	45
CM1PR <sub>n</sub> 0 to CM1PR <sub>n</sub> 2 (n = 0, 1) .....	116	CY .....	53
CM20 to CM24 .....	164		
CM2IC0 to CM2IC4 .....	116	<b>[D]</b>	
CM2IF0 to CM2IF4 .....	116	DAD .....	241
CM2MK0 to CM2MK4 .....	116	data space .....	58, 69, 97
CM2PR <sub>n</sub> 0 to CM2PR <sub>n</sub> 2 (n = 0 to 4) .....	116	data wait control register .....	86
CMS00 to CMS03 .....	167	DCLK0, DCLK1 .....	142
CMS3 .....	171	DFC .....	241
COI .....	239	diagram of processing status .....	141
command register .....	78	direct mode .....	136

$\overline{\text{DSTB}}$ .....	41	flash memory programming mode .....	54, 392
DWC .....	86	FR0 to FR2 .....	299
DWn0, DWn1 (n = 0 to 7) .....	86	frequency divider .....	124
		FS0 to FS2 .....	154
		function block configuration .....	28
<b>[E]</b>		<b>[G]</b>	
EBS .....	211	general register .....	51
ECLR0 .....	168	global pointer .....	51
ECR .....	52		
edge detection function .....	120	<b>[H]</b>	
EDV0 to EDV2 .....	124	halfword access .....	84
EDVC0 to EDVC2 .....	125	HALT mode .....	140, 143
EICC .....	52	$\overline{\text{HLDK}}$ .....	42
EIPC .....	52	$\overline{\text{HLDRQ}}$ .....	42
EIPSW .....	52		
element pointer .....	51	<b>[I]</b>	
ENTOn (n = 00, 01, 20 to 24) .....	172	I/O circuit of pins .....	47
EP .....	53, 128	I <sup>2</sup> C bus .....	231
ES300, ES301 .....	123	I <sup>2</sup> C interrupt .....	250
ESE .....	125	ID .....	53, 118
ESN0 .....	106	IDLE .....	142
ESn0, ESn1 (n = 00 to 05, 10 to 14, 20 to 24, AD, 50 to 53) .....	121	IDLE mode .....	140, 145
event divide control register 0 to 2 .....	125	idle state insertion function .....	100
event divide counter .....	124	IIC .....	242
event selection register .....	125	IIC clock selection register .....	241
EVS .....	125	IIC control register .....	237
example of CSI system configuration .....	230	IIC shift register .....	241
example of inserting wait states .....	87	IIC status register .....	239
EXC .....	239	IICC .....	237
exception processing function .....	99	IICCL .....	241
exception status flag .....	128	IICE .....	237
exception table .....	61	IICS .....	239
exception trap .....	129	IIC0 .....	116
extension code .....	271	IIF0 .....	116
external count clock .....	175, 184, 189	IIMK0 .....	116
external expansion mode .....	67	IIPR00 to IIPR02 .....	116
external interrupt mode register 1 to 6 .....	121	ILGOP .....	100
external interrupt mode register 7 .....	123	illegal op code .....	129
external memory area .....	65	IMS00 to IMS03 .....	167
external wait function .....	87	IMS04, IMS05 .....	168
external trigger mode .....	302, 315	IMS1 .....	169
		IMS20 to IMS24 .....	170
<b>[F]</b>		in-service priority register .....	118
FE .....	212	initial value after reset of each register .....	380
FECC .....	52	initialize .....	379
FEPC .....	52	input clock selection (clock generator) .....	136
FEPSW .....	52	INTAD .....	101
flash memory .....	383		

INTC .....	29	<b>[M]</b>	
INTCM10, INTCM11 .....	100	maskable interrupt .....	107
INTCM20 to INTCM24 .....	100	maskable interrupt status flag .....	118
INTCP10 to INTCP13 .....	100	measurement of cycle .....	201
INTCSI0 to INTCSI3 .....	101	measurement of pulse width .....	196
internal block diagram .....	28	memory block function .....	85
internal count clock .....	175, 183, 188	memory boundary operation condition .....	97
internal peripheral I/O interface .....	98	memory expansion mode register .....	68
internal RAM area .....	62	memory map .....	59
Internal ROM area .....	60	memory read .....	90
internal unit .....	29	memory write .....	94
interrupt control register .....	116	MM .....	68
interrupt controller .....	29	MM0 to MM3 .....	68
interrupt list .....	100	MOD0 to MOD3 .....	222
interrupt processing function .....	99	MODE0 to MODE2 .....	44
interrupt request .....	215	modulo H register .....	329
interrupt response time .....	133	modulo L register .....	329
interrupt source register .....	52	MS .....	297
interrupt stack pointer .....	51	MSTS .....	239
interrupt table .....	61	multiple interrupt .....	131
interval timer .....	195		
INTIIC .....	250	<b>[N]</b>	
INTM0 .....	116	NMI .....	38, 100
INTM1 to INTM6 .....	121	NMI pin edge detection function .....	106
INTM7 .....	123	NMI pin noise elimination .....	106
INTOV0, INTOV1 .....	100	noise elimination .....	119
INTP00 to INTP05 .....	37	normal operation mode .....	54
INTP0n/IINTCC0n (n = 0 to 3) .....	100	non-maskable interrupt .....	102
INTP10 to INTP14 .....	37	non-maskable interrupt status flag .....	106
INTP20 .....	37	NOT .....	211
INTP21 to INTP24 .....	43	NP .....	53, 106
INTP2n/INTCM2n (n = 0 to 4) .....	100	number of access clock .....	83
INTP30 .....	38		
INTP30/INTCC3 .....	101	<b>[O]</b>	
INTP50 to INTP53 .....	38, 101	off-board programming .....	383
INTSER .....	215	on-board programming .....	383
INTSR .....	215	operation in power save mode .....	89
INTST .....	215	operation in the stand-by mode .....	155
ISPR .....	118	operation mode .....	54
ISPR0 to ISPR7 .....	118	ordering information .....	25
		OST0 .....	166
<b>[L]</b>		OST1 .....	169
$\overline{\text{LBEN}}$ .....	41	output latch .....	322
link pointer .....	51	OV .....	53
LREL .....	237	OVE .....	212
LV .....	154	overflow (timer 0) .....	176
		overflow (timer 1) .....	184
		overflow (timer 2) .....	189

overflow (timer 3) ..... 192  
 OVF<sub>n</sub> (n = 0, 1, 20 to 24, 3) ..... 173  
 OVIC0, OVIC1 ..... 116  
 OVIF0, OVIF1 ..... 116  
 OVMK0, OVMK1 ..... 116  
 OVPR<sub>n</sub>0 to OVPR<sub>n</sub>2 (n = 0, 1) ..... 116

**[P]**

P0 ..... 348  
 P00 to P07 ..... 37, 348  
 P1 ..... 350  
 P10 ..... 366  
 P10 to P17 ..... 37, 350  
 P100 to P103 ..... 42, 356  
 P11 ..... 368  
 P110 to P117 ..... 42, 368  
 P12 ..... 371  
 P120 to P127 ..... 43, 371  
 P13 ..... 373  
 P130 to P137 ..... 43, 373  
 P14 ..... 375  
 P140 to P147 ..... 44, 375  
 P1IC0 to P1IC3 ..... 116  
 P1IF0 to P1IF3 ..... 116  
 P1MK0 to P1MK3 ..... 116  
 P1PR<sub>n</sub>0 to P1PR<sub>n</sub>2 (n = 0 to 3) ..... 116  
 P2 ..... 352  
 P20 to P26 ..... 38, 352  
 P3 ..... 354  
 P30 to P36 ..... 38, 354  
 P4 ..... 357  
 P40 to P47 ..... 39, 357  
 P5 ..... 359  
 P50 to P57 ..... 39, 359  
 P5IC0 to P5IC3 ..... 116  
 P5IF0 to P5IF3 ..... 116  
 P5MK0 to P5MK3 ..... 116  
 P5PR<sub>n</sub>0 to P5PR<sub>n</sub>2 (n = 0 to 3) ..... 116  
 P6 ..... 361  
 P60 to P67 ..... 40, 361  
 P7 ..... 363  
 P70 to P77 ..... 40, 363  
 P8 ..... 368  
 P80 to P87 ..... 40, 368  
 P9 ..... 364  
 P90 to P96 ..... 41, 364  
 PALV0 to PALV3 ..... 327  
 PB ..... 322

PC ..... 51  
 PE ..... 212  
 period in which interrupt is not acknowledged .... 133  
 peripheral I/O area ..... 63  
 peripheral I/O register ..... 71  
 pin configuration ..... 26  
 pin function ..... 31, 37  
 pin status ..... 36  
 PLL lock up ..... 139  
 PLL mode ..... 136  
 PLLSEL ..... 44  
 PM0 ..... 349  
 PM00 to PM07 ..... 349  
 PM1 ..... 351  
 PM10 (register) ..... 367  
 PM10 to PM17 (bit) ..... 351  
 PM100 to PM103 ..... 367  
 PM11 ..... 369  
 PM110 to PM117 ..... 369  
 PM12 ..... 371  
 PM120 to PM127 ..... 371  
 PM13 ..... 373  
 PM130 to PM137 ..... 373  
 PM14 ..... 375  
 PM140 to PM147 ..... 375  
 PM2 ..... 353  
 PM21 to PM26 ..... 353  
 PM3 ..... 355  
 PM30 to PM36 ..... 355  
 PM4 ..... 357  
 PM40 to PM47 ..... 357  
 PM5 ..... 360  
 PM50 to PM57 ..... 360  
 PM6 ..... 362  
 PM60 to PM67 ..... 362  
 PM9 ..... 365  
 PM90 to PM96 ..... 365  
 PMC0 ..... 349  
 PMC00 to PMC07 ..... 349  
 PMC1 ..... 351  
 PMC10 ..... 367  
 PMC10 to PMC16 ..... 351  
 PMC100 to PMC103 ..... 367  
 PMC11 ..... 370  
 PMC110 to PMC117 ..... 370  
 PMC12 ..... 372  
 PMC120 to PMC125, PMC127 ..... 372  
 PMC13 ..... 374

PMC130 to PMC137.....	374	PRCMD .....	78
PMC2 .....	353	PRERR .....	79, 139
PMC21 to PMC26.....	353	PRM00 to PRM03.....	166
PMC3 .....	356	PRM10 to PRM13.....	169
PMC30 to PMC35.....	356	PRM2n0 to PRM2n4 (n = 0 to 4).....	170
PMPn0 to PMPn2 (n = 0 to 3) .....	327	PRM30 to PRM33.....	171
port function .....	337	program counter .....	51
ports		program register set .....	51
Port 0.....	348	program space .....	58, 69, 97
Port 1.....	350	program status word.....	53
Port 2.....	352	programmable wait function .....	86
Port 3.....	354	PS .....	297
Port 4.....	357	PS0, PS1 .....	210
Port 5.....	359	PSC .....	142
Port 6.....	361	PSW .....	53, 106, 118, 128
Port 7.....	363	PWM .....	30
Port 8.....	363	PWM control register 0 to 3 .....	327
Port 9.....	364	PWM modulo register 0 to 3 .....	329
Port 10 .....	366	PWM output .....	198
Port 11 .....	368	PWM prescaler register 0 to 3 .....	328
Port 12 .....	371	PWM unit .....	325
Port 13 .....	373	PWM0 to PWM3 .....	42, 328
Port 14 .....	375	PWMC0 to PWMC3 .....	327
Port mode control register		PWME0 to PWME3 .....	327
Port 0 mode control register .....	349	PWPn0, PWPn1 (n = 0 to 3).....	328
Port 1 mode control register .....	351	PWPR0 to PWPR3 .....	328
Port 2 mode control register .....	353		
Port 3 mode control register .....	356	<b>[R]</b>	
Port 10 mode control register .....	367	R/W .....	41
Port 11 mode control register .....	370	r0 to r31 .....	51
Port 12 mode control register .....	372	RAM .....	29
Port 13 mode control register .....	374	$\overline{RD}$ .....	42
Port mode register		real-time output function .....	321
Port 0 mode register.....	349	real-time pulse unit .....	29, 157
Port 1 mode register.....	351	receive buffer .....	213
Port 2 mode register.....	353	reception completion interrupt .....	215
Port 3 mode register.....	355	reception error interrupt .....	215
Port 4 mode register.....	358	recommended connection of unused pins .....	48
Port 5 mode register.....	360	REG0 to REG7 .....	78
Port 6 mode register.....	362	repetition frequency .....	335
Port 9 mode register.....	365	$\overline{RESET}$ .....	100
Port 10 mode register .....	367	$\overline{RESET}$ .....	45
Port 11 mode register .....	369	reset function .....	377
Port 12 mode register .....	371	ROM .....	29
Port 13 mode register .....	374	ROM-less mode .....	54
Port 14 mode register .....	375	RPU .....	29
power save control .....	140	RTP .....	30, 322
power save control register .....	142	RTP0 to RTP7 .....	43

RXB, RXBL .....	213	SRIF0 .....	116
RXB0 to RXB7 .....	213	SRMK0 .....	116
RXD .....	38	SRPR00 to SRPR02.....	116
RXE .....	209	stack pointer .....	51
RXEB.....	213	start condition .....	243
<b>[S]</b>		status saving register for interrupt .....	54
S .....	53	status saving register for NMI .....	52
SAT .....	53	STD .....	240
$\overline{\text{SCK0}}$ .....	39	STIC0 .....	116
$\overline{\text{SCK1}}$ .....	39	STIF0.....	116
$\overline{\text{SCK2}}$ .....	43	STMK0 .....	116
$\overline{\text{SCK3}}$ .....	43	stop condition.....	247
SCL .....	38	STP .....	142
SCLS .....	211	STPR00 to STPR02 .....	116
SCS0, SCS1 .....	123	STT .....	238
SDA .....	38	SVA .....	242
securing oscillation stabilization time .....	149	SYC .....	82
SEIC0 .....	116	SYN0 to SYN3 .....	327
SEIF0 .....	116	SYS .....	79, 139
SEMK0 .....	116	system control register .....	82
SEPR00 to SEPR02 .....	116	system register set .....	52
serial I/O shift register 0 to 3 .....	223	system status register .....	79
serial interface .....	29, 205	<b>[T]</b>	
SI0 .....	39	TBC .....	150
SI1 .....	39	TBCS .....	142
SI2 .....	43	TCLR0 .....	37
SI3 .....	43	text pointer .....	51
single-chip mode .....	54	TI0 .....	37
SIO .....	29	TI1 .....	37
SIO0 to SIO3 .....	213	TI20 .....	37
SIO <sub>n</sub> 0 to SIO <sub>n</sub> 7 (n = 0 to 3) .....	213	TI21 to TI24 .....	43
SL .....	210	time base counter .....	150
slave address register .....	242	timer 0 operation .....	174
SMC .....	24	timer 0, 0L .....	161
SO0 .....	39	timer 1 operation .....	183
SO1 .....	39	timer 1, 1L .....	162
SO2 .....	43	timer 2 .....	164
SO3 .....	43	timer 2 operation .....	188
software exception .....	126	timer 20 to 24 .....	164
software STOP mode .....	140, 147	timer 3 .....	165
SOT .....	212	timer 3 operation .....	192
SPD .....	240	timer control register 00 .....	166
specific register .....	77	timer control register 01 .....	167
specification of transfer direction .....	245	timer control register 02 .....	168
SPIE .....	237	timer control register 1 .....	169
SPT .....	238	timer control register 20 to 24 .....	170
SRIC0 .....	116	timer control register 3 .....	171

timer output control register 0, 1 ..... 172  
timer overflow status register ..... 173  
timer trigger mode ..... 302, 311  
timer/counter function ..... 157  
timing of 3-wire serial I/O mode ..... 226, 227, 229  
TM0, TM0L ..... 161  
TM1, TM1L ..... 162  
TM20 to TM24 ..... 164  
TM3 ..... 165  
TMC00 ..... 166  
TMC01 ..... 167  
TMC02 ..... 168  
TMC1 ..... 169  
TMC20 to TMC24 ..... 170  
TMC3 ..... 171  
TO00, TO01 ..... 37  
TO20 ..... 37  
TO21 to TO24 ..... 43  
TOC0, TOC1 ..... 172  
toggle output ..... 191  
TOVS ..... 173  
transmission completion interrupt ..... 215  
transmission shift register ..... 214  
TRAP0n, TRAP1n (n = 0 to F) ..... 100  
TRC ..... 240  
TRG0, TRG1 ..... 299  
TXD ..... 38  
TXE ..... 209  
TXED ..... 214  
TXS, TXSL ..... 214  
TXS0 to TXS7 ..... 214

$\overline{\text{WRL}}$  ..... 42  
WTIM ..... 238  
Word access ..... 84  
Wrap-around ..... 58, 69

**[X]**

X1, X2 ..... 45

**[Z]**

Z ..... 53  
zero register ..... 51

**[U]**

UART ..... 206  
 $\overline{\text{UBEN}}$  ..... 41  
UNLOCK ..... 79, 139

**[V]**

V<sub>DD</sub> ..... 45  
V<sub>PP</sub> ..... 46  
V<sub>SS</sub> ..... 45

**[W]**

$\overline{\text{WAIT}}$  ..... 44  
Wake-up function ..... 273  
Wait function ..... 86  
WREL ..... 237  
 $\overline{\text{WRH}}$  ..... 42

## Facsimile Message

From:

Name

Company

Tel.

FAX

Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

*Thank you for your kind support.*

### North America

NEC Electronics Inc.  
Corporate Communications Dept.  
Fax: 1-800-729-9288  
1-408-588-6130

### Hong Kong, Philippines, Oceania

NEC Electronics Hong Kong Ltd.  
Fax: +852-2886-9022/9044

### Asian Nations except Philippines

NEC Electronics Singapore Pte. Ltd.  
Fax: +65-250-3583

### Europe

NEC Electronics (Europe) GmbH  
Technical Documentation Dept.  
Fax: +49-211-6503-274

### Korea

NEC Electronics Hong Kong Ltd.  
Seoul Branch  
Fax: 02-528-4411

### Japan

NEC Semiconductor Technical Hotline  
Fax: 044-548-7900

### South America

NEC do Brasil S.A.  
Fax: +55-11-6465-6829

### Taiwan

NEC Electronics Taiwan Ltd.  
Fax: 02-2719-5951

I would like to report the following error/make the following suggestion:

Document title: \_\_\_\_\_

Document number: \_\_\_\_\_ Page number: \_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Technical Accuracy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>