

To our customers,

---

## Old Company Name in Catalogs and Other Documents

---

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

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

April 1<sup>st</sup>, 2010  
Renesas Electronics Corporation

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

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

## Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
  - “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
  - “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
  - “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

## **V850E/MS2<sup>TM</sup>**

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

#### **Hardware**

---

#### **μPD703130**

[MEMO]

### ① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

### ② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

### ③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

**V850E/MS1, V850E/MS2, 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.**

**Green Hills Software is a trademark of Green Hills Software, Inc.**

- **The information contained in this document is being issued in advance of the production cycle for the device. The parameters for the device may change before final production or NEC Corporation, at its own discretion, may withdraw the device prior to its production.**
  - **Not all devices/types available in every country. Please check with local NEC representative for availability and additional information.**
  - No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.
  - NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.
  - Descriptions of circuits, software, and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software, and information in the design of the customer's equipment shall be done under the full responsibility of the customer. NEC Corporation assumes no responsibility for any losses incurred by the customer or third parties arising from the use of these circuits, software, and information.
  - While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.
  - NEC devices are classified into the following three quality grades:  
 "Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device before using it in a particular application.
    - Standard: Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots
    - Special: Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support)
    - Specific: Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.
- The quality grade of NEC devices is "Standard" unless otherwise specified in NEC's Data Sheets or Data Books. If customers intend to use NEC devices for applications other than those specified for Standard quality grade, they should contact an NEC sales representative in advance.

M5D 98.12

# Regional Information

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

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

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

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

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

**NEC Electronics (Germany) GmbH**

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

**NEC Electronics (UK) Ltd.**

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

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

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

**NEC Electronics (Germany) GmbH**

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

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

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

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

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

**NEC Electronics (Germany) GmbH**

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

**NEC Electronics Hong Kong Ltd.**

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

**NEC Electronics Hong Kong Ltd.**

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

**NEC Electronics Singapore Pte. Ltd.**

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

**NEC Electronics Taiwan Ltd.**

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

**NEC do Brasil S.A.**

Electron Devices Division  
Guarulhos-SP Brasil  
Tel: 55-11-6462-6810  
Fax: 55-11-6462-6829

J00.7

[MEMO]



## INTRODUCTION

<b>Target Readers</b>	This manual is intended for users who wish to understand the functions of the V850E/MS2 ( $\mu$ PD703130) to design application systems using the V850E/MS2.
<b>Purpose</b>	This manual is designed to help users understand the hardware functions of the V850E/MS2.
<b>Organization</b>	The <b>V850E/MS2 User's Manual</b> consists of two manuals: Hardware (this manual) and Architecture ( <b>V850E/MS1™ User's Manual Architecture</b> ). The organization of each manual is as follows:

Hardware	Architecture
<ul style="list-style-type: none"><li>• Pin functions</li><li>• CPU function</li><li>• Internal peripheral functions</li></ul>	<ul style="list-style-type: none"><li>• Data type</li><li>• Register set</li><li>• Instruction format and instruction set</li><li>• Interrupts and exceptions</li><li>• Pipeline operation</li></ul>

<b>How to Use This Manual</b>	<p>It is assumed that the readers of this manual have general knowledge of electrical engineering, logic circuits, and microcontrollers.</p> <ul style="list-style-type: none"><li>• To find the details of a register where the name is known → Refer to <b>APPENDIX A REGISTER INDEX</b>.</li><li>• To find the details of a function, etc. where the name is known → Refer to <b>APPENDIX C INDEX</b>.</li><li>• To understand the details of an instruction function → Refer to the <b>V850E/MS1 User's Manual Architecture</b>.</li><li>• To understand the overall functions of the V850E/MS2 → Read this manual in the order of the <b>CONTENTS</b>.</li></ul>
-------------------------------	---

## Conventions

Data significance:	Higher digits on the left and lower digits on the right
Active low representation:	$\overline{\text{xxx}}$ (overscore over pin or signal name)
Memory map address:	Higher address on the top and lower address on the bottom
<b>Note:</b>	Footnote for item marked with <b>Note</b> in the text
<b>Caution:</b>	Information requiring particular attention
<b>Remark:</b>	Supplementary information
Numerical representation:	Binary ... xxxx or xxxxB
	Decimal ... xxxx
	Hexadecimal ... xxxxH
Prefix indicating the power of 2 (address space, memory capacity):	K (kilo) ... $2^{10} = 1,024$
	M (mega) ... $2^{20} = 1,024^2$
	G (giga) ... $2^{30} = 1,024^3$
	Word ... 32 bits
Data type:	Halfword ... 16 bits
	Byte ... 8 bits

## Related Documents

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

### Document related to device

Document Name	Document No.
V850E/MS2 User's Manual Hardware	This manual
V850E/MS1 User's Manual Architecture	U12197E

## CONTENTS

<b>CHAPTER 1 INTRODUCTION</b>	<b>19</b>
1.1 Outline	19
1.2 Features	20
1.3 Applications	22
1.4 Ordering Information	22
1.5 Pin Configuration (Top View)	23
1.6 Function Block	25
1.6.1 Internal block diagram	25
1.6.2 Internal units	26
<b>CHAPTER 2 PIN FUNCTIONS</b>	<b>29</b>
2.1 List of Pin Functions	29
2.2 Pin Status	33
2.3 Description of Pin Functions	35
2.4 Pin I/O Circuits and Recommended Connection of Unused Pins	46
2.5 Pin I/O Circuits	48
<b>CHAPTER 3 CPU FUNCTION</b>	<b>49</b>
3.1 Features	49
3.2 CPU Register Set	50
3.2.1 Program register set	51
3.2.2 System register set	52
3.3 Operation Modes	54
3.3.1 Operation modes	54
3.3.2 Operation mode specification	54
3.4 Address Space	55
3.4.1 CPU address space	55
3.4.2 Image	56
3.4.3 Wrap-around of CPU address space	57
3.4.4 Memory map	58
3.4.5 Area	59
3.4.6 External expansion mode	63
3.4.7 Recommended use of address space	65
3.4.8 Peripheral I/O registers	67
3.4.9 Specific registers	73
<b>CHAPTER 4 BUS CONTROL FUNCTION</b>	<b>77</b>
4.1 Features	77
4.2 Bus Control Pins	77
4.3 Memory Block Function	78
4.4 Bus Cycle Type Control Function	79
4.4.1 Bus cycle type configuration register (BCT)	79
4.5 Bus Access	81
4.5.1 Number of access clocks	81

4.5.2	Bus sizing function .....	82
4.5.3	Bus width .....	83
<b>4.6</b>	<b>Wait Function .....</b>	<b>87</b>
4.6.1	Programmable wait function .....	87
4.6.2	External wait function .....	88
4.6.3	Relationship between programmable wait and external wait .....	88
4.6.4	Bus cycles in which the wait function is valid .....	89
<b>4.7</b>	<b>Idle State Insertion Function .....</b>	<b>91</b>
<b>4.8</b>	<b>Bus Hold Function .....</b>	<b>93</b>
4.8.1	Outline of function .....	93
4.8.2	Bus hold procedure .....	94
4.8.3	Operation in power save mode .....	94
4.8.4	Bus hold timing .....	95
<b>4.9</b>	<b>Bus Priority Order .....</b>	<b>96</b>
<b>4.10</b>	<b>Boundary Operation Conditions .....</b>	<b>96</b>
4.10.1	Program space .....	96
4.10.2	Data space .....	97
<b>CHAPTER 5</b>	<b>MEMORY ACCESS CONTROL FUNCTION .....</b>	<b>99</b>
<b>5.1</b>	<b>SRAM, External ROM, External I/O Interface .....</b>	<b>99</b>
5.1.1	SRAM connections .....	99
5.1.2	SRAM, external ROM, external I/O access .....	100
<b>5.2</b>	<b>Page ROM Controller (ROMC) .....</b>	<b>104</b>
5.2.1	Features .....	104
5.2.2	Page ROM connections .....	104
5.2.3	On-page/off-page judgment .....	106
5.2.4	Page ROM configuration register (PRC) .....	108
5.2.5	Page ROM access .....	109
<b>5.3</b>	<b>DRAM Controller .....</b>	<b>110</b>
5.3.1	Features .....	110
5.3.2	DRAM connections .....	111
5.3.3	Address multiplex function .....	112
5.3.4	DRAM configuration registers 0 to 3 (DRC0 to DRC3) .....	113
5.3.5	DRAM type configuration register (DTC) .....	116
5.3.6	DRAM access .....	117
5.3.7	DRAM access during DMA flyby transfer .....	125
5.3.8	Refresh control function .....	127
5.3.9	Self-refresh functions .....	132
<b>CHAPTER 6</b>	<b>DMA FUNCTIONS (DMA CONTROLLER) .....</b>	<b>135</b>
<b>6.1</b>	<b>Features .....</b>	<b>135</b>
<b>6.2</b>	<b>Configuration .....</b>	<b>136</b>
<b>6.3</b>	<b>Control Registers .....</b>	<b>137</b>
6.3.1	DMA source address registers 0 to 3 (DSA0 to DSA3) .....	137
6.3.2	DMA destination address registers 0 to 3 (DDA0 to DDA3) .....	139
6.3.3	DMA byte count registers 0 to 3 (DBC0 to DBC3) .....	141
6.3.4	DMA addressing control registers 0 to 3 (DADC0 to DADC3) .....	142
6.3.5	DMA channel control registers 0 to 3 (DCHC0 to DCHC3) .....	144

6.3.6	DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)	145
6.3.7	DMA disable status register (DDIS)	147
6.3.8	DMA restart register (DRST)	147
6.3.9	Flyby transfer data wait control register (FDW)	148
<b>6.4</b>	<b>DMA Bus States</b>	<b>149</b>
6.4.1	Types of bus states	149
6.4.2	DMAC state transition	152
<b>6.5</b>	<b>Transfer Mode</b>	<b>153</b>
6.5.1	Single transfer mode	153
6.5.2	Single-step transfer mode	154
6.5.3	Block transfer mode	154
<b>6.6</b>	<b>Transfer Types</b>	<b>155</b>
6.6.1	Two-cycle transfer	155
6.6.2	Flyby transfer	159
<b>6.7</b>	<b>Transfer Objects</b>	<b>163</b>
6.7.1	Transfer type and transfer objects	163
6.7.2	External bus cycle during DMA transfer	163
<b>6.8</b>	<b>DMA Channel Priorities</b>	<b>164</b>
<b>6.9</b>	<b>Next Address Setting Function</b>	<b>164</b>
<b>6.10</b>	<b>DMA Transfer Start Factors</b>	<b>165</b>
<b>6.11</b>	<b>Interrupting DMA Transfer</b>	<b>166</b>
6.11.1	Interruption factors	166
6.11.2	Forcible interruption	166
<b>6.12</b>	<b>Terminating DMA Transfer</b>	<b>166</b>
6.12.1	DMA transfer end interrupt	166
6.12.2	Forcible termination	167
<b>6.13</b>	<b>Boundary of Memory Area</b>	<b>168</b>
<b>6.14</b>	<b>Transfer of Misalign Data</b>	<b>168</b>
<b>6.15</b>	<b>Clocks of DMA Transfer</b>	<b>168</b>
<b>6.16</b>	<b>Maximum Response Time to DMA Request</b>	<b>168</b>
<b>6.17</b>	<b>One Time Single Transfer with DMARQ0 to DMARQ3</b>	<b>170</b>
<b>6.18</b>	<b>Bus Arbitration for CPU</b>	<b>171</b>
<b>6.19</b>	<b>Precaution</b>	<b>171</b>
<b>CHAPTER 7</b>	<b>INTERRUPT/EXCEPTION PROCESSING FUNCTION</b>	<b>173</b>
<b>7.1</b>	<b>Features</b>	<b>173</b>
<b>7.2</b>	<b>Non-Maskable Interrupt</b>	<b>177</b>
7.2.1	Operation	178
7.2.2	Restore	180
7.2.3	Non-maskable interrupt status flag (NP)	181
7.2.4	Noise elimination	181
7.2.5	Edge detection function	181
<b>7.3</b>	<b>Maskable Interrupts</b>	<b>182</b>
7.3.1	Operation	182
7.3.2	Restore	184
7.3.3	Priorities of maskable interrupts	185
7.3.4	Interrupt control register (xxICn)	189
7.3.5	In-service priority register (ISPR)	191

7.3.6	Maskable interrupt status flag (ID)	191
7.3.7	Noise elimination	192
7.3.8	Edge detection function	193
<b>7.4</b>	<b>Software Exception</b>	<b>195</b>
7.4.1	Operation	195
7.4.2	Restore	196
7.4.3	Exception status flag (EP)	197
<b>7.5</b>	<b>Exception Trap</b>	<b>198</b>
7.5.1	Illegal op code definition	198
7.5.2	Operation	199
7.5.3	Restore	199
<b>7.6</b>	<b>Multiple Interrupt Servicing Control</b>	<b>200</b>
<b>7.7</b>	<b>Interrupt Latency Time</b>	<b>202</b>
<b>7.8</b>	<b>Periods in Which Interrupt Is Not Acknowledged</b>	<b>202</b>
<b>CHAPTER 8</b>	<b>CLOCK GENERATOR FUNCTIONS</b>	<b>203</b>
<b>8.1</b>	<b>Features</b>	<b>203</b>
<b>8.2</b>	<b>Configuration</b>	<b>203</b>
<b>8.3</b>	<b>Input Clock Selection</b>	<b>204</b>
8.3.1	Direct mode	204
8.3.2	PLL mode	204
8.3.3	Clock control register (CKC)	205
<b>8.4</b>	<b>PLL Lockup</b>	<b>206</b>
<b>8.5</b>	<b>Power Saving Control</b>	<b>207</b>
8.5.1	Outline	207
8.5.2	Control registers	209
8.5.3	HALT mode	210
8.5.4	IDLE mode	212
8.5.5	Software STOP mode	214
8.5.6	Clock output inhibit mode	215
<b>8.6</b>	<b>Securing Oscillation Stabilization Time</b>	<b>216</b>
8.6.1	Specifying securing of oscillation stabilization time	216
8.6.2	Time base counter (TBC)	218
<b>CHAPTER 9</b>	<b>TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)</b>	<b>219</b>
<b>9.1</b>	<b>Features</b>	<b>219</b>
<b>9.2</b>	<b>Basic Configuration</b>	<b>220</b>
9.2.1	Timer 1	223
9.2.2	Timer 4	225
<b>9.3</b>	<b>Control Registers</b>	<b>226</b>
<b>9.4</b>	<b>Timer 1 Operation</b>	<b>233</b>
9.4.1	Count operation	233
9.4.2	Count clock selection	234
9.4.3	Overflow	235
9.4.4	Clearing/starting timer by TCLR1n signal input	236
9.4.5	Capture operation	237
9.4.6	Compare operation	240
<b>9.5</b>	<b>Timer 4 Operation</b>	<b>242</b>

9.5.1	Count operation .....	242
9.5.2	Count clock selection.....	242
9.5.3	Overflow .....	242
9.5.4	Compare operation .....	243
<b>9.6</b>	<b>Application Example .....</b>	<b>245</b>
<b>9.7</b>	<b>Precaution .....</b>	<b>252</b>
<b>CHAPTER 10</b>	<b>SERIAL INTERFACE FUNCTION .....</b>	<b>255</b>
<b>10.1</b>	<b>Features.....</b>	<b>255</b>
<b>10.2</b>	<b>Asynchronous Serial Interfaces 0, 1 (UART0, UART1) .....</b>	<b>256</b>
10.2.1	Features .....	256
10.2.2	Configuration .....	257
10.2.3	Control registers .....	259
10.2.4	Interrupt request .....	266
10.2.5	Operation .....	267
<b>10.3</b>	<b>Clocked Serial Interfaces 0, 1 (CSI0, CSI1) .....</b>	<b>271</b>
10.3.1	Features .....	271
10.3.2	Configuration .....	271
10.3.3	Control registers .....	273
10.3.4	Basic operation.....	276
10.3.5	Transmission by CSI0, CSI1 .....	278
10.3.6	Reception by CSI0, CSI1 .....	279
10.3.7	Transmission and reception by CSI0, CSI1 .....	280
10.3.8	Example of system configuration.....	281
<b>10.4</b>	<b>Dedicated Baud Rate Generators 0, 1 (BRG0, BRG1).....</b>	<b>282</b>
10.4.1	Configuration and function.....	282
10.4.2	Baud rate generator compare registers 0, 1 (BRGC0, BRGC1).....	285
10.4.3	Baud rate generator prescaler mode registers 0, 1 (BPRM0, BPRM1) .....	286
<b>CHAPTER 11</b>	<b>A/D CONVERTER.....</b>	<b>287</b>
<b>11.1</b>	<b>Features.....</b>	<b>287</b>
<b>11.2</b>	<b>Configuration .....</b>	<b>287</b>
<b>11.3</b>	<b>Control Registers .....</b>	<b>290</b>
<b>11.4</b>	<b>A/D Converter Operation .....</b>	<b>295</b>
11.4.1	Basic operation of A/D converter .....	295
11.4.2	Operation mode and trigger mode.....	296
<b>11.5</b>	<b>Operation in A/D Trigger Mode .....</b>	<b>300</b>
11.5.1	Select mode operations .....	300
11.5.2	Scan mode operations.....	302
<b>11.6</b>	<b>Operation in Timer Trigger Mode.....</b>	<b>303</b>
11.6.1	Select mode operations .....	304
11.6.2	Scan mode operations.....	308
<b>11.7</b>	<b>Operating Precautions .....</b>	<b>310</b>
11.7.1	Stopping conversion operation .....	310
11.7.2	Timer trigger interval.....	310
11.7.3	Operation of standby mode .....	310
11.7.4	Compare match interrupt when in timer trigger mode .....	310

<b>CHAPTER 12 PORT FUNCTIONS.....</b>	<b>311</b>
<b>12.1 Features.....</b>	<b>311</b>
<b>12.2 Port Configuration.....</b>	<b>312</b>
<b>12.3 Port Pin Functions.....</b>	<b>328</b>
12.3.1 Port 0.....	328
12.3.2 Port 1.....	331
12.3.3 Port 2.....	334
12.3.4 Port 3.....	337
12.3.5 Port 4.....	339
12.3.6 Port 5.....	341
12.3.7 Port 6.....	343
12.3.8 Port 7.....	345
12.3.9 Port 8.....	346
12.3.10 Port 9.....	350
12.3.11 Port 10.....	353
12.3.12 Port A.....	355
12.3.13 Port B.....	357
12.3.14 Port X.....	359
<b>CHAPTER 13 RESET FUNCTIONS.....</b>	<b>361</b>
<b>13.1 Features.....</b>	<b>361</b>
<b>13.2 Pin Functions.....</b>	<b>361</b>
<b>13.3 Initialization.....</b>	<b>362</b>
<b>APPENDIX A CAUTIONS.....</b>	<b>365</b>
<b>A.1 Restriction on Repeated Execution of sld Instruction.....</b>	<b>365</b>
A.1.1 Details of malfunction.....	365
A.1.2 Countermeasures.....	367
<b>APPENDIX B REGISTER INDEX.....</b>	<b>369</b>
<b>APPENDIX C INSTRUCTION SET LIST.....</b>	<b>375</b>
<b>C.1 General Examples.....</b>	<b>375</b>
<b>C.2 Instruction Set (in Alphabetical Order).....</b>	<b>378</b>
<b>APPENDIX D INDEX.....</b>	<b>385</b>



## LIST OF FIGURES (1/3)

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.....	55
3-5	Image on Address Space.....	56
3-6	Recommended Memory Map.....	66
4-1	Example of Inserting Wait States .....	88
5-1	Example of Connection to SRAM.....	99
5-2	SRAM, External ROM, External I/O Access Timing.....	100
5-3	Example of Page ROM Connections .....	104
5-4	On-Page/Off-Page Judgment for Page ROM Connection .....	106
5-5	Page ROM Access Timing .....	109
5-6	Examples of Connections to DRAM .....	111
5-7	Row Address/Column Address Output .....	112
5-8	High-Speed Page DRAM Access Timing .....	117
5-9	EDO DRAM Access Timing.....	121
5-10	DRAM Access Timing During DMA Flyby Transfer .....	125
5-11	CBR Refresh Timing.....	131
5-12	CBR Self-Refresh Timing .....	133
6-1	DMAC Bus Cycle State Transition Diagram .....	152
6-2	Single Transfer Example 1 .....	153
6-3	Single Transfer Example 2 .....	153
6-4	Single-Step Transfer Example 1 .....	154
6-5	Single-Step Transfer Example 2 .....	154
6-6	Block Transfer Example .....	154
6-7	Timing of Two-Cycle Transfer.....	155
6-8	Timing of Flyby Transfer (DRAM → External I/O) .....	159
6-9	Timing of Flyby Transfer (Internal Peripheral I/O → Internal RAM).....	162
6-10	Buffer Register Configuration .....	164
6-11	Example of Forcible Termination of DMA Transfer.....	167
7-1	Block Diagram of Interrupt Control Function.....	176
7-2	Processing Configuration of Non-Maskable Interrupt .....	178
7-3	Acknowledging Non-Maskable Interrupt Request .....	179
7-4	RETI Instruction Processing .....	180
7-5	Maskable Interrupt Servicing .....	183
7-6	RETI Instruction Processing .....	184
7-7	Example of Servicing in Which Another Interrupt Request Is Issued While Interrupt Is Being Serviced.....	186

## LIST OF FIGURES (2/3)

Figure No.	Title	Page
7-8	Example of Servicing Interrupt Requests Simultaneously Generated .....	188
7-9	Example of Noise Elimination Timing.....	192
7-10	Software Exception Processing.....	195
7-11	RETI Instruction Processing .....	196
7-12	Exception Trap Processing.....	199
7-13	Pipeline Operation at Interrupt Request Acknowledgement (Outline) .....	202
8-1	Power Save Mode State Transition Diagram.....	208
9-1	Basic Operation of Timer 1 .....	233
9-2	Operation After Overflow (If ECLR1n = 0 and OSTn = 1) .....	235
9-3	Timer Clear/Start Operation by TCLR1n Signal Input (If ECLR1n = 1 and OSTn = 0) .....	236
9-4	Relationship Between Clear/Start by TCLR1n Signal Input and Overflow Operation (If ECLR1n = 1 and OSTn = 1) .....	237
9-5	Example of Capture Operation.....	238
9-6	Example of TM11 Capture Operation (When Both Edges Are Specified).....	239
9-7	Example of Compare Operation .....	240
9-8	Example of TM11 Compare Operation (Set/Reset Output Mode) .....	241
9-9	Basic Operation of Timer 4 .....	242
9-10	Example of TM40 Compare Operation.....	243
9-11	Example of Timing in Interval Timer Operation .....	245
9-12	Example of Interval Timer Operation Setting Procedure .....	245
9-13	Example of Pulse Measurement Timing.....	246
9-14	Example of Pulse Width Measurement Setting Procedure .....	247
9-15	Example of Interrupt Request Servicing Routine Which Calculates the Pulse Width .....	247
9-16	Example of PWM Output Timing .....	248
9-17	Example of PWM Output Setting Procedure .....	249
9-18	Example of Interrupt Request Servicing Routine for Rewriting Compare Value.....	249
9-19	Example of Frequency Measurement Timing.....	250
9-20	Example of Frequency Measurement Setting Procedure .....	251
9-21	Example of Interrupt Request Servicing Routine Which Calculates the Frequency .....	251
10-1	Block Diagram of Asynchronous Serial Interface .....	258
10-2	Transmission/Reception Data Format of Asynchronous Serial Interface .....	267
10-3	Asynchronous Serial Interface Transmission Completion Interrupt Timing .....	268
10-4	Asynchronous Serial Interface Reception Complete Interrupt Timing.....	270
10-5	Receive Error Timing .....	270
10-6	Block Diagram of Clocked Serial Interface .....	272
10-7	Timing of 3-Wire Serial I/O Mode (Transmission).....	278
10-8	Timing of 3-Wire Serial I/O Mode (Reception).....	279
10-9	Timing of 3-Wire Serial I/O Mode (Transmission/Reception) .....	281
10-10	Example of CSI System Configuration .....	281

## LIST OF FIGURES (3/3)

Figure No.	Title	Page
10-11	Block Diagram of Dedicated Baud Rate Generator .....	282
11-1	A/D Converter Block Diagram .....	289
11-2	Relationship Between Analog Input Voltage and A/D Conversion Results.....	294
11-3	Select Mode Operation Timing: 1-Buffer Mode (ANI1) .....	297
11-4	Select Mode Operation Timing: 4-Buffer Mode (ANI3) .....	298
11-5	Scan Mode Operation Timing: 4-Channel Scan (ANI0 to ANI3) .....	299
11-6	Example of 1-Buffer Mode (A/D Trigger Select 1-Buffer) Operation .....	300
11-7	Example of 4-Buffer Mode (A/D Trigger Select 4-Buffer) Operation .....	301
11-8	Example of Scan Mode (A/D Trigger Scan) Operation.....	302
11-9	Example of 1-Trigger Mode (Timer Trigger Select 1-Buffer 1-Trigger) Operation .....	304
11-10	Example of 4-Trigger Mode (Timer Trigger Select 1-Buffer 4-Trigger) Operation .....	305
11-11	Example of 1-Trigger Mode (Timer Trigger Select 4-Buffer 1-Trigger) Operation .....	306
11-12	Example of 4-Trigger Mode (Timer Trigger Select 4-Buffer 4-Trigger) Operation .....	307
11-13	Example of 1-Trigger Mode (Timer Trigger Scan 1-Trigger) Operation.....	308
11-14	Example of 4-Trigger Mode (Timer Trigger Scan 4-Trigger) Operation.....	309
12-1	Type A Block Diagram .....	316
12-2	Type B Block Diagram .....	317
12-3	Type C Block Diagram .....	318
12-4	Type D Block Diagram .....	319
12-5	Type E Block Diagram .....	320
12-6	Type F Block Diagram .....	321
12-7	Type G Block Diagram .....	321
12-8	Type H Block Diagram .....	322
12-9	Type I Block Diagram.....	322
12-10	Type K Block Diagram .....	323
12-11	Type M Block Diagram.....	324
12-12	Type O Block Diagram .....	325
12-13	Type P Block Diagram .....	326
12-14	Type Q Block Diagram .....	327
A-1	Malfunction When Executing sld Instructions .....	366

## LIST OF TABLES

Table No.	Title	Page
3-1	Program Registers.....	51
3-2	System Register Numbers.....	52
3-3	Interrupt/Exception Table .....	60
4-1	Bus Cycles in Which the Wait Function Is Valid.....	89
4-2	Bus Priority Order.....	96
5-1	Example of DRAM and Address Multiplex Width.....	112
5-2	Example of DRAM Refresh Interval .....	129
5-3	Example of Interval Factor Settings .....	129
6-1	Relationship Between Transfer Type and Transfer Object.....	163
6-2	External Bus Cycle During DMA Transfer .....	163
6-3	Minimum Execution Clock in DMA Cycle .....	168
6-4	DMAAKn Active → DMARQn Inactive Time for Single Transfer to External Memory .....	170
7-1	Interrupt List.....	174
7-2	Interrupt Control Register Addresses and Bits .....	190
8-1	Clock Generator Operation by Power Save Control .....	208
8-2	Operating States When in HALT Mode .....	210
8-3	Operations After HALT Mode Is Released by Interrupt Request.....	211
8-4	Operating States When in IDLE Mode .....	212
8-5	Operating States When in Software STOP Mode.....	214
8-6	Example of Count Time ( $\phi = 5 \times f_{xx}$ ) .....	218
9-1	RPU Configuration List.....	220
9-2	Capture Trigger Signals (TM1n) to 16-Bit Capture Registers .....	237
9-3	Interrupt Request Signals (TM1n) from 16-Bit Compare Registers .....	240
10-1	Default Priority of Interrupt.....	266
10-2	Baud Rate Generator Setup Values .....	284
13-1	Operating State of Each Pin During Reset .....	361
13-2	Initial Values of CPU, Internal RAM, and Internal Peripheral I/O After Reset .....	363
A-1	Instructions That Write to a Register Immediately Before the sld Instructions.....	367

## CHAPTER 1 INTRODUCTION

The V850E/MS2 is one of NEC's "V850 Family™" of single-chip microcontrollers. This chapter gives a simple outline of the V850E/MS2.

### 1.1 Outline

The V850E/MS2 is a 32-/16-bit single-chip microcontroller which uses the V850 Family's "V850E" CPU, and incorporates peripheral functions such as RAM, various types of memory controllers, a DMA controller, real-time pulse unit, serial interface and A/D converter, realizing large volume data processing and sophisticated real-time control.

#### (1) "V850E" CPU included

The "V850E" CPU supports the RISC instruction set, and through the use of basic instructions, each of which can be executed in 1 clock period, and an optimized pipeline, achieves a marked improvement in instruction execution speed. In addition, in order to make it ideal for use in digital servo control, a 32-bit hardware multiplier enables this CPU to support multiply instructions, saturated multiply instructions, bit operation instructions, etc.

Also, through 2-byte basic instructions and instructions compatible with high level languages, etc., the object code efficiency in a C compiler is increased, and the program size can be made more compact.

Further, since the on-chip interrupt controller provides a high speed interrupt response, including processing, this device is suited to high level real-time control fields.

#### (2) External memory interface function

The V850E/MS1 features various on-chip external memory interfaces including separately address configured (24 bits) and data (16 bits) buses, and SRAM and ROM interfaces, as well as on-chip memory controllers that can be directly linked to EDO DRAM, high-speed page DRAM, page ROM, etc., thereby raising the system performance and reducing the number of parts needed for application systems.

Also, through the DMA controller, CPU internal calculations and data transfers can be performed simultaneously with transfers with external memory, so it is possible to process large volumes of image data or voice data, etc., and through the high-speed execution of instructions using internal RAM, motor control, communications control and other real-time control tasks can be realized simultaneously.

#### (3) A full range of middleware and development environment products

The V850E/MS2 can execute middleware such as JPEG, JBIG and MH/MR/MMR at high speed. Also, middleware that enables voice recognition, voice synthesis and other such processing is available; by including these middleware programs, a multimedia system can be easily realized.

A development environment system that includes an optimized C compiler, debugger, in-circuit emulator, simulator, system performance analyzer and other elements is also available.

## 1.2 Features

- Number of instructions: 81
- Minimum instruction execution time: 33 ns (at internal 30 MHz)
- General-purpose registers: 32 bits × 32
- Instruction set:
  - Upwardly compatible with V850 CPU
  - Signed multiplication (16 bits × 16 bits → 32 bits or 32 bits × 32 bits → 64 bits)
  - Saturated operation instructions (with overflow/underflow detection function)
  - 32-bit shift instructions
  - Bit manipulation instructions
  - Load/store instructions with long/short format
  - Signed load instructions
- Memory space:
  - 22 MB linear address space (common program/data use)
  - Chip select output function: 4 spaces
  - Memory block division function: 2, 4, 8 MB/block
  - Programmable wait function
  - Idle state insertion function
- External bus interface:
  - 16-bit data bus (address/data multiplexed)
  - 16-/8-bit bus sizing function
  - Bus hold function
  - External wait function
- Internal memory:
 

Part Number	Internal ROM	Internal RAM
μPD703130	None	4 KB
- Interrupt/exception:
  - External interrupts: 10 (including NMI)
  - Internal interrupts: 35 sources
  - Exceptions: 1 source
  - Eight levels of priorities can be set.
- Memory access controller:
  - DRAM controller (Compatible with EDO DRAM and high-speed page DRAM)
  - Page-ROM controller

- DMA controller:
  - 4 channels
  - Transfer units: 8 bits/16 bits
  - Maximum transfer count: 65,536 ( $2^{16}$ )
  - Transfer type: Flyby (1-cycle)/2-cycle
  - Transfer mode: Single/Single step/Block
  
- I/O lines:
  - Input ports: 5
  - I/O ports: 76
  
- Real-time pulse unit:
  - 16-bit timer/event counter: 4 channels
  - 16-bit timers: 4
  - 16-bit capture/compare registers: 9
  - 16-bit compare registers: 7
  - 16-bit interval timer: 2 channels
  
- Serial interface:
  - Asynchronous serial interface (UART)
  - Clocked serial interface (CSI)
  - UART/CSI: 2 channels
  - Dedicated baud rate generator: 2 channels
  
- A/D converter:
  - 10-bit resolution A/D converter: 4 channels
  
- Clock generator:
  - A multiply-by-five function via a PLL clock synthesizer.
  - A divide-by-two function via external clock input.
  
- Power save function:
  - HALT/IDLE/software STOP mode
  - Clock output stop function
  
- Package:
  - 100-pin plastic LQFP: pin pitch 0.5 mm
  
- CMOS technology:
  - All static circuits

### 1.3 Applications

- OA devices (printers, facsimiles, PPCs, etc.)
- Multimedia devices (digital still cameras, video printers, etc.)
- Consumer appliances (single lens reflex cameras, etc.)
- Industrial devices (motor control, NC machine tools, etc.)

### 1.4 Ordering Information

Part Number	Package	Maximum Operating Frequency	On-chip ROM	HV <sub>DD</sub>
$\mu$ PD703130GC-8EU <sup>Note</sup>	100-pin plastic LQFP (Fine pitch) (14 × 14 mm)	30 MHz	None	4.5 to 5.5 V

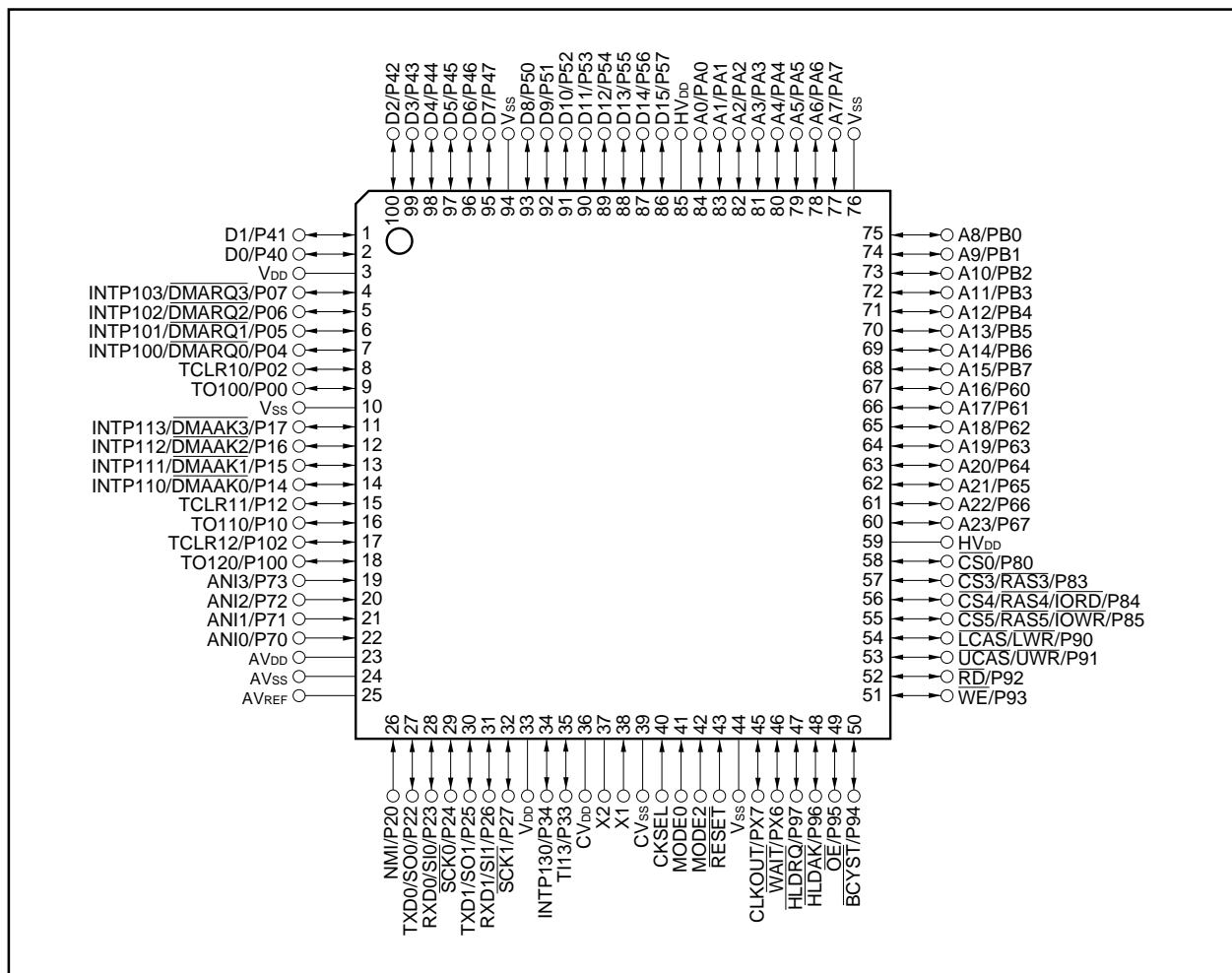
**Note** Under development



## 1.5 Pin Configuration (Top View)

100-pin plastic LQFP (fine pitch) (14 × 14 mm)

- $\mu$ PD703130GC-8EU

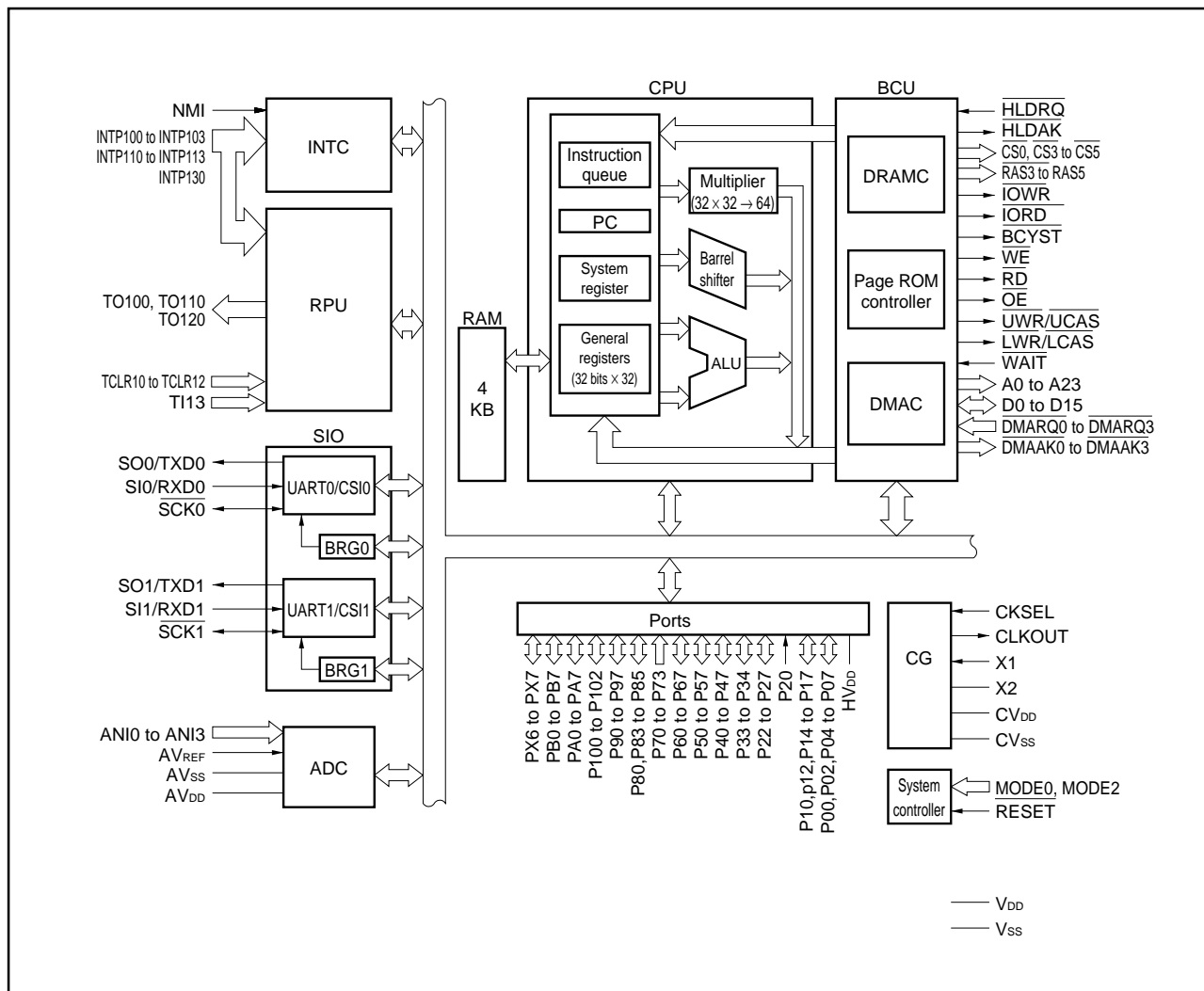


## Pin Name

A0 to A23:	Address Bus	P60 to P67:	Port 6
ANI0 to ANI3:	Analog Input	P70 to P73:	Port 7
AV <sub>DD</sub> :	Analog Power Supply	P80, P83 to P85:	Port 8
AV <sub>REF</sub> :	Analog Reference Voltage	P90 to P97:	Port 9
AV <sub>SS</sub> :	Analog Ground	P100, P102:	Port 10
BCYST:	Bus Cycle Start Timing	PA0 to PA7:	Port A
CKSEL:	Clock Generator Operating Mode Select	PB0 to PB7:	Port B
CLKOUT:	Clock Output	PX6, PX7:	Port X
CS0, CS3 to CS5:	Chip Select	RAS3 to RAS5:	Row Address Strobe
CV <sub>DD</sub> :	Clock Generator Power Supply	RD:	Read
CV <sub>SS</sub> :	Clock Generator Ground	RESET:	Reset
D0 to D15:	Data Bus	RXD0, RXD1:	Receive Data
DMAAK0 to DMAAK3:	DMA Acknowledge	SCK0, SCK1:	Serial Clock
DMARQ0 to DMARQ3:	DMA Request	SI0, SI1:	Serial Input
HLD <sub>AK</sub> :	Hold Acknowledge	SO0, SO1:	Serial Output
HLD <sub>RQ</sub> :	Hold Request	TCLR10 to TCLR12:	Timer Clear
HV <sub>DD</sub> :	Power Supply for External Pins	TI13:	Timer Input
INTP100 to INTP103, INTP110 to INTP113, INTP130:	Interrupt Request from Peripherals	TO100, TO110, TO120:	Timer Output
IORD:	I/O Read Strobe	TXD0, TXD1:	Transmit Data
IOWR:	I/O Write Strobe	UCAS:	Upper Column Address Strobe
LCAS:	Lower Column Address Strobe	UWR:	Upper Write Strobe
LWR:	Lower Write Strobe	V <sub>DD</sub> :	Power Supply for Internal Unit
MODE0, MODE2:	Mode	V <sub>SS</sub> :	Ground
NMI:	Non-Maskable Interrupt Request	WAIT:	Wait
OE:	Output Enable	WE:	Write Enable
P00, P02, P04 to P07:	Port 0	X1, X2:	Crystal
P10, P12, P14 to P17:	Port 1		
P20, P22 to P27:	Port 2		
P33, P34:	Port 3		
P40 to P47:	Port 4		
P50 to P57:	Port 5		

## 1.6 Function Block

### 1.6.1 Internal block diagram



## 1.6.2 Internal units

### (1) CPU

The CPU uses five-stage pipeline control to enable single-clock execution of address calculations, arithmetic logic operations, data transfers, and almost all other instruction processing.

Other dedicated on-chip hardware, such as a multiplier (16 bits  $\times$  16 bits  $\rightarrow$  32 bits or 32 bits  $\times$  32 bits  $\rightarrow$  64 bits) and a barrel shifter (32 bits), help accelerate processing of complex instructions.

### (2) Bus control unit (BCU)

The BCU starts a required external bus cycle based on the physical address obtained by the CPU. When an instruction is fetched from external memory space and the CPU does not send a bus cycle start request, the BCU generates a prefetch address and prefetches the instruction code. The prefetched instruction code is stored in an instruction queue in the CPU.

The BCU incorporates a DRAM controller (DRAMC), page ROM controller, and DMA controller (DMAC).

#### (a) DRAM controller (DRAMC)

This controller generates the  $\overline{\text{RAS}}$ ,  $\overline{\text{UCAS}}$  and  $\overline{\text{LCAS}}$  signals (2CAS control) and controls DRAM access.

It is compatible with high-speed DRAM and EDO DRAM. When accessing DRAM, there are 2 types of cycle; normal access (off page) and page access (on page).

Also, it includes a refresh function that is compatible with the CBR refresh cycle.

#### (b) Page ROM controller

This controller is compatible with ROM that includes a page access function.

It performs address comparisons with the immediately preceding bus cycle and executes wait control for normal access (off page)/page access (on page). It can handle page widths of 8 to 64 bytes.

#### (c) DMA controller (DMAC)

This controller transfers data between memory and I/O in place of the CPU.

There are two address modes, flyby (1 cycle) transfer, and 2-cycle transfer. There are three bus modes, single transfer, single step transfer, and block transfer.

### (3) RAM

4 KB of RAM is mapped from address FFFFE000H. During instruction fetch, data can be accessed from the CPU in 1-clock cycles.

### (4) Interrupt controller (INTC)

This controller handles hardware interrupt requests (NMI, INTP100 to INTP103, INTP110 to INTP113, INTP130) from internal peripheral I/O and external hardware. Eight levels of interrupt priorities can be specified for these interrupt requests, and multiplexed servicing control can be performed for interrupt sources.

### (5) Clock generator (CG)

This clock generator supplies frequencies that are 5 times the input clock (f<sub>xx</sub>) (used by the internal PLL) and 1/2 the input clock (when the internal PLL is not used) as an internal system clock ( $\phi$ ). As the input clock, an external oscillator is connected to pins X1 and X2 (only when an internal PLL synthesizer is used) or an external clock is input from pin X1.

**(6) Real-time pulse unit (RPU)**

This unit has a 4-channel 16-bit timer/event counter and 2-channel 16-bit interval timer on-chip, and it is possible to measure pulse widths or frequency and to output a programmable pulse.

**(7) Serial interface (SIO)**

The serial interface has a total of 2 channels of asynchronous serial interfaces (UART) and synchronous or clocked serial interfaces (CSI). These channels can be switched between UART and CSI.

UART transfers data by using the TXD and RXD pins and the CSI transfers data by using the SO, SI, and  $\overline{\text{SCK}}$  pins.

The serial clock source can be selected from dedicated baud rate generator output or internal system clock.

**(8) A/D converter (ADC)**

This high-speed, high-resolution 10-bit A/D converter includes 4 analog input pins. Conversion uses the successive approximation method.

**(9) Ports**

As shown below, the following ports have general port functions and control pin functions.

Port	Port Function	Control Function
Port 0	6-bit I/O	Real-time pulse unit input/output, external interrupt input, DMA controller input
Port 1	6-bit I/O	Real-time pulse unit input/output, external interrupt input, DMA controller output
Port 2	1-bit input, 6-bit I/O	NMI input, serial interface input/output
Port 3	2-bit I/O	Real-time pulse unit input, external interrupt input
Port 4	8-bit I/O	External data bus
Port 5	8-bit I/O	External data bus
Port 6	8-bit I/O	External address bus
Port 7	4-bit input	A/D converter input
Port 8	4-bit I/O	External bus interface control signal output
Port 9	8-bit I/O	External bus interface control signal input/output
Port 10	2-bit I/O	Real-time pulse unit input/output
Port A	8-bit I/O	External address bus
Port B	8-bit I/O	External address bus
Port X	2-bit I/O	Wait insertion signal input, internal system clock output

[MEMO]

## CHAPTER 2 PIN FUNCTIONS

The names and functions of this product's pins are listed below. These pins can be divided into port pins and non-port pins according to their functions.

### 2.1 List of Pin Functions

#### (1) Port pins

(1/2)

Pin Name	I/O	Function	Alternate Function
P00	I/O	Port 0 6-bit I/O port Input/output mode can be specified in 1-bit units.	TO100
P02			TCLR10
P04			INTP100/ $\overline{\text{DMARQ0}}$
P05			INTP101/ $\overline{\text{DMARQ1}}$
P06			INTP102/ $\overline{\text{DMARQ2}}$
P07			INTP103/ $\overline{\text{DMARQ3}}$
P10	I/O	Port 1 6-bit I/O port Input/output mode can be specified in 1-bit units.	TO110
P12			TCLR11
P14			INTP110/ $\overline{\text{DMAAK0}}$
P15			INTP111/ $\overline{\text{DMAAK1}}$
P16			INTP112/ $\overline{\text{DMAAK2}}$
P17			INTP113/ $\overline{\text{DMAAK3}}$
P20	Input	Port 2 P20 is an input-only port. If a valid edge is input, it operates as an NMI input. Also, the status of the NMI input is shown by bit 0 of the P2 register. P22 to P27 are 6-bit I/O ports. Input/output mode can be specified in 1-bit units.	NMI
P22	I/O		TXD0/SO0
P23			RXD0/SI0
P24			$\overline{\text{SCK0}}$
P25			TXD1/SO1
P26			RXD1/SI1
P27			$\overline{\text{SCK1}}$
P33	I/O	Port 3 2-bit I/O port Input/output mode can be specified in 1-bit units.	TI13
P34			INTP130
P40 to P47	I/O	Port 4 8-bit I/O port Input/output mode can be specified in 1-bit units.	D0 to D7
P50 to P57	I/O	Port 5 8-bit I/O port Input/output mode can be specified in 1-bit units.	D8 to D15
P60 to P67	I/O	Port 6 8-bit I/O port Input/output mode can be specified in 1-bit units.	A16 to A23

(1) Port pins

(2/2)

Pin Name	I/O	Function	Alternate Function
P70 to P73	Input	Port 7 4-bit input-only port	ANI0 to ANI3
P80	I/O	Port 8 4-bit I/O port Input/output mode can be specified in 1-bit units.	$\overline{\text{CS0}}$
P83			$\overline{\text{CS3/RAS3}}$
P84			$\overline{\text{CS4/RAS4/IOWR}}$
P85			$\overline{\text{CS5/RAS5/IORD}}$
P90	I/O	Port 9 8-bit I/O port Input/output mode can be specified in 1-bit units.	$\overline{\text{LCAS/LWR}}$
P91			$\overline{\text{UCAS/UWR}}$
P92			$\overline{\text{RD}}$
P93			$\overline{\text{WE}}$
P94			$\overline{\text{BCYST}}$
P95			$\overline{\text{OE}}$
P96			$\overline{\text{HLDK}}$
P97			$\overline{\text{HLDRQ}}$
P100	I/O	Port 10 2-bit I/O port Input/output mode can be specified in 1-bit units.	TO120
P102			TCLR12
PA0	I/O	Port A 8-bit I/O port Input/output mode can be specified in 1-bit units.	A0
PA1			A1
PA2			A2
PA3			A3
PA4			A4
PA5			A5
PA6			A6
PA7			A7
PB0	I/O	Port B 8-bit I/O port Input/output mode can be specified in 1-bit units.	A8
PB1			A9
PB2			A10
PB3			A11
PB4			A12
PB5			A13
PB6			A14
PB7			A15
PX6	I/O	Port X 2-bit I/O port Input/output mode can be specified in 1-bit units.	$\overline{\text{WAIT}}$
PX7			CLKOUT



## (2) Non-port pins

(1/2)

Pin Name	I/O	Function	Alternate Function
TO100	Output	Pulse signal output of timers 10 to 12	P00
TO110			P10
TO120			P100
TCLR10	Input	External clear signal input of timers 10 to 12	P02
TCLR11			P12
TCLR12			P102
TI13	Input	External count clock input of timer 13	P33
INTP100	Input	External maskable interrupt request input, or timer 10 external capture trigger input	P04/D $\overline{\text{MARQ0}}$
INTP101			P05/D $\overline{\text{MARQ1}}$
INTP102			P06/D $\overline{\text{MARQ2}}$
INTP103			P07/D $\overline{\text{MARQ3}}$
INTP110	Input	External maskable interrupt request input, or timer 11 external capture trigger input	P14/D $\overline{\text{MAAK0}}$
INTP111			P15/D $\overline{\text{MAAK1}}$
INTP112			P16/D $\overline{\text{MAAK2}}$
INTP113			P17/D $\overline{\text{MAAK3}}$
INTP130	Input	External maskable interrupt request input, or timer 13 external capture trigger input	P34
SO0	Input	CSI0, CSI1 serial transmission data output (3-wire)	P22/TXD0
SO1			P25/TXD1
SI0	Input	CSI0, CSI1 serial reception data input (3-wire)	P23/RXD0
SI1			P26/RXD1
$\overline{\text{SCK0}}$	I/O	CSI0, CSI1 serial clock input/output (3-wire)	P24
$\overline{\text{SCK1}}$			P27
TXD0	Output	UART0 and UART1 serial transmission data output	P22/SO0
TXD1			P25/SO1
RXD0	Input	UART0 and UART1 serial reception data input	P23/SI0
RXD1			P26/SI1
D0 to D7	I/O	16-bit data bus for external memory	P40 to P47
D8 to D15			P50 to P57
A0 to A7	Output	24-bit address bus for external memory	PA0 to PA7
A8 to A15			PB0 to PB7
A16 to A23			P60 to P67
$\overline{\text{LWR}}$	Output	External data bus lower byte write enable signal output	P90/L $\overline{\text{CAS}}$
$\overline{\text{UWR}}$	Output	External data bus higher byte write enable signal output	P91/U $\overline{\text{CAS}}$
$\overline{\text{RD}}$	Output	External data bus read strobe signal output	P92
$\overline{\text{WE}}$	Output	Write enable signal output for DRAM	P93
$\overline{\text{OE}}$	Output	Output enable signal output for DRAM	P95

## (2) Non-port pins

(2/2)

Pin Name	I/O	Function	Alternate Function
$\overline{\text{LCAS}}$	Output	Column address strobe signal output for DRAM lower data	P90/ $\overline{\text{LWR}}$
$\overline{\text{UCAS}}$	Output	Column address strobe signal output for DRAM higher data	P91/ $\overline{\text{UWR}}$
$\overline{\text{RAS3}}$	Output	Row address strobe signal output for DRAM	P83/ $\overline{\text{CS3}}$
$\overline{\text{RAS4}}$			P84/ $\overline{\text{CS4}}/\overline{\text{IOWR}}$
$\overline{\text{RAS5}}$			P85/ $\overline{\text{CS5}}/\overline{\text{IORD}}$
$\overline{\text{BCYST}}$	Output	Strobe signal output that shows the start of the bus cycle	P94
$\overline{\text{CS0}}$	Output	Chip select signal output	P80
$\overline{\text{CS3}}$			P83/ $\overline{\text{RAS3}}$
$\overline{\text{CS4}}$			P84/ $\overline{\text{RAS4}}/\overline{\text{IOWR}}$
$\overline{\text{CS5}}$			P85/ $\overline{\text{RAS5}}/\overline{\text{IORD}}$
$\overline{\text{WAIT}}$	Input	Control signal input that inserts a wait in the bus cycle	PX6
$\overline{\text{IOWR}}$	Output	DMA write strobe signal output	P84/ $\overline{\text{RAS4}}/\overline{\text{CS4}}$
$\overline{\text{IORD}}$	Output	DMA read strobe signal output	P85/ $\overline{\text{RAS5}}/\overline{\text{CS5}}$
$\overline{\text{DMARQ0}}$ to $\overline{\text{DMARQ3}}$	Input	DMA request signal input	P04/ $\overline{\text{INTP100}}$ to P07/ $\overline{\text{INTP103}}$
$\overline{\text{DMAAK0}}$ to $\overline{\text{DMAAK3}}$	Output	DMA acknowledge signal output	P14/ $\overline{\text{INTP110}}$ to P17/ $\overline{\text{INTP113}}$
$\overline{\text{HLD\!AK}}$	Output	Bus hold acknowledge output	P96
$\overline{\text{HLDRQ}}$	Input	Bus hold request input	P97
ANI0 to ANI3	Input	Analog inputs to the A/D converter	P70 to P73
NMI	Input	Non-maskable interrupt request input	P20
CLKOUT	Output	System clock output	PX7
CKSEL	Input	Input which specifies the clock generator's operating mode	—
MODE0, MODE2	Input	Operation mode specification	—
$\overline{\text{RESET}}$	Input	System reset input	—
X1	Input	Connects the system clock oscillator. In the case of an external source supplying the clock, it is input to X1.	—
X2	—		—
$\text{AV}_{\text{REF}}$	Input	Reference voltage applied to A/D converter	—
$\text{AV}_{\text{DD}}$	—	Positive power supply to A/D converter	—
$\text{AV}_{\text{SS}}$	—	Ground for A/D converter	—
$\text{CV}_{\text{DD}}$	—	Supplies a positive power supply for the dedicated clock generator.	—
$\text{CV}_{\text{SS}}$	—	Ground potential for the dedicated clock generator	—
$\text{V}_{\text{DD}}$	—	Supplies the positive power supply (internal unit power supply).	—
$\text{HV}_{\text{DD}}$	—	Supplies the positive power supply (external pin power supply).	—
$\text{V}_{\text{SS}}$	—	Ground potential	—

## 2.2 Pin Status

The state of each pin after reset, in a power save mode (software STOP, IDLE, HALT), during bus hold (TH), and in the idle state (TI), is shown below.

Pin \ Operating State	Reset	Software STOP Mode	IDLE Mode	HALT Mode	Bus Hold (TH)	Idle State (TI)
D0 to D15	Hi-Z	Hi-Z (output) — (input)	Hi-Z (output) — (input)	Operating	Hi-Z	Hi-Z
A0 to A23	Hi-Z	Hi-Z	Hi-Z	Operating	Hi-Z	Hold
$\overline{WE}$ , $\overline{OE}$ , $\overline{RD}$ , $\overline{BCYST}$	Hi-Z	Hi-Z	Hi-Z	Operating	Hi-Z	H
$\overline{UWR}$ , $\overline{LWR}$ , $\overline{IORD}$ , $\overline{IOWR}$ , $\overline{CS0}$ , $\overline{CS3}$ to $\overline{CS5}$	Hi-Z	H	H	Operating	Hi-Z	H
$\overline{RAS3}$ to $\overline{RAS5}$	Hi-Z	Operating	Operating	Operating	Hi-Z	Hold <sup>Note</sup>
$\overline{UCAS}$ , $\overline{LCAS}$	Hi-Z	Operating	Operating	Operating	Hi-Z	H
$\overline{HLDRQ}$	—	—	—	Operating	Operating	Operating
$\overline{HLDAK}$	Hi-Z	Hi-Z	Hi-Z	Operating	L	Operating
$\overline{WAIT}$	—	—	—	Operating	—	—
CLKOUT	Operating	L	L	Operating	Operating	Operating
$\overline{DMARQ0}$ to $\overline{DMARQ3}$	—	—	—	Operating	Operating	Operating
$\overline{DMAAK0}$ to $\overline{DMAAK3}$	Hi-Z	H	H	Operating	H	H
INTP100 to INTP103, INTP110 to INTP113, INTP130	—	—	—	Operating	Operating	Operating
NMI	—	Operating	Operating	Operating	Operating	Operating
P00, P02, P04 to P07, P10, P12, P14 to P17, P20, P22 to P27, P33, P34, P40 to P47, P50 to P57, P60 to P67, P70 to P73, P80, P83 to P85, P90 to P97, P100, P102, PA0 to PA7, PB0 to PB7, PX6, PX7	Hi-Z	Hold (output) — (input)	Hold (output) — (input)	Operating	Operating	Operating
TCLR10 to TCLR12	—	—	—	Operating	Operating	Operating
TI13	—	—	—	Operating	Operating	Operating
TO100, TO110, TO120	Hi-Z	Hold	Hold	Operating	Operating	Operating

Pin \ Operating State	Reset	Software STOP Mode	IDLE Mode	HALT Mode	Bus Hold (TH)	Idle State (TI)
SI0, SI1	—	—	—	Operating	Operating	Operating
SO0, SO1	Hi-Z	Hold	Hold	Operating	Operating	Operating
SCK0, SCK1	Hi-Z	Hold (output) — (input)	Hold (output) — (input)	Operating	Operating	Operating
RXD0, RXD1	—	—	—	Operating	Operating	Operating
TXD0, TXD1	Hi-Z	Hold	Hold	Operating	Operating	Operating
ANI0 to ANI3	—	—	—	Operating	Operating	Operating

**Note** In the idle state (TI) just before and just after bus hold, H

**Remark** Hi-Z: High-impedance

Hold: State during immediately preceding external bus cycle is held

H: High-level output

L: Low-level output

—: No sampling of input

#### Cautions when turning on/off power supply

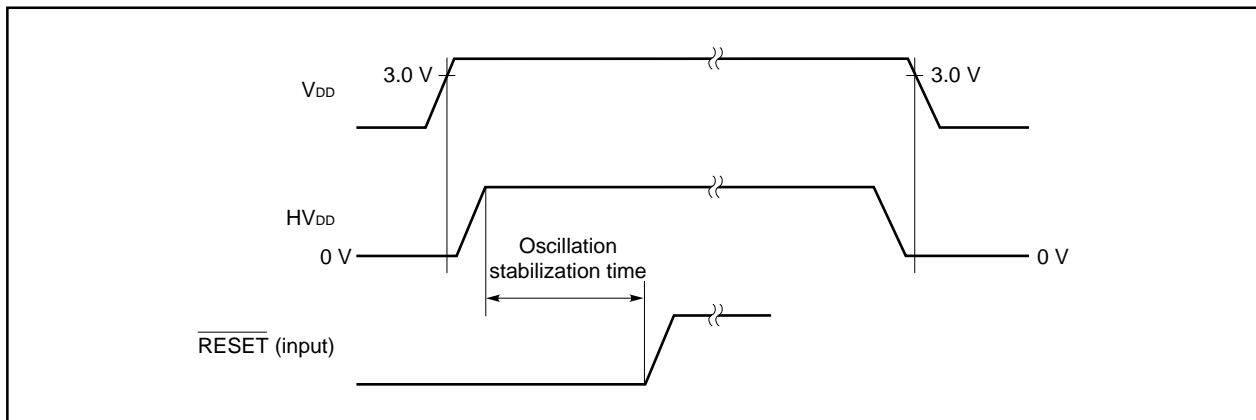
The V850E/MS2 is configured with two power supply pins: the internal unit power supply pin ( $V_{DD}$ ) and the external pin power supply pin ( $HV_{DD}$ ). If the voltage exceeds its operation guaranteed range, the input/output state of the I/O pins may become undefined. If this input/output undefined state causes problems in the system, the pin status can be made high impedance by taking the following countermeasures.

- **When turning on the power**

Apply 0 V to the  $HV_{DD}$  pin until the voltage of the  $V_{DD}$  pin is within the operation guaranteed range (3.0 to 3.6 V).

- **When turning off the power**

Apply a voltage within the operation guaranteed range (3.0 to 3.6 V) to the  $V_{DD}$  pin until the voltage of the  $HV_{DD}$  pin becomes 0 V.



## 2.3 Description of Pin Functions

### (1) P00, P02, P04 to P07 (Port 0) ... 3-state I/O

Port 0 is a 6-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode it operates as the input/output for the real-time pulse unit (RPU), the external interrupt request input and the DMA request input.

The operation mode can be set as port or control in 1-bit units, specified by the port 0 mode control register (PMC0).

#### (a) Port mode

P00, P02, and P04 to P07 can be set to input or output in bit units by the port 0 mode register (PM0).

#### (b) Control mode

P00, P02, and P04 to P07 can be set in the port/control mode in bit units by the PMC0 register.

#### (i) TO100 (Timer Output) ... output

Outputs the pulse signal for timer 1.

#### (ii) TCLR10 (Timer Clear) ... input

This is an input pin for external clear signals for timer 1.

#### (iii) INTP100 to INTP103 (Interrupt Request from Peripherals) ... input

These are input pins for external interrupt requests for timer 1.

#### (iv) DMARQ0 to DMARQ3 (DMA Request) ... input

These are DMA service request signals. They correspond to DMA channels 0 to 3, respectively, and operate independently of each other. The priority order is fixed at DMARQ0 > DMARQ1 > DMARQ2 > DMARQ3.

This signal is sampled when the CLKOUT signal falls. Maintain the active level until a DMA request is received.

**(2) P10, P12, P14 to P17 (Port 1) ... 3-state I/O**

Port 1 is a 6-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode it operates as the input/output for the real-time pulse unit (RPU), the external interrupt request input and the DMA request input.

The operation mode can be set as port or control in 1-bit units, specified by the port 1 mode control register (PMC1).

**(a) Port mode**

P10, P12, and P14 to P17 can be set to input or output in bit units by the port 1 mode register (PM1).

**(b) Control Mode**

P10, P12, and P14 to P17 can be set in the port/control mode in bit units by the PMC1 register.

**(i) TO110 (Timer Output) ... output**

Outputs the pulse signal for timer 1.

**(ii) TCLR11 (Timer Clear) ... input**

This is an input pin for external clear signals for timer 1.

**(iii) INTP110 to INTP113 (Interrupt Request from Peripherals) ... input**

These are input pins for external interrupt requests for timer 1.

**(iv) DMAAK0 to DMAAK3 (DMA Acknowledge) ... output**

This signal shows that a DMA service request was acknowledged.

They correspond to DMA channels 0 to 3, respectively, and operate independently of each other.

These signals become active only when external memory is being accessed. When DMA transfers are being executed between internal RAM and internal peripheral I/O, they do not become active.

These signals are activated on the falling of the CLKOUT signal in the T0, T1R, or T1FH state of the DMA cycle, and are retained at the active level during DMA transfers.

**(3) P20, P22 to P27 (Port 2) ... 3-state I/O**

Port 2, except for P20, which is an input-only pin, is an input/output port which can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode it operates as the input/output for the serial interface (UART0/CSI0, UART1/CSI1).

The operation mode can be set as port or control in 1-bit units, specified by the port 2 mode control register (PMC2).

**(a) Port mode**

P22 to P27 can be set to input or output in bit units by the port 2 mode register (PM2). P20 is an exclusive input port, and if a valid edge is input, it operates as an NMI input.

**(b) Control mode**

P22 to P27 can be set in the port/control mode in bit units by the PMC2 register.

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

This is the input pin for non-maskable interrupt requests.

**(ii) TXD0, TXD1 (Transmit Data) ... output**

Output UART0, UART1 serial transmit data.

**(iii) RXD0, RXD1 (Receive Data) ... input**

Input UART0, UART1 serial receive data.

**(iv) SO0, SO1 (Serial Output) ... output**

Output CSI0, CSI1 serial transmit data.

**(v) SI0, SI1 (Serial Input) ... input**

Input CSI0, CSI1 serial receive data.

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

These are the input/output pins for the CSI0, CSI1 serial clock.

**(4) P33, P34 (Port 3) ... 3-state I/O**

Port 3 is a 2-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode it operates as the input for the real-time pulse unit (RPU) and the external request input. The operation mode can be set as port or control in 1-bit units, specified by the port 3 mode control register (PMC3).

**(a) Port mode**

P33, P34 can be set to input or output in bit units by the port 3 mode register (PM3).

**(b) Control mode**

P33, P34 can be set in the port/control mode in bit units by the PMC3 register.

**(i) TI13 (Timer Input) ... input**

This is an input pin for an external count clock for timer 1.

**(ii) INTP130 (Interrupt Request from Peripherals) ... input**

This is an input pin for external interrupt requests for timer 1.

**(5) P40 to P47 (Port 4) ... 3-state I/O**

Port 4 is an 8-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode (external expansion mode) it operates as a data bus (D0 to D7) when memory is externally expanded.

The operation mode is specified by the mode specification pins (MODE0 and MODE2) and the memory expansion mode register (MM).

**(a) Port mode**

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

**(b) Control mode (External expansion mode)**

P40 to P47 can be set as D0 to D7 by using the MODE0 and MODE2 pins and MM register.

**(i) D0 to D7 (Data) ... 3-state I/O**

These pins constitute the data bus that is used for external access. They operate as the lower 8-bit input/output bus pins for 16-bit data. The output changes in synchronization with the falling of the clock in the T1 state CLKOUT signal of the bus cycle. In the idle state (TI), the impedance becomes high.



**(6) P50 to P57 (Port 5) ... 3-state I/O**

Port 5 is an 8-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as an I/O port, in the control mode (external expansion mode) it operates as a data bus (D8 to D15) when memory is externally expanded.

The operation mode is specified by the mode specification pins (MODE0 and MODE2) and the memory expansion mode register (MM).

**(a) Port mode**

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

**(b) Control mode (External expansion mode)**

P50 to P57 can be set as D8 to D15 by using the MODE0 and MODE2 pins and MM register.

**(i) D8 to D15 (Data) ... 3-state I/O**

These pins constitute the data bus that is used for external access. They operate as the higher 8-bit input/output bus pins for 16-bit data. The output changes in synchronization with the falling of the clock in the T1 state CLKOUT signal of the bus cycle. In the idle state (TI), the impedance becomes high.

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

Port 6 is an 8-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode (external expansion mode) it operates as an address bus (A16 to A23) when memory is externally expanded.

The operation mode can be set as port or control in 2-bit units, specified by the mode specification pins (MODE0 and MODE2) and the memory expansion mode register (MM).

**(a) Port mode**

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

**(b) Control mode (External expansion mode)**

P60 to P67 can be set as A16 to A23 by using the MODE0 and MODE2 pins and MM register.

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

These pins constitute the higher 8-bits of a 24-bits address bus when the external memory is accessed. The output changes in synchronization with the falling edge of the CLKOUT signal in the T1 state of the bus cycle. In the idle state (TI), the previous bus cycle's address is held.

**(8) P70 to P73 (Port 7) ... input**

Port 7 is a 4-bit input-only port in which all pins are fixed as input pins.

Besides functioning as a port, in the control mode it operates as analog input for the A/D converter. However, the input port and analog input pin cannot be switched.

**(a) Port mode**

P70 to P73 are input-only pins.

**(b) Control mode**

P70 to P73 function alternately pins ANI0 to ANI3, but these alternate functions are not switchable.

**(i) ANI0 to ANI3 (Analog Input) ... input**

These are analog input pins for the A/D converter.

Connect a capacitor between these pins and AV<sub>SS</sub> to prevent noise-related operation faults. Also, do not apply voltage that is outside the range for AV<sub>SS</sub> and AV<sub>REF</sub> to pins that are being used as inputs for the A/D converter. If it is possible for noise above the AV<sub>REF</sub> range or below the AV<sub>SS</sub> to enter, clamp these pins using a diode that has a small V<sub>F</sub> value.

**(9) P80, P83 to P85 (Port 8) ... 3-state I/O**

Port 8 is a 4-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode it operates as a control signal output when memory and peripheral I/O are externally expanded.

The operation mode can be set as port or control in 1-bit units, specified by the port 8 mode control register (PMC8).

**(a) Port mode**

P80, P83 to P85 can be set to input or output in bit units by the port 8 mode register (PM8).

**(b) Control mode**

P80, P83 to P85 can be set in the port/control mode in bit units by the PMC8 register.

**(i)  $\overline{CS0}$ ,  $\overline{CS3}$  to  $\overline{CS5}$  (Chip Select) ... 3-state output**

This is the chip select signal for SRAM, external ROM, external peripheral I/O, and page ROM area.

The  $\overline{CSn}$  signal is assigned to memory block n (n = 0, 3 to 5).

It becomes active at the time the bus cycle when the corresponding memory block is accessed starts.

In the idle state (TI), it becomes inactive.

**(ii)  $\overline{RAS3}$  to  $\overline{RAS5}$  (Row Address Strobe) ... 3-state output**

This is the strobe signal for the row address for the DRAM area and the strobe signal for the CBR refresh cycle.

The  $\overline{RASn}$  signal is assigned to memory block n (n = 3 to 5).

During on-page disable, after the DRAM access bus cycle ends, it becomes inactive.

During on-page enable, even after the DRAM access bus cycle ends, it is kept in the active state.

During the reset period and during a hold period, it is in the high impedance state, so connect it to HV<sub>DD</sub> via a resistor.

**(iii)  $\overline{\text{IORD}}$  (I/O Read) ... 3-state output**

This is the read strobe signal for external I/O during DMA flyby transfer. It indicates whether the bus cycle currently being executed is a read cycle for external I/O during flyby transfer, or a read cycle for the SRAM area.

In order to make it possible to connect directly to memory or external I/O during DMA flyby transfer,  $\overline{\text{UWR}}$  or  $\overline{\text{LWR}}$  rises before  $\overline{\text{IORD}}$  rises.

**(iv)  $\overline{\text{IOWR}}$  (I/O Write) ... 3-state output**

This is the write strobe signal for external I/O during DMA flyby transfer. It indicates whether the bus cycle currently being executed is a write cycle for external I/O during flyby transfer, or a write cycle for the SRAM area.

In order to make it possible to connect directly to memory or external I/O during DMA flyby transfer,  $\overline{\text{IOWR}}$  rises before  $\overline{\text{RD}}$  rises.

**(10) P90 to P97 (Port 9) ... 3-state I/O**

Port 9 is an 8-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode it operates as a control signal output and bus hold control signal input/output when memory is externally expanded.

The operation mode can be set as port or control in 1-bit units, specified by the port 9 mode control register (PMC9).

**(a) Port mode**

P90 to P97 can be set to input or output in bit units by the port 9 mode register (PM9).

**(b) Control mode**

P90 to P97 can be set in the port/control mode in bit units by the PMC9 register.

**(i)  $\overline{\text{LCAS}}$  (Lower Column Address Strobe) ... 3-state output**

This is the strobe signal for column address for DRAM and the strobe signal for the CBR refresh cycle.

In the data bus, the lower byte is valid.

**(ii)  $\overline{\text{UCAS}}$  (Upper Column Address Strobe) ... 3-state output**

This is the strobe signal for column address for DRAM and the strobe signal for the CBR refresh cycle.

In the data bus, the higher byte is valid.

**(iii)  $\overline{\text{LWR}}$  (Lower Byte Write Strobe) ... 3-state output**

This strobe signal shows whether the bus cycle currently being executed is a write cycle for the SRAM, external ROM, external peripheral I/O, or page ROM.

In the data bus, the lower byte becomes valid. If the bus cycle is a lower memory write, it becomes active at the rise of the T1 state's CLKOUT signal and becomes inactive at the rise of the T2 state's CLKOUT signal.

**(iv)  $\overline{\text{UWR}}$  (Upper Byte Write Strobe) ... 3-state output**

This strobe signal shows whether the bus cycle currently being executed is a write cycle for the SRAM, external ROM, external peripheral I/O, or page ROM.

In the data bus, the higher byte becomes valid. If the bus cycle is a higher memory write, it becomes active at the rise of the T1 state's CLKOUT signal and becomes inactive at the rise of the T2 state's CLKOUT signal.

**(v)  $\overline{RD}$  (Read Strobe) ... 3-state output**

This strobe signal shows that the bus cycle currently being executed is a read cycle for the SRAM, external ROM, external peripheral I/O, or page ROM.

In the idle state (TI), it becomes inactive.

**(vi)  $\overline{WE}$  (Write Enable) ... 3-state output**

This signal shows that the bus cycle currently being executed is a write cycle for the SRAM area. In the idle state (TI), it becomes inactive.

**(vii)  $\overline{BCYST}$  (Bus Cycle Start Timing) ... 3-state output**

This outputs a status signal showing the start of the bus cycle. It becomes active for 1 clock cycle from the start of each cycle.

In the idle state (TI), it becomes inactive.

**(viii)  $\overline{OE}$  (Output Enable) ... 3-state output**

This signal shows that the bus cycle currently being executed is a read cycle for the DRAM area.

In the idle state (TI), it becomes inactive.

**(ix)  $\overline{HLD\overline{AK}}$  (Hold Acknowledge) ... output**

In this mode, this pin is the output pin for the acknowledge signal that indicates high impedance status for the address bus, data bus, and control bus when the V850E/MS2 receives a bus hold request.

While this signal is active, the impedance of the address bus, data bus and control bus becomes high and the bus mastership is transferred to the external bus master.

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

In this mode, this pin is the input pin by which an external device requests the V850E/MS2 to release the address bus, data bus, and control bus. This pin accepts asynchronous input for the CLKOUT signal. When this pin is active, the address bus, data bus, and control bus are set to high impedance. This occurs either when the V850E/MS2 completes execution of the current bus cycle or immediately if no bus cycle is being executed, then the  $\overline{HLD\overline{AK}}$  signal is set as active and the bus is released.

In order to make the bus hold state secure, keep the  $\overline{HLDRQ}$  signal active until the  $\overline{HLD\overline{AK}}$  signal is output.

**(11) P100, P102 (Port 10) ... 3-state I/O**

Port 10 is a 2-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode it operates as an output for real time pulse unit (RPU).

The operation mode can be set as port or control in 1-bit units, specified by the port 10 mode control register (PMC10).

**(a) Port mode**

P100 and P102 can be set to input or output in bit units by the port 10 mode register (PM10).

**(b) Control mode**

P100 and P102 can be set in the port/control mode in bit units by the PMC10 register.

**(i) TO120 (Timer Output) ... output**

Outputs the pulse signal of timer 1.

**(ii) TCLR12 (Timer Clear) ... input**

This is an input pin for external clear signals for timer 1.

**(12) PA0 to PA7 (Port A) ... 3-state I/O**

Port A is an 8-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode (external expansion mode) it operates as an address bus (A0 to A7) when memory is externally expanded.

The operation mode is specified by the mode specification pins (MODE0, MODE2) and the memory expansion mode register (MM).

**(a) Port mode**

PA0 to PA7 can be set to input or output in bit units by the port A mode register (PMA).

**(b) Control mode (External expansion mode)**

PA0 to PA7 can be set as A0 to A7 by using the MODE0 and MODE2 pins and MM register.

**(i) A0 to A7 (Address) ... output**

These pins constitute the address bus that is used for external access. The output changes in synchronization with the falling of the CLKOUT signal in the T1 state of the bus cycle. In the idle state (TI), the previous bus cycle's address is held.

**(13) PB0 to PB7 (Port B) ... 3-state I/O**

Port B is an 8-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode (external expansion mode) it operates as an address bus (A8 to A15) when memory is externally expanded.

The operation mode can be set as port or control in 2-bit or 4-bit units, specified by the mode specification pins (MODE0, MODE2) and memory expansion mode register (MM).

**(a) Port mode**

PB0 to PB7 can be set to input or output in bit units by the port B mode register (PMB).

**(b) Control mode (External expansion mode)**

PB0 to PB7 can be set as A8 to A15 by using the MODE0 and MODE2 pins and MM register.

**(i) A8 to A15 (Address) ... output**

These pins constitute the address bus when the external memory is accessed.

The output changes in synchronization with the rising edge of the CLKOUT signal in the T1 state of the bus cycle. In the idle state (TI), the impedance becomes high.

**(14) PX6, PX7 (Port X) ... 3-state I/O**

Port X is a 2-bit input/output port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode it operates as a wait insertion signal input and system clock output.

The operation mode can be set as port or control in 1-bit units, specified by the port X mode control register (PMCX).

**(a) Port mode**

PX6 and PX7 can be set to input or output in bit units by the port X mode register (PMX).

**(b) Control mode**

PX6 and PX7 can be set in the port/control mode in bit units by the PMCX register.

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

This is the control signal input pin that inserts a data wait in the bus cycle, and it can be input asynchronously with respect to the CLKOUT signal. When the CLKOUT signal falls, sampling is executed. When the set/hold time is not terminated within the sampling timing, the wait insertion may not be executed.

**(ii) CLKOUT (Clock Output) ... output**

This is the internal system clock output pin. Output from the CLKOUT pin can be executed even during reset.

**(15) CKSEL (Clock Generator Operating Mode Select) ... input**

This is the input pin that specifies the clock generator's operation mode.

Make sure the input level does not change during operation.

**(16) MODE0, MODE2 (Mode) ... input**

These are the input pins that specify the operation mode. There are two operation modes: ROM-less modes 0 and 1 (for details, refer to **3.3 Operation Modes**). The operation mode is determined by sampling the status of each of the MODE0 and MODE2 pins during reset.

Note that this status must be fixed so that the input level does not change during operation.

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

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

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

$\overline{\text{RESET}}$  input is asynchronous input for a signal that has a constant low-level width regardless of the operating clock's status. When this signal is input, a system reset is executed as the first priority ahead of all other operations.

In addition to being used for ordinary initialization/start operations, this pin can also be used to release a power save mode (HALT, IDLE, or software STOP).

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

These pins are used to connect the resonator that generates the system clock.

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

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

This pin supplies positive power to the clock generator.

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

This is the ground pin of the clock generator.

**(21)  $\text{V}_{\text{DD}}$  (Power Supply for Internal Unit)**

These are the positive power supply pins for each internal unit. All the  $\text{V}_{\text{DD}}$  pins should be connected to a positive power source (3.3 V).

**(22)  $\text{HV}_{\text{DD}}$  (Power Supply for External Pins)**

These are the positive power supply pins for external pins. All the  $\text{HV}_{\text{DD}}$  pins should be connected to a positive power source (5 V).

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

These are ground pins. All the  $\text{V}_{\text{SS}}$  pins should be connected to ground.

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

This is the analog power supply pin for the A/D converter.

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

This is the ground pin for the A/D converter.

**(26)  $\text{AV}_{\text{REF}}$  (Analog Reference Voltage) ... input**

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

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

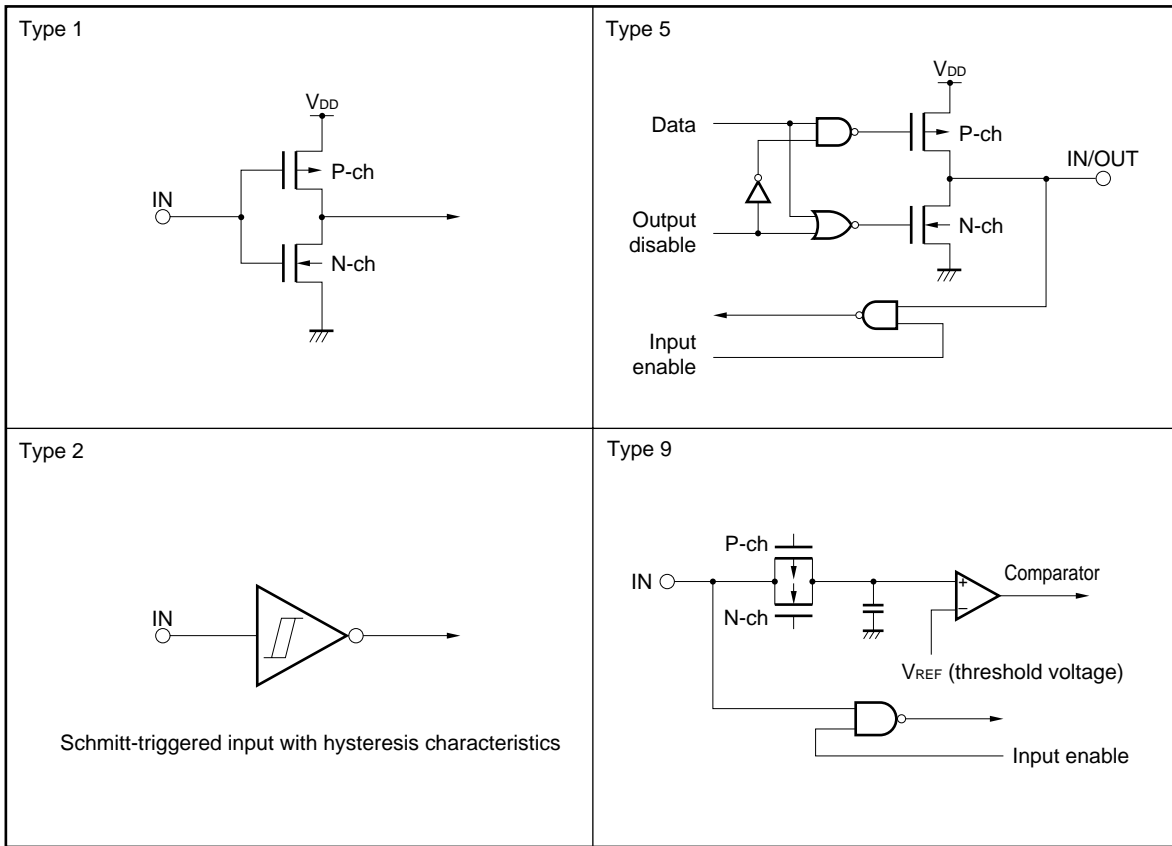
If connecting to  $V_{DD}$  or  $V_{SS}$  via resistors, it is recommended that 1 to 10 k $\Omega$  resistors be connected.

Pin Name	I/O Circuit Type	Recommended Connection of Unused Pins
P00/TO100	5	Input: Independently connect to $HV_{DD}$ or $V_{SS}$ via a resistor. Output: Leave open.
P02/TCLR10		
P04/INTP100/ $\overline{DMARQ0}$ to P07/INTP103/ $\overline{DMARQ3}$		
P10/TO110		
P12/TCLR11		
P14/INTP110/ $\overline{DMAAK0}$ to P17/INTP113/ $\overline{DMAAK3}$		
P20/NMI	2	Connect directly to $V_{SS}$ .
P22/TXD0/SO0	5	Input: Independently connect to $HV_{DD}$ or $V_{SS}$ via a resistor. Output: Leave open.
P23/RXD0/SI0		
P24/ $\overline{SCK0}$		
P25/TXD1/SO1		
P26/RXD1/SI1		
P27/ $\overline{SCK1}$		
P33/TI13		
P34/INTP130		
P40/D0 to P47/D7		
P50/D8 to P57/D15		
P60/A16 to P67/A23		
P70/ANI0 to P73/ANI3	9	Connect directly to $V_{SS}$ .
P80/ $\overline{CS0}$ , P83/ $\overline{CS3}$ / $\overline{RAS3}$	5	Input: Independently connect to $HV_{DD}$ or $V_{SS}$ via a resistor. Output: Leave open.
P84/ $\overline{CS4}$ / $\overline{RAS4}$ / $\overline{IOWR}$ , P85/ $\overline{CS5}$ / $\overline{RAS5}$ / $\overline{IORD}$		
P90/ $\overline{LCAS}$ / $\overline{LWR}$		
P91/ $\overline{UCAS}$ / $\overline{UWR}$		
P92/ $\overline{RD}$		
P93/ $\overline{WE}$		
P94/ $\overline{BCYST}$		
P95/ $\overline{OE}$		
P96/ $\overline{HLDAK}$		
P97/ $\overline{HLDRQ}$		
P100/TO120		



Pin Name	I/O Circuit Type	Recommended Connection of Unused Pins
P102/TCLR12	5	Input: Independently connect to HV <sub>DD</sub> or V <sub>SS</sub> via a resistor. Output: Leave open.
PA0/A0 to PA7/A7		
PB0/A8 to PB7/A15		
PX6/WAIT		
PX7/CLKOUT		
CKSEL	1	Connect directly to HV <sub>DD</sub> .
RESET	2	—
MODE0, MODE2		
AV <sub>REF</sub> , AV <sub>SS</sub>	—	Connect directly to V <sub>SS</sub> .
AV <sub>DD</sub>	—	Connect directly to HV <sub>DD</sub> .

## 2.5 Pin I/O Circuits



**Caution** Note that  $V_{DD}$  in the circuit diagram is replaced by  $HV_{DD}$ .

## CHAPTER 3 CPU FUNCTION

The CPU of the V850E/MS2 is based on RISC architecture and executes the instructions using 5-stage pipeline control.

### 3.1 Features

- Minimum instruction execution time: 33 ns (at internal 30 MHz operation)
- Memory space    Program space: 64 MB linear  
                              Data space: 4 GB linear
- Thirty-two 32-bit general-purpose registers
- Internal 32-bit architecture
- Five-stage pipeline control
- Multiplication/division instructions
- Saturated operation instructions
- 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 V850E/MS2 can be classified into two categories: a general-purpose program register set and a dedicated system register set. The size of the registers is 32 bits.

For details, refer to **V850E/MS1 User's Manual Architecture**.

#### (1) Program register set

31	0
r0	Zero Register
r1	Reserved for Address Generation
r2	
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

#### (2) 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
31	0
CTPC	CALLT Caller PC
CTPSW	CALLT Caller PSW
31	0
DBPC	ILGOP Caller PC
DBPSW	ILGOP Caller PSW
31	0
CTBP	CALLT Base Pointer

### 3.2.1 Program register set

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

#### (1) General-purpose registers

Thirty-two general-purpose 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, r3, r4, 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. r2 may be used by the real-time OS. When not being used by the real-time OS, r2 can be used as a variable register.

**Table 3-1. Program Registers**

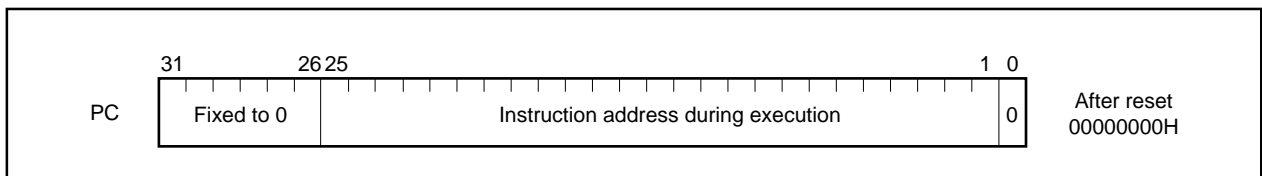
Name	Usage	Operation
r0	Zero register	Always holds 0
r1	Assembler-reserved register	Working register for generating 32-bit immediate data
r2	Address/data variable register (when this register is not used by the real-time OS)	
r3	Stack pointer	Used to generate stack frame when function is called
r4	Global pointer	Used to access global variable in data area
r5	Text pointer	Register to indicate the start of the text area (where program code is located)
r6 to r29	Address/data variable registers	
r30	Element pointer	Base pointer when memory is accessed
r31	Link pointer	Used by compiler when calling function
PC	Program counter	Holds instruction address during program execution

#### (2) Program counter

This register holds the instruction address during program execution. The lower 26 bits of this register are valid, and bits 31 to 26 are fixed to 0. If a carry occurs from bit 25 to 26, 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 register during interrupt	These registers save the PC and PSW when a software 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 register during NMI	These registers save the PC and PSW when an NMI occurs.
3	FEPSW		
4	ECR	Interrupt source register	If an exception, maskable interrupt, or NMI occurs, this register will contain information referencing the interrupt source. The higher 16 bits of this register are called FECC, to which the exception code of the NMI is set. The lower 16 bits are called EICC, to which the exception code of the exception/interrupt is set. Refer to <b>Figure 3-2</b> .
5	PSW	Program status word	The program status word is a collection of flags that indicate the program status (instruction execution result) and CPU status. Refer to <b>Figure 3-3</b> .
16	CTPC	Status saving register during CALLT execution	If the CALLT instruction is executed, this register saves the PC and PSW.
17	CTPSW		
18	DBPC	Status saving register during exception trap	If an exception trap is generated due to detection of an illegal instruction code, this register saves the PC and PSW.
19	DBPSW		
20	CTBP	CALLT base pointer	This is used to specify the table address and generate the target address.
6 to 15	Reserved		
21 to 31			

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

**Figure 3-2. Interrupt Source Register (ECR)**

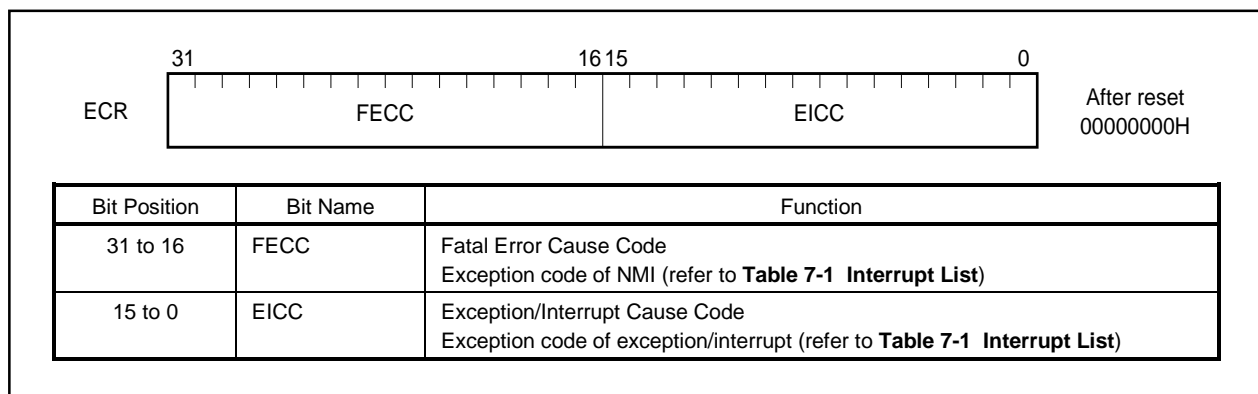
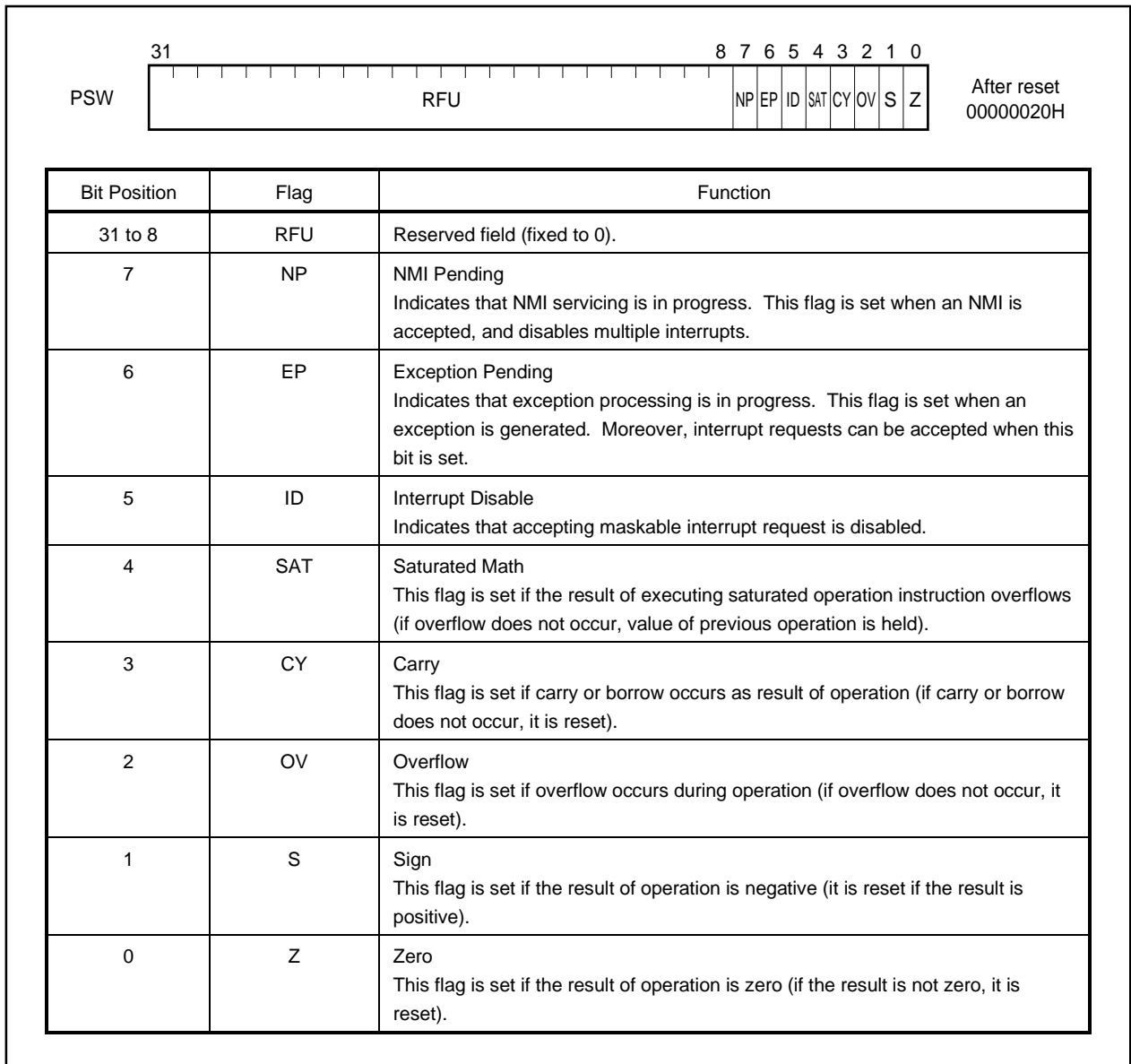


Figure 3-3. Program Status Word (PSW)



### 3.3 Operation Modes

#### 3.3.1 Operation modes

The V850E/MS2 has the following operation modes. Mode specification is carried out by pins MODE0 and MODE2.

##### (1) Normal operation mode

###### (a) ROM-less modes 0, 1

After system reset is cancelled, each pin related to the bus interface enters the control mode, branches to the external device (memory) reset entry address and starts instruction processing.

In ROM-less mode 0, the data bus is a 16-bit data bus and in ROM-less mode 1, the data bus is an 8-bit data bus.

#### 3.3.2 Operation mode specification

The operation mode is specified according to the status of pins MODE0 and MODE2. In an application system fix the specification of these pins and do not change them during operation.

Operation is not guaranteed if these pins are changed during operation.

MODE2	MODE0	Operation Mode		External Data Bus Width	Remarks
L	L	Normal operation mode	ROM-less mode 0	16 bits	—
L	H		ROM-less mode 1	8 bits	
Other than above		Setting prohibited		—	

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



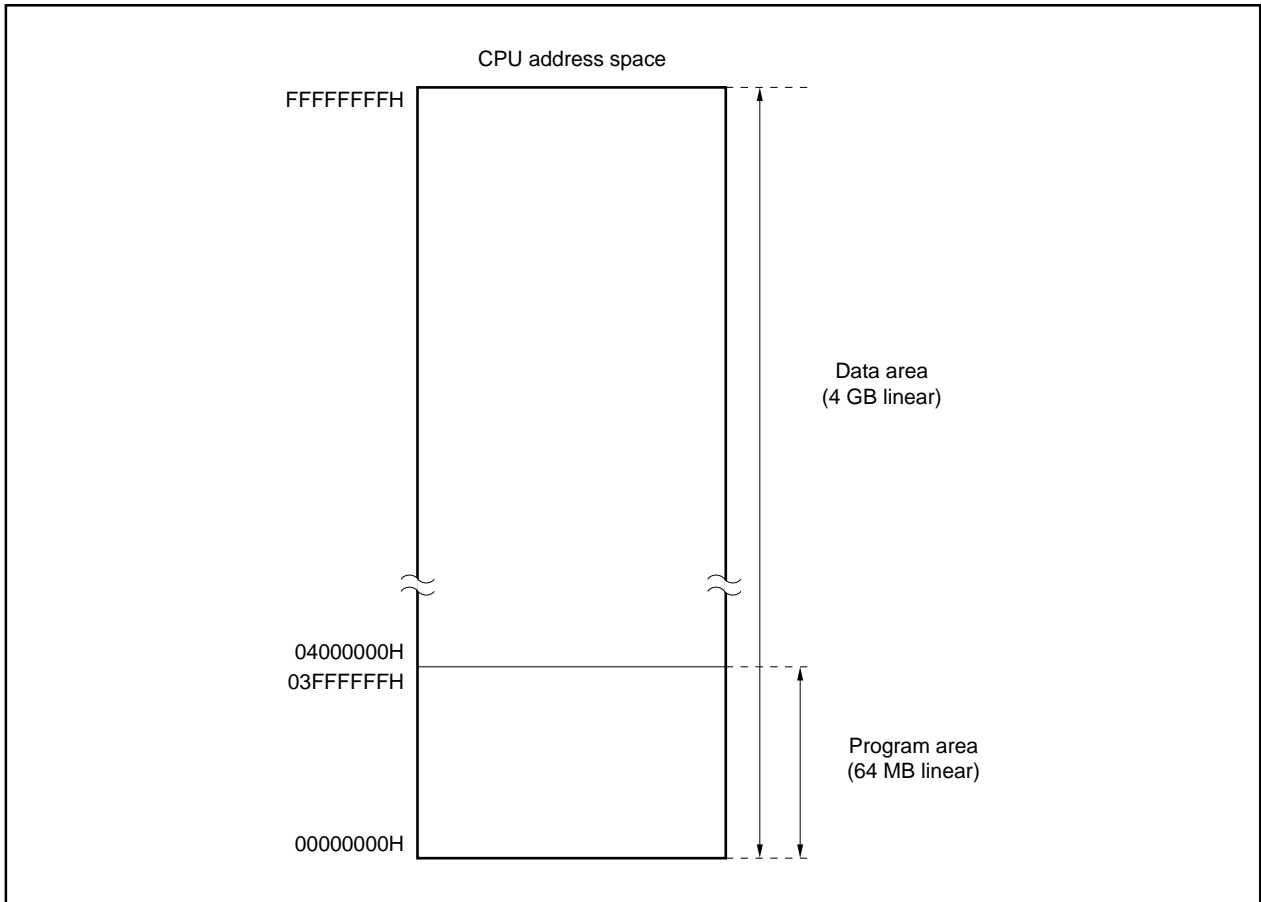
### 3.4 Address Space

#### 3.4.1 CPU address space

The CPU of the V850E/MS2 is of 32-bit architecture and supports up to 4 GB of linear address space (data space) during operand addressing (data access). Also, in instruction address addressing, a maximum of 64 MB of linear address space (program space) is supported.

Figure 3-4 shows the CPU address space.

Figure 3-4. CPU Address Space

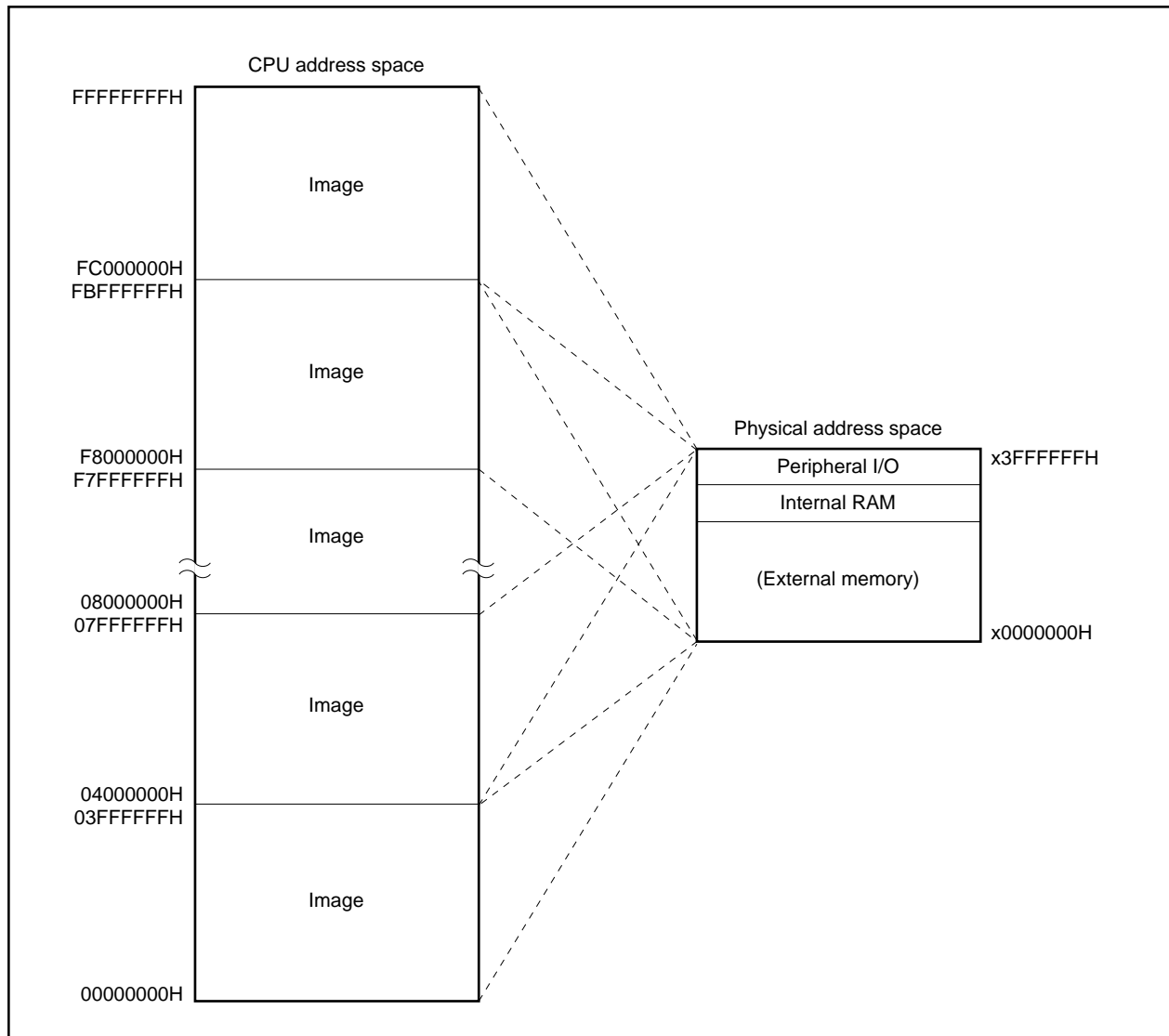


### 3.4.2 Image

The core CPU supports 4 GB of “virtual” addressing space, or 64 memory blocks, each containing 64 MB physical address space. In actuality, the same 64 MB physical address space is accessed regardless of the values of bits 31 to 26 of the CPU address. Figure 3-5 shows the image of the virtual addressing space.

Because the higher 6 bits of a 32-bit CPU address are disregarded and access is made to a 26-bit physical address, physical address x0000000H can be seen as CPU address 00000000H, and in addition, can be seen as address 04000000H, address 08000000H, address F8000000H or address FC000000H.

Figure 3-5. Image on Address Space



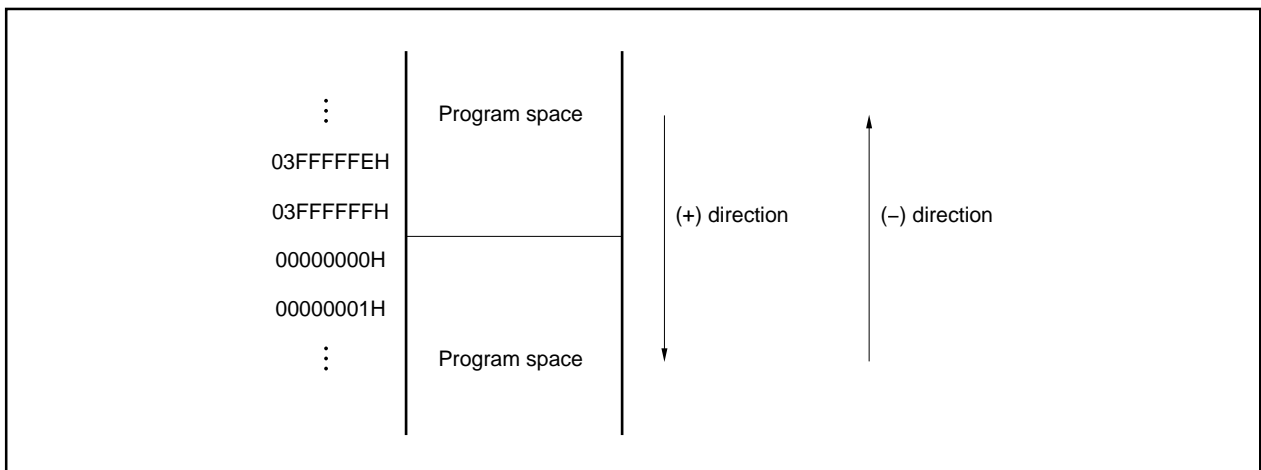
### 3.4.3 Wrap-around of CPU address space

#### (1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are set to 0, and only the lower 26 bits are valid. Even if a carry or borrow occurs from bit 25 to 26 as a result of branch address calculation, the higher 6 bits ignore the carry or borrow.

Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address 03FFFFFFH become contiguous addresses. Wrap-around refers to the situation that the lower-limit address and upper-limit address become contiguous like this.

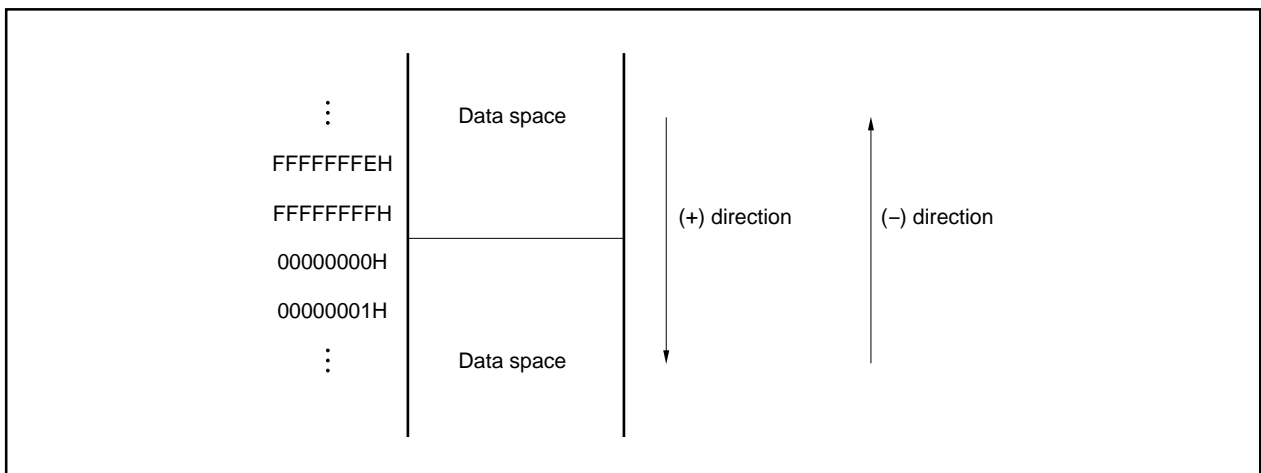
**Caution** No instruction can be fetched from the 4 KB area of 03FFF000H to 03FFFFFFH because this area is defined as the peripheral I/O area. Therefore, do not execute any branch address calculation in which the result will reside in any part of this area.



#### (2) Data space

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

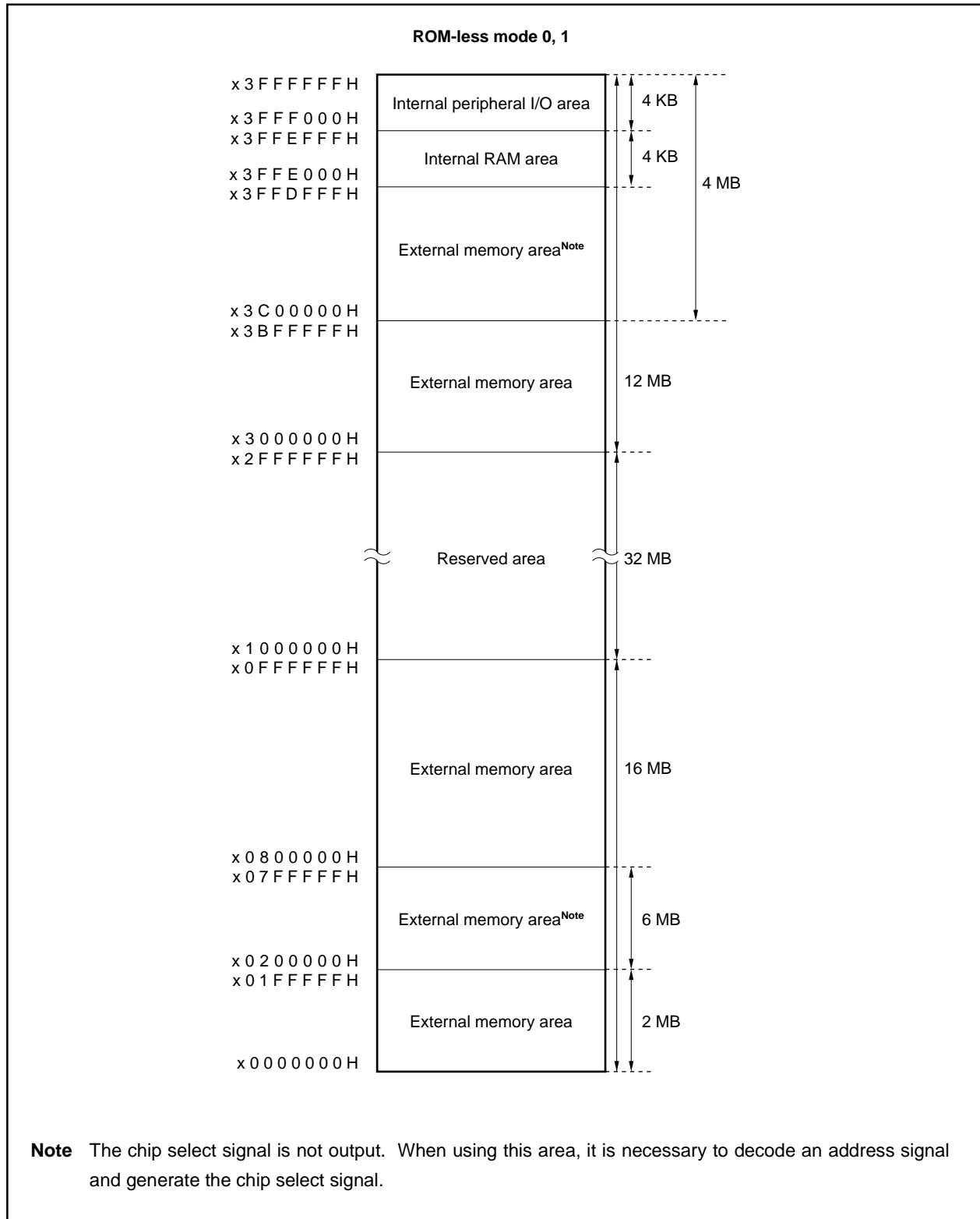
Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address FFFFFFFFH are contiguous addresses, and the data space is wrapped around at the boundary of these addresses.



### 3.4.4 Memory map

The V850E/MS2 reserves areas as shown below.

Each mode is specified by the MM register and the MODE0 and MODE2 pins.



### 3.4.5 Area

#### (1) Interrupt/exception table

The V850E/MS2 increases the interrupt response speed by assigning handler addresses corresponding to interrupts/exceptions.

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

Table 3-3. Interrupt/Exception Table (1/2)

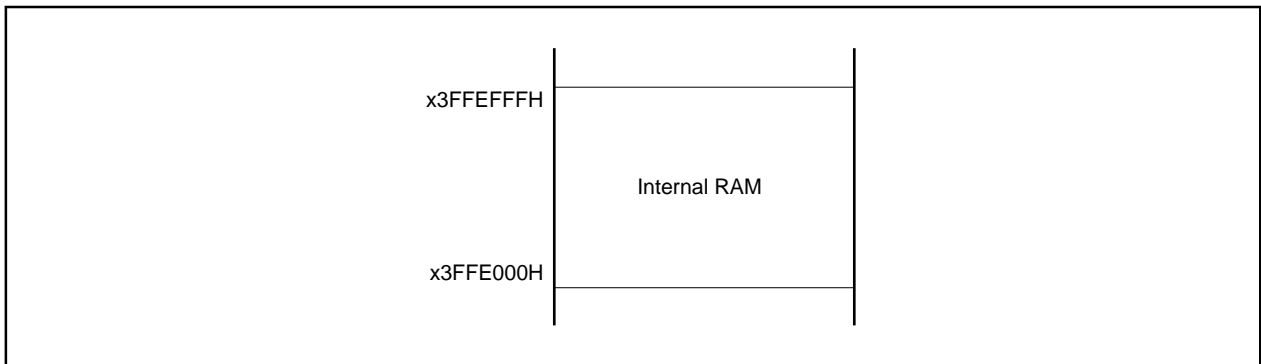
Start Address of Interrupt/Exception Table	Interrupt/Exception Source
00000000H	RESET
00000010H	NMI
00000040H	TRAP0n (n = 0 to FH)
00000050H	TRAP1n (n = 0 to FH)
00000060H	ILGOP
00000080H	INTOV10
00000090H	INTOV11
000000A0H	INTOV12
000000B0H	INTOV13
00000100H	INTP100/INTCC100
00000110H	INTP101/INTCC101
00000120H	INTP102/INTCC102
00000130H	INTP103/INTCC103
00000140H	INTP110/INTCC110
00000150H	INTP111/INTCC111
00000160H	INTP112/INTCC112
00000170H	INTP113/INTCC113
00000180H	INTCC120
00000190H	INTCC121
000001A0H	INTCC122
000001B0H	INTCC123
000001C0H	INTP130/INTCC130
000001D0H	INTCC131
000001E0H	INTCC132
000001F0H	INTCC133
00000280H	INTCM40
00000290H	INTCM41
000002A0H	INTDMA0
000002B0H	INTDMA1
000002C0H	INTDMA2
000002D0H	INTDMA3
00000300H	INTCSI0
00000310H	INTSER0
00000320H	INTSR0
00000330H	INTST0
00000340H	INTCSI1

**Table 3-3. Interrupt/Exception Table (2/2)**

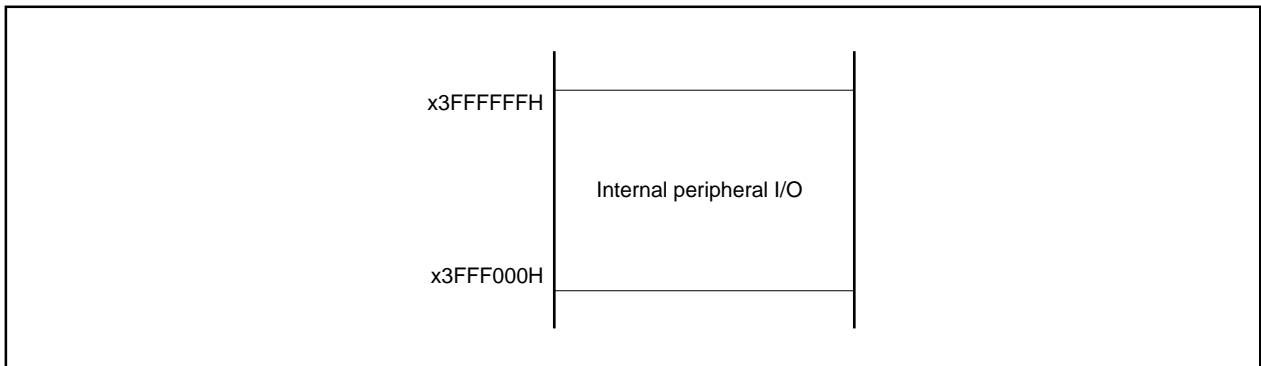
Start Address of Interrupt/Exception Table	Interrupt/Exception Source
00000350H	INTSER1
00000360H	INTSR1
00000370H	INTST1
00000400H	INTAD

**(2) Internal RAM area**

4 KB of memory, addresses 3FFE000H to 3FFFFFFH, is provided as a physical internal RAM area.

**(3) Internal peripheral I/O area**

4 KB of memory, addresses 3FFF000H to 3FFFFFFH, is provided as an internal peripheral I/O area.



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

- Cautions**
1. The least significant bit of an address is not decoded. If byte access is executed in the register at an odd address ( $2n + 1$ ), the register at the even address ( $2n$ ) will be accessed because of the hardware specification.
  2. In the V850E/MS2, no registers exist which are capable of word access, but if word access is executed in the register, for the word area, disregarding the bottom 2 bits of the address, halfword access is performed twice in the order of lower, then higher.
  3. For registers in which byte access is possible, if halfword access is executed, the higher 8 bits become non-specific during the read operation, and the lower 8 bits of data are written to the register during the write operation.
  4. 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 following areas can be used as external memory area. However, the reserved area from x1000000H to x2FFFFFFH is excluded.

- x0000000H to x3FFDFFFH

Access to the external memory area uses the chip select signal assigned to each memory block (refer to **4.4 Bus Cycle Type Control Function**).

Note that the internal RAM and internal peripheral I/O areas cannot be accessed as external memory areas.



### 3.4.6 External expansion mode

The V850E/MS2 allows external devices to be connected to the external memory space by using the pins of ports 4, 5, 6, A, and B. Setting the external expansion mode is carried out by selecting each pin of ports 4, 5, 6, A, and B in the control mode by means of the MM register.

Note that the status at reset time differs as shown below in accordance with the operating mode specification set by pins MODE0 and MODE2 (refer to **3.3 Operation Modes** for details of the operation modes).

#### (1) Status at reset time in each operation mode

##### (a) In the case of ROM-less mode 0

At reset time, each pin of ports 4, 5, 6, A, and B enters the control mode, so the external expansion mode is set without changing the MM register (the external data bus width is 16 bits).

##### (b) In the case of ROM-less mode 1

At reset time, each pin of ports 4, 5, 6, A, and B enters the control mode, so the external expansion mode is set without changing the setting of the MM register (the external data bus width is 8 bits).

#### (2) Memory expansion mode register (MM)

This register sets the mode of each pin of ports 4, 5, 6, A, and B. In the external expansion mode, an external device can be connected to an external memory area of up to 22 MB. However, an external device cannot be connected to the internal RAM area and internal peripheral I/O area (even if connected physically, it does not become an access target.).

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

	7	6	5	4	3	2	1	0		
MM	0	0	0	0	MM3	MM2	MM1	MM0	Address FFFFFF04CH	After reset Note

**Note** When in ROM-less mode 0: 07H When in ROM-less mode 1: 0FH

Bit Position	Bit Name	Function																																																																																																												
3 to 0	MM3 to MM0	Memory Expansion Mode Set the function of ports 4, 5, 6, A, and B.																																																																																																												
		MM3	MM2	MM1	MM0	Port 4	Port 5	Port A	Port B		Port 6				0	0	0	0	P40 to P47	P50 to P57	PA0 to PA7	PB0 to PB3	PB4, PB5, A12, A13	PB6, PB7, A14, A15	P60, P61, A16, A17	P62, P63, A18, A19	P64, P65, A20, A21	P66, P67, A22, A23	0	0	0	1	D0 to D7	D8 to D15	A0 to A7	A8 to A11	PB0 to PB3	PB4, PB5, A12, A13	PB6, PB7, A14, A15	P60, P61, A16, A17	P62, P63, A18, A19	P64, P65, A20, A21	P66, P67, A22, A23	0	0	1	0	0	0	1	1	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1	1	0	0	0	P40 to P47	P50 to P57	PA0 to PA7	PB0 to PB3	PB4, PB5, A12, A13	PB6, PB7, A14, A15	P60, P61, A16, A17	P62, P63, A18, A19	P64, P65, A20, A21	P66, P67, A22, A23	1	0	0	1	D0 to D7	1	0	1	0	1	0	1	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	1	1
		MM3	MM2	MM1	MM0	Port 4	Port 5	Port A	Port B		Port 6																																																																																																			
		0	0	0	0	P40 to P47	P50 to P57	PA0 to PA7	PB0 to PB3	PB4, PB5, A12, A13	PB6, PB7, A14, A15	P60, P61, A16, A17	P62, P63, A18, A19	P64, P65, A20, A21	P66, P67, A22, A23																																																																																															
		0	0	0	1	D0 to D7	D8 to D15	A0 to A7	A8 to A11	PB0 to PB3	PB4, PB5, A12, A13	PB6, PB7, A14, A15	P60, P61, A16, A17	P62, P63, A18, A19	P64, P65, A20, A21	P66, P67, A22, A23																																																																																														
		0	0	1	0																																																																																																									
		0	0	1	1																																																																																																									
		0	1	0	0																																																																																																									
		0	1	0	1																																																																																																									
		0	1	1	0																																																																																																									
		0	1	1	1																																																																																																									
		1	0	0	0	P40 to P47	P50 to P57	PA0 to PA7	PB0 to PB3	PB4, PB5, A12, A13	PB6, PB7, A14, A15	P60, P61, A16, A17	P62, P63, A18, A19	P64, P65, A20, A21	P66, P67, A22, A23																																																																																															
		1	0	0	1	D0 to D7																																																																																																								
		1	0	1	0																																																																																																									
		1	0	1	1																																																																																																									
		1	1	0	0																																																																																																									
		1	1	0	1																																																																																																									
		1	1	1	0																																																																																																									
		1	1	1	1																																																																																																									

**Caution** Write to the MM register after reset, and then do not change the set value. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the MM register is complete. However, it is possible to access an external memory area whose initialization is complete.

**Remarks** 1. For details of the operation of each port's pins, refer to **2.3 Description of Pin Functions**.  
2. The function of each port at system reset time is as shown below.

Operation Mode	MM Register	Port 4	Port 5	Port A	Port B	Port 6
ROM-less mode 0	07H	D0 to D7	D8 to D15	A0 to A7	A8 to A15	A16 to A23
ROM-less mode 1	0FH		P50 to P57			

### 3.4.7 Recommended use of address space

The architecture of the V850E/MS2 requires that a register that serves as a pointer be secured for address generation when accessing the operand data in the data space. An instruction can be used to directly access operand data at the address in this pointer register  $\pm 32$  KB. However, the general-purpose registers that can be used as a pointer register are limited. Therefore, by minimizing the deterioration of address calculation performance when changing the pointer value, the number of usable general-purpose registers for handling variables is maximized, and the program size can be saved.

To enhance the efficiency of using the pointer in connection with the memory map of the V850E/MS2, the following points are recommended:

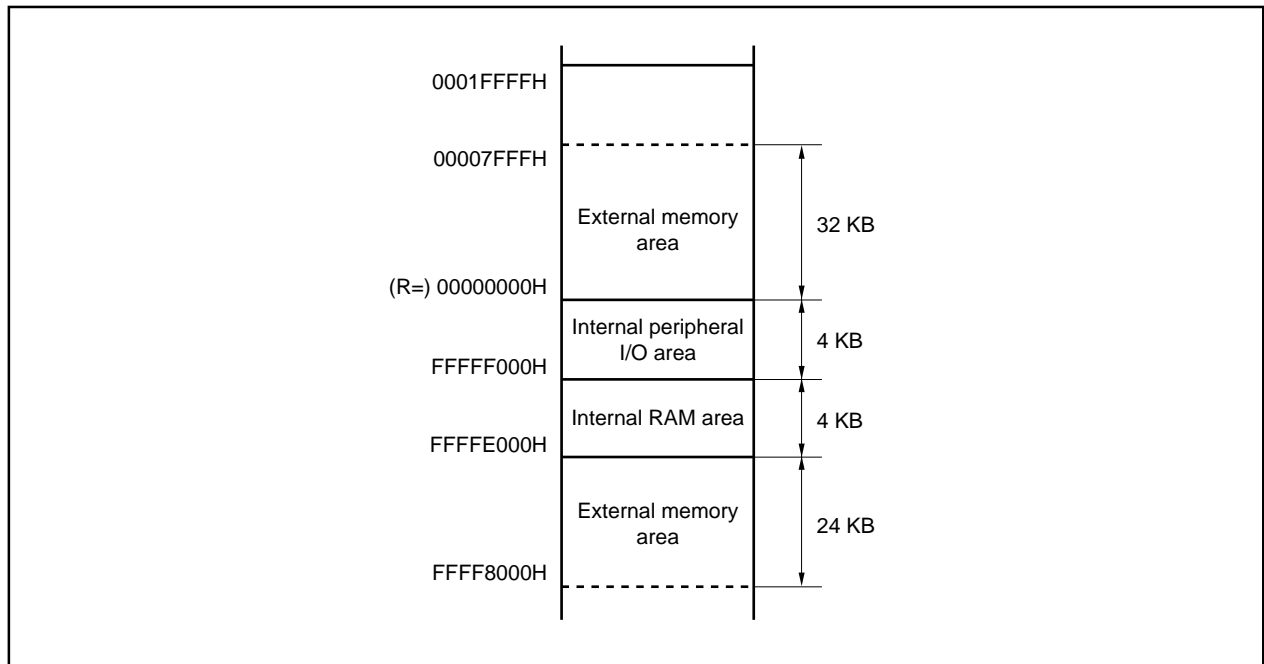
#### (1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed to 0, and only the lower 26 bits are valid. Therefore, a contiguous 64 MB space, starting from address 00000000H, unconditionally corresponds to the memory map of the program space.

#### (2) Data space

For the efficient use of resources using the wrap-around feature of the data space, the continuous 16 MB address spaces 00000000H to 00FFFFFFH and FF000000H to FFFFFFFFH of the 4 GB CPU are used as the data space. With the V850E/MS2, the 64 MB physical address space is seen as 64 images in the 4 GB CPU address space. The highest bit (bit 25) of this 26-bit address is assigned as address sign-extended to 32 bits.

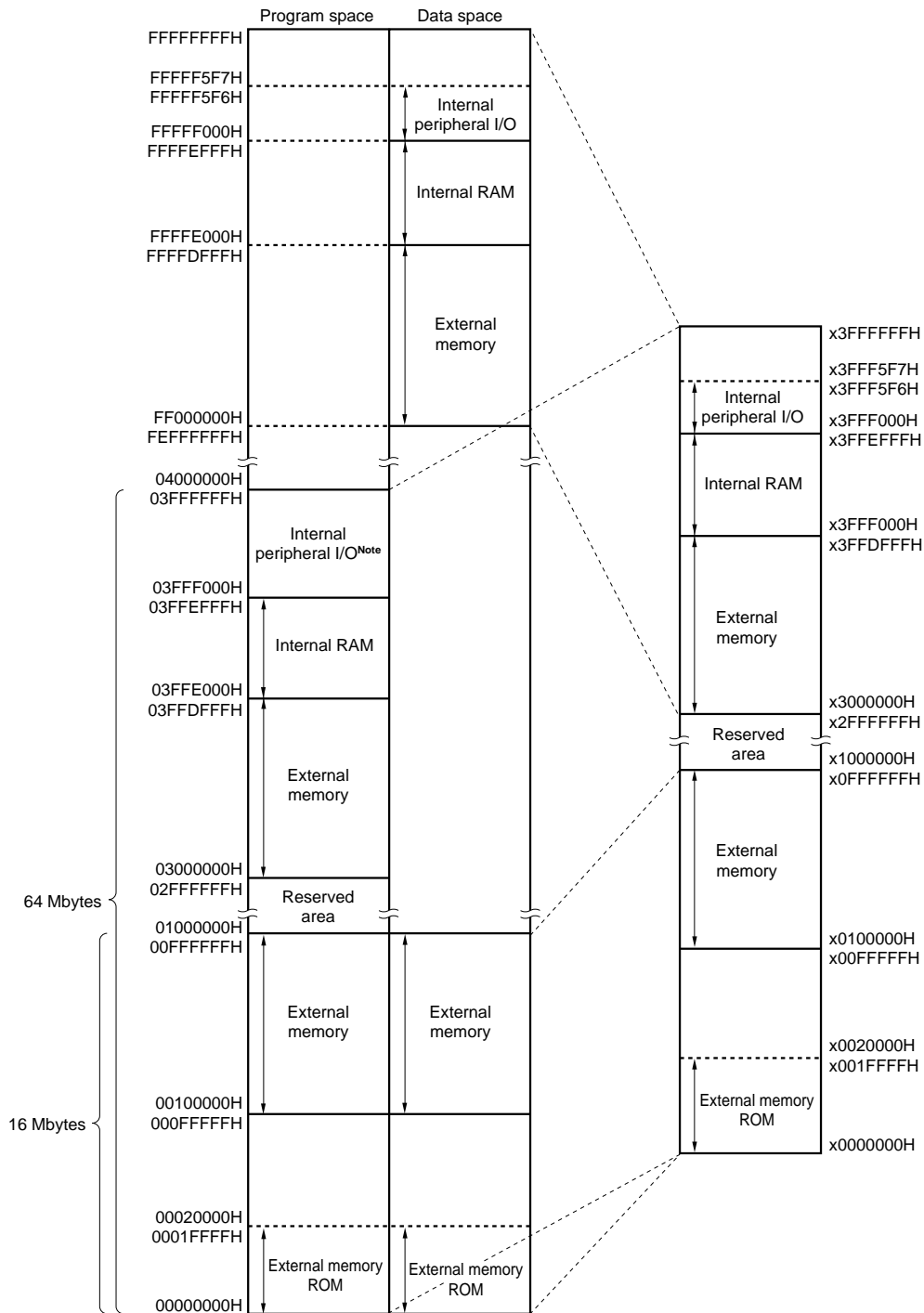
#### Example Application of wrap-around



When R = r0 (zero register) is specified for the LD/ST disp16 [R] instruction, an addressing range of 00000000H  $\pm 32$  KB can be referenced with the sign-extended, 16-bit displacement value. By mapping the external memory in the 24 KB area in the figure, all resources including internal hardware can be accessed with one pointer.

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

Figure 3-6. Recommended Memory Map



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

**Remark** The arrows indicate the recommended area.

## 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	
FFFFF000H	Port 0	P0	R/W	○	○		Undefined
FFFFF002H	Port 1	P1		○	○		
FFFFF004H	Port 2	P2		○	○		
FFFFF006H	Port 3	P3		○	○		
FFFFF008H	Port 4	P4		○	○		
FFFFF00AH	Port 5	P5		○	○		
FFFFF00CH	Port 6	P6		○	○		
FFFFF00EH	Port 7	P7	R	○	○		
FFFFF010H	Port 8	P8	R/W	○	○		
FFFFF012H	Port 9	P9		○	○		
FFFFF014H	Port 10	P10		○	○		
FFFFF01CH	Port A	PA		○	○		
FFFFF01EH	Port B	PB		○	○		
FFFFF020H	Port 0 mode register	PM0		○	○		FFH
FFFFF022H	Port 1 mode register	PM1		○	○		
FFFFF024H	Port 2 mode register	PM2		○	○		
FFFFF026H	Port 3 mode register	PM3		○	○		
FFFFF028H	Port 4 mode register	PM4		○	○		
FFFFF02AH	Port 5 mode register	PM5		○	○		
FFFFF02CH	Port 6 mode register	PM6		○	○		
FFFFF030H	Port 8 mode register	PM8		○	○		
FFFFF032H	Port 9 mode register	PM9		○	○		
FFFFF034H	Port 10 mode register	PM10		○	○		
FFFFF03CH	Port A mode register	PMA		○	○		
FFFFF03EH	Port B mode register	PMB		○	○		
FFFFF040H	Port 0 mode control register	PMC0		○	○		00H
FFFFF042H	Port 1 mode control register	PMC1		○	○		
FFFFF044H	Port 2 mode control register	PMC2		○	○		01H
FFFFF046H	Port 3 mode control register	PMC3		○	○		00H
FFFFF04CH	Memory expansion mode register	MM		○	○		07H/0FH
FFFFF050H	Port 8 mode control register	PMC8		○	○		FFH
FFFFF052H	Port 9 mode control register	PMC9		○	○		
FFFFF054H	Port 10 mode control register	PMC10		○	○		00H
FFFFF060H	Data wait control register 1	DWC1				○	FFFFH
FFFFF062H	Bus cycle control register	BCC				○	5555H
FFFFF064H	Bus cycle type control register	BCT				○	0000H

(2/6)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF066H	Bus size configuration register	BSC	R/W			○	5555H/ 0000H
FFFFF06AH	Data wait control register 2	DWC2		○	○		FFH
FFFFF06CH	Fly-by transfer data wait control register	FDW		○	○		00H
FFFFF070H	Power save control register	PSC		○	○		
FFFFF072H	Clock control register	CKC		○	○		
FFFFF078H	System status register	SYS		○	○		0000000×B
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
FFFFF0C0H	Asynchronous serial interface mode register 00	ASIM00		○	○		80H
FFFFF0C2H	Asynchronous serial interface mode register 01	ASIM01		○	○		00H
FFFFF0C4H	Asynchronous serial interface status register 0	ASIS0	R	○	○		
FFFFF0C8H	Receive buffer 0 (9 bits)	RXB0				○	Undefined
FFFFF0CAH	Receive buffer 0L (lower 8 bits)	RXB0L	W	○	○		
FFFFF0CCH	Transmit shift register 0 (9 bits)	TXS0				○	
FFFFF0CEH	Transmit shift register 0L (lower 8 bits)	TXS0L	R/W		○		80H
FFFFF0D0H	Asynchronous serial interface mode register 10	ASIM10		○	○		
FFFFF0D2H	Asynchronous serial interface mode register 11	ASIM11	R	○	○		00H
FFFFF0D4H	Asynchronous serial interface status register 1	ASIS1		○	○		
FFFFF0D8H	Receive buffer 1 (9 bits)	RXB1	W			○	Undefined
FFFFF0DAH	Receive buffer 1L (lower 8 bits)	RXB1L		○	○		
FFFFF0DCH	Transmit shift register 1 (9 bits)	TXS1	R/W			○	
FFFFF0DEH	Transmit shift register 1L (lower 8 bits)	TXS1L			○		47H
FFFFF100H	Interrupt control register	OVIC10	R/W	○	○		
FFFFF102H	Interrupt control register	OVIC11		○	○		
FFFFF104H	Interrupt control register	OVIC12		○	○		
FFFFF106H	Interrupt control register	OVIC13		○	○		
FFFFF10CH	Interrupt control register	CMIC40		○	○		
FFFFF10EH	Interrupt control register	CMIC41		○	○		
FFFFF110H	Interrupt control register	P10IC0		○	○		
FFFFF112H	Interrupt control register	P10IC1		○	○		

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF114H	Interrupt control register	P10IC2	R/W	○	○		47H
FFFFF116H	Interrupt control register	P10IC3		○	○		
FFFFF118H	Interrupt control register	P11IC0		○	○		
FFFFF11AH	Interrupt control register	P11IC1		○	○		
FFFFF11CH	Interrupt control register	P11IC2		○	○		
FFFFF11EH	Interrupt control register	P11IC3		○	○		
FFFFF120H	Interrupt control register	P12IC0		○	○		
FFFFF122H	Interrupt control register	P12IC1		○	○		
FFFFF124H	Interrupt control register	P12IC2		○	○		
FFFFF126H	Interrupt control register	P12IC3		○	○		
FFFFF128H	Interrupt control register	P13IC0		○	○		
FFFFF12AH	Interrupt control register	P13IC1		○	○		
FFFFF12CH	Interrupt control register	P13IC2		○	○		
FFFFF12EH	Interrupt control register	P13IC3		○	○		
FFFFF140H	Interrupt control register	DMAIC0		○	○		
FFFFF142H	Interrupt control register	DMAIC1		○	○		
FFFFF144H	Interrupt control register	DMAIC2		○	○		
FFFFF146H	Interrupt control register	DMAIC3		○	○		
FFFFF148H	Interrupt control register	CSIC0		○	○		
FFFFF14AH	Interrupt control register	CSIC1		○	○		
FFFFF150H	Interrupt control register	SEIC0		○	○		
FFFFF152H	Interrupt control register	SRIC0		○	○		
FFFFF154H	Interrupt control register	STIC0		○	○		
FFFFF156H	Interrupt control register	SEIC1		○	○		
FFFFF158H	Interrupt control register	SRIC1		○	○		
FFFFF15AH	Interrupt control register	STIC1		○	○		
FFFFF15CH	Interrupt control register	ADIC		○	○		
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		○	○		
FFFFF188H	External interrupt mode register 4	INTM4		○	○		
FFFFF1A0H	DMA source address register 0H	DSA0H				○	Undefined
FFFFF1A2H	DMA source address register 0L	DSA0L				○	
FFFFF1A4H	DMA destination address register 0H	DDA0H				○	
FFFFF1A6H	DMA destination address register 0L	DDA0L				○	

(4/6)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF1A8H	DMA source address register 1H	DSA1H	R/W			○	Undefined
FFFFF1AAH	DMA source address register 1L	DSA1L				○	
FFFFF1ACH	DMA destination address register 1H	DDA1H				○	
FFFFF1AEH	DMA destination address register 1L	DDA1L				○	
FFFFF1B0H	DMA source address register 2H	DSA2H				○	
FFFFF1B2H	DMA source address register 2L	DSA2L				○	
FFFFF1B4H	DMA destination address register 2H	DDA2H				○	
FFFFF1B6H	DMA destination address register 2L	DDA2L				○	
FFFFF1B8H	DMA source address register 3H	DSA3H				○	
FFFFF1BAH	DMA source address register 3L	DSA3L				○	
FFFFF1BCH	DMA destination address register 3H	DDA3H				○	
FFFFF1BEH	DMA destination address register 3L	DDA3L				○	
FFFFF1E0H	DMA byte count register 0	DBC0				○	
FFFFF1E2H	DMA byte count register 1	DBC1				○	
FFFFF1E4H	DMA byte count register 2	DBC2				○	
FFFFF1E6H	DMA byte count register 3	DBC3				○	
FFFFF1F0H	DMA addressing control register 0	DADC0	R			○	0000H
FFFFF1F2H	DMA addressing control register 1	DADC1				○	
FFFFF1F4H	DMA addressing control register 2	DADC2				○	
FFFFF1F6H	DMA addressing control register 3	DADC3				○	
FFFFF200H	DRAM configuration register 0	DRC0				○	3FC1H
FFFFF202H	DRAM configuration register 1	DRC1				○	
FFFFF204H	DRAM configuration register 2	DRC2				○	
FFFFF206H	DRAM configuration register 3	DRC3				○	
FFFFF210H	Refresh control register 0	RFC0				○	0000H
FFFFF212H	Refresh control register 1	RFC1				○	
FFFFF214H	Refresh control register 2	RFC2				○	
FFFFF216H	Refresh control register 3	RFC3				○	
FFFFF218H	Refresh wait control register	RWC		○	○		00H
FFFFF220H	DRAM type configuration register	DTC				○	0000H
FFFFF224H	Page-ROM configuration register	PRC		○	○		E0H
FFFFF230H	Timer overflow status register	TOVS		○	○		00H
FFFFF240H	Timer unit mode register 10	TUM10	R/W			○	0000H
FFFFF242H	Timer control register 10	TMC10		○	○		00H
FFFFF244H	Timer output control register 10	TOC10		○	○		
FFFFF250H	Timer 10	TM10	R			○	0000H
FFFFF252H	Capture/compare register 100	CC100	R/W			○	Undefined



(5/6)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF254H	Capture/compare register 101	CC101	R/W			○	Undefined
FFFFF256H	Capture/compare register 102	CC102				○	
FFFFF258H	Capture/compare register 103	CC103				○	
FFFFF260H	Timer unit mode register 11	TUM11				○	0000H
FFFFF262H	Timer control register 11	TMC11		○	○		00H
FFFFF264H	Timer output control register 11	TOC11		○	○		
FFFFF270H	Timer 11	TM11	R			○	0000H
FFFFF272H	Capture/compare register 110	CC110	R/W			○	Undefined
FFFFF274H	Capture/compare register 111	CC111				○	
FFFFF276H	Capture/compare register 112	CC112				○	
FFFFF278H	Capture/compare register 113	CC113				○	0000H
FFFFF280H	Timer unit mode register 12	TUM12				○	
FFFFF282H	Timer control register 12	TMC12		○	○		00H
FFFFF284H	Timer output control register 12	TOC12		○	○		
FFFFF290H	Timer 12	TM12	R			○	0000H
FFFFF292H	Capture/compare register 120	CC120	R/W			○	Undefined
FFFFF294H	Capture/compare register 121	CC121				○	
FFFFF296H	Capture/compare register 122	CC122				○	
FFFFF298H	Capture/compare register 123	CC123				○	0000H
FFFFF2A0H	Timer unit mode register 13	TUM13				○	
FFFFF2A2H	Timer control register 13	TMC13		○	○		00H
FFFFF2B0H	Timer 13	TM13	R			○	Undefined
FFFFF2B2H	Capture/compare register 130	CC130	R/W			○	
FFFFF2B4H	Capture/compare register 131	CC131				○	
FFFFF2B6H	Capture/compare register 132	CC132				○	
FFFFF2B8H	Capture/compare register 133	CC133				○	00H
FFFFF342H	Timer control register 40	TMC40		○	○		
FFFFF346H	Timer control register 41	TMC41		○	○		
FFFFF350H	Timer 40	TM40	R			○	0000H
FFFFF352H	Compare register 40	CM40	R/W			○	Undefined
FFFFF354H	Timer 41	TM41	R			○	0000H
FFFFF356H	Compare register 41	CM41	R/W			○	Undefined
FFFFF380H	A/D converter mode register 0	ADM0		○	○		00H
FFFFF382H	A/D converter mode register 1	ADM1		○	○		07H
FFFFF390H	A/D conversion result register 0	ADCR0	R			○	Undefined
FFFFF392H	A/D conversion result register 0H	ADCR0H		○	○		
FFFFF394H	A/D conversion result register 1	ADCR1				○	

(6/6)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF396H	A/D conversion result register 1H	ADCR1H	R	○	○		Undefined
FFFFF398H	A/D conversion result register 2	ADCR2				○	
FFFFF39AH	A/D conversion result register 2H	ADCR2H		○	○		
FFFFF39CH	A/D conversion result register 3	ADCR3				○	
FFFFF39EH	A/D conversion result register 3H	ADCR3H		○	○		
FFFFF41AH	Port X	PX	R/W	○	○		
FFFFF43AH	Port X mode register	PMX	W		○		FFH
FFFFF45AH	Port X mode control register	PMCX			○		E0H
FFFFF580H	Port/control select register 0	PCS0	R/W	○	○		00H
FFFFF582H	Port/control select register 1	PCS1		○	○		
FFFFF590H	Port/control select register 8	PCS8		○	○		
FFFFF5D0H	DMA disable status register	DDIS	R	○	○		
FFFFF5D2H	DMA restart register	DRST	R/W	○	○		
FFFFF5E0H	DMA trigger factor register 0	DTFR0		○	○		
FFFFF5E2H	DMA trigger factor register 1	DTFR1		○	○		
FFFFF5E4H	DMA trigger factor register 2	DTFR2		○	○		
FFFFF5E6H	DMA trigger factor register 3	DTFR3		○	○		
FFFFF5F0H	DMA channel control register 0	DCHC0		○	○		
FFFFF5F2H	DMA channel control register 1	DCHC1		○	○		
FFFFF5F4H	DMA channel control register 2	DCHC2		○	○		
FFFFF5F6H	DMA channel control register 3	DCHC3		○	○		

### 3.4.9 Specific registers

Specific registers are registers that are protected from being written with illegal data due to erroneous program execution, etc. The write access of these specific registers is executed in a specific sequence, and if abnormal store operations occur, the system status register (SYS) is notified. The V850E/MS2 has two specific registers, clock control register (CKC) and the power save control register (PSC). For details of the CKC register, refer to 8.3.3 and for details of the PSC register, refer to 8.5.2.

The access sequence to the specific registers is shown below.

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

- <1> Provide data in the desired general-purpose register to be set in the specific register.
- <2> Write the general-purpose register prepared in <1> in the command register (PRCMD).
- <3> Write to the specific register using the general-purpose register prepared in <1> (do this using the following instructions).
  - Store instruction (ST/SST instruction)
  - Bit operation instruction (SET1/CLR1/NOT1 instruction)
- <4> If the system moves to the IDLE or software STOP mode, insert a NOP instruction (1 instruction).

**Example**

```

<1> MOV    0x04, r10
<2> ST.B   r10, PRCMD [r0]
<3> ST.B   r10, PSC [r0]
<4> NOP

```

No special sequence is required when reading the specific registers.

**Caution** Do not write to the PRCMD register or to a specific register by DMA transfer.

- Remarks**
1. A store instruction to a command register will not be received with an interrupt.  
This presupposes that this is done with the continuous store instructions in <1> and <2> above in the program. If another instruction is placed between <1> and <2>, when an interrupt is received by that instruction, the above sequence may not be established, and cause a malfunction, so caution is necessary.
  2. The data written in the PRCMD register is dummy data, but use the same general-purpose register for writing to the PRCMD register (<2> in the example above) as was used in setting data in the specific register (<3> in the example above). Addressing is the same in the case where a general-purpose register is used.
  3. It is necessary to insert 1 or more NOP instructions just after a store instruction to the PSC register for setting it in the software STOP or IDLE mode. When releasing each power save mode by interrupt, or when resetting after executing interrupt servicing, start execution from the next instruction without executing the instruction just after the store instruction.

[Example of Description]

```

ST reg_code, PRCMD ; PRCMD write
                    (reg_code: Registration code)

ST data, PSC       ; Setting of the PSC register
NOP               ; Dummy instruction (1 instruction)
(next instruction) ; Execution routine after releasing the software
                  STOP/IDLE mode
                  :
                  :
```

The case where bit operation instructions are used in the PSC register settings is the same.

#### (1) Command register (PRCMD)

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

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

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

	7	6	5	4	3	2	1	0		
PRCMD	REG7	REG6	REG5	REG4	REG3	REG2	REG1	REG0	Address FFFFFF170H	After reset Undefined

Bit Position	Bit Name	Function						
7 to 0	REG7 to REG0	Registration Code						
		<table><tr><td>Specific Register</td><td>Registration Code</td></tr><tr><td>CKC</td><td>Any 8-bit data</td></tr><tr><td>PSC</td><td>Any 8-bit data</td></tr></table>	Specific Register	Registration Code	CKC	Any 8-bit data	PSC	Any 8-bit data
		Specific Register	Registration Code					
		CKC	Any 8-bit data					
PSC	Any 8-bit data							

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

This register is assigned 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	LOCK	Address FFFFFF078H	After reset 0000000xB

Bit Position	Bit Name	Function
4	PRERR	Protection Error Flag This is a cumulative flag that shows that writing to a specific register was not done in the correct sequence and that a protection error occurred <sup>Note</sup> . 0: Protection error did not occur 1: Protection error occurred
0	LOCK	Lock Status Flag This is an exclusive read out flag. It shows that the PLL is in the locked state (for details, refer to <b>8.4 PLL Lockup</b> ). 0: Locked. 1: Unlocked.

**Note** 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 V850E/MS2 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/8-bit data bus sizing function
- 4-space chip select output function
- Wait function
  - Programmable wait function, capable of inserting up to 7 wait states for each memory block
  - External wait function via  $\overline{\text{WAIT}}$  pin
- Idle state insertion function
- Bus mastership arbitration function
- Bus hold function
- Capable of connecting to external devices via alternate function pins

### 4.2 Bus Control Pins

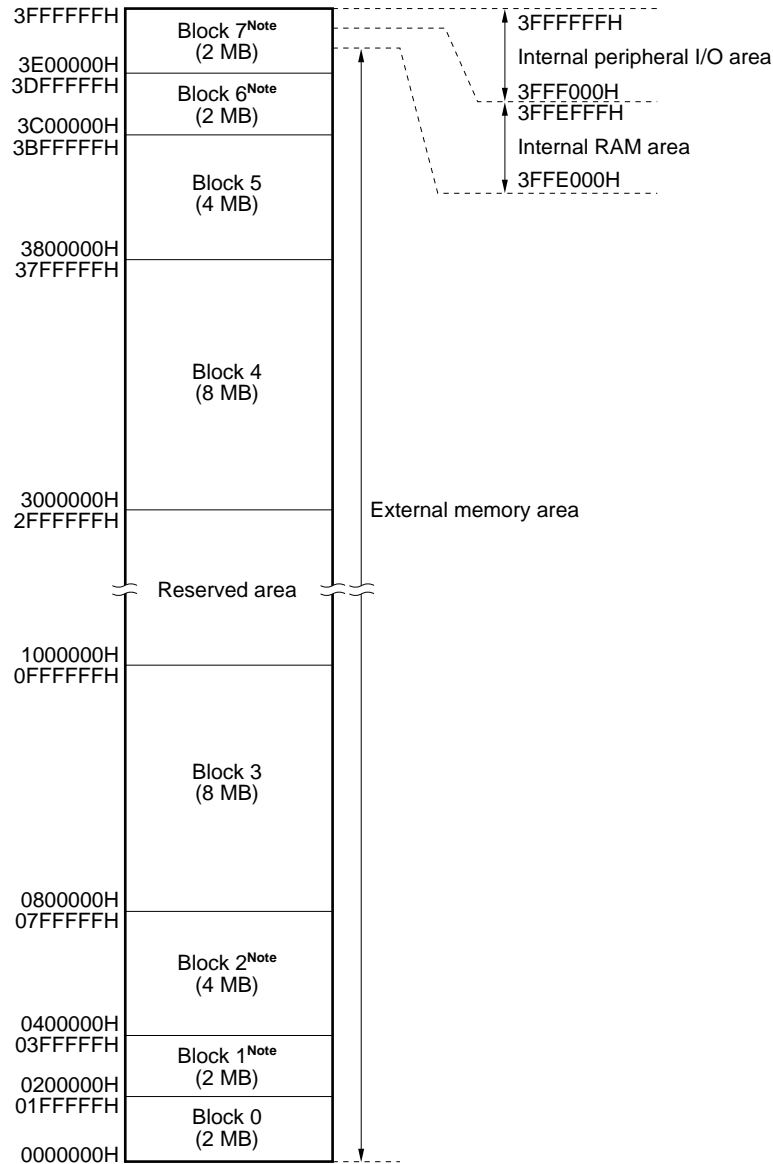
The following pins are used for connecting to external devices:

Bus Control Pin (Function When in the Control Mode)	Function When in the Port Mode	Register Which Performs Port/Control Mode Switching
Data bus (D0 to D7)	P40 to P47 (Port 4)	MM
Data bus (D8 to D15)	P50 to P57 (Port 5)	MM
Address bus (A0 to A7)	PA0 to PA7 (Port A)	MM
Address bus (A8 to A15)	PB0 to PB7 (Port B)	MM
Address bus (A16 to A23)	P60 to P67 (Port 6)	MM
Chip select ( $\overline{\text{CS0}}$ , $\overline{\text{CS3}}$ to $\overline{\text{CS5}}$ , $\overline{\text{RAS3}}$ to $\overline{\text{RAS5}}$ , $\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$ )	P80, P83 to P85 (Port 8)	PMC8
Read/write control ( $\overline{\text{LCAS}}$ , $\overline{\text{UCAS}}$ , $\overline{\text{LWR}}$ , $\overline{\text{UWR}}$ , $\overline{\text{RD}}$ , $\overline{\text{WE}}$ , $\overline{\text{OE}}$ )	P90 to P93, P95 (Port 9)	PMC9
Bus cycle start (BCYST)	P94 (Port 9)	PMC9
External wait control ( $\overline{\text{WAIT}}$ )	PX6 (Port X)	PMCX
Bus hold control ( $\overline{\text{HLD\text{AK}}}$ , $\overline{\text{HLD\text{RQ}}}$ )	P96, P97 (Port 9)	PMC9
Internal system clock (CLKOUT)	PX7 (Port X)	PMCX

**Remark** When the system is reset, each bus control pin becomes unconditionally valid (however, D8 to D15 are valid only in ROM-less mode 0). For details, refer to **3.4.6 External expansion mode**.

### 4.3 Memory Block Function

The 64 MB memory space is divided into memory blocks of 2 MB, 4 MB, and 8 MB units. The programmable wait function and bus cycle operation mode can be independently controlled for each individual memory block.



**Note** The chip select signal is not output. When using this area, it is necessary to decode an address signal and generate the chip select signal. The settings for the BCT, DWC1, DWC0, and BCC registers are also valid for this area.



## 4.4 Bus Cycle Type Control Function

In the V850E/MS2, the following external devices can be connected directly to each memory block.

- SRAM, external ROM, external I/O
- Page ROM
- DRAM

Connected external devices are specified by the bus cycle type configuration register (BCT).

### 4.4.1 Bus cycle type configuration register (BCT)

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

BCT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BT71	BT70	BT61	BT60	BT51	BT50	BT41	BT40	BT31	BT30	BT21	BT20	BT11	BT10	Note 1 BT01	BT00

Address  
FFFFF064H

After reset  
0000H

Memory block

7

6

5

4

3

2

1

0

Bit Position	Bit Name	Function															
15 to 0	BTn1, BTn0 (n = 7 to 0)	Bus Cycle Type Specifies the external device connected to memory block n. <table border="1" style="border-collapse: collapse; width: 100%; margin-top: 10px;"> <thead> <tr> <th style="width: 15%;">BTn1</th><th style="width: 15%;">BTn0</th><th style="width: 70%;">External Device Connected Directly to Memory Block n</th></tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td><td style="text-align: center;">0</td><td>SRAM, external ROM, external I/O</td></tr> <tr> <td style="text-align: center;">0</td><td style="text-align: center;">1</td><td>Page ROM</td></tr> <tr> <td style="text-align: center;">1</td><td style="text-align: center;">0</td><td>DRAM<sup>Note 2</sup></td></tr> <tr> <td style="text-align: center;">1</td><td style="text-align: center;">1</td><td>Setting prohibited</td></tr> </tbody> </table>	BTn1	BTn0	External Device Connected Directly to Memory Block n	0	0	SRAM, external ROM, external I/O	0	1	Page ROM	1	0	DRAM <sup>Note 2</sup>	1	1	Setting prohibited
BTn1	BTn0	External Device Connected Directly to Memory Block n															
0	0	SRAM, external ROM, external I/O															
0	1	Page ROM															
1	0	DRAM <sup>Note 2</sup>															
1	1	Setting prohibited															

**Notes**

1. Be sure to set bit BT01 to 0.
2. Using the DTC register, one DRAM access type setting can be selected out of 4 types for each memory block (refer to **5.3.5 DRAM type configuration register (DTC)**).

**Caution** Write to the BCT register after reset, and then do not change the set value. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the BCT register is complete. However, it is possible to access an external memory area whose initialization is complete.

The chip select signals ( $\overline{CS0}$ ,  $\overline{CS3/RAS3}$  to  $\overline{CS5/RAS5}$ ) are output as follows in correspondence with blocks 0 to 7.

Memory Block \ External Device	SRAM, External ROM, External I/O Page ROM	DRAM
Block 0	$\overline{CS0}$	—
Block 3	$\overline{CS3}$	$\overline{RAS3}$
Block 4	$\overline{CS4}$	$\overline{RAS4}$
Block 5	$\overline{CS5}$	$\overline{RAS5}$

## 4.5 Bus Access

### 4.5.1 Number of access clocks

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

Bus Cycle Configuration Resource (Bus Width)			Instruction Fetch		Operand Data Access	
			Normal Access	Burst Access	Normal Access	Burst Access
Internal RAM (32 bits)			1 or 2	—	1	—
Internal peripheral I/O (16 bits)			—	—	3 + n	—
External device	SRAM, external ROM, external I/O (16/8 bits)		2 + n	—	2 + n	—
		During DMA flyby transfer	—	—	2 + n	—
	Page ROM (16/8 bits)		2 + n	2 + n	2 + n	2 + n
	High-speed page DRAM (16/8 bits)		3 + n	2 + n	3 + n	2 + n
	During DMA flyby transfer	During read	—	—	3 + n	2 + n
		During write	—	—	3 + n	3 + n
	EDO DRAM (16/8 bits)		3 + n	1 + n	3 + n	1 + n
	During DMA flyby transfer	During read	—	—	3 + n	2 + n
		During write	—	—	3 + n	3 + n

- Remarks**
1. Unit: Clock/access
  2. n: Number of wait insertions

#### (1) Internal peripheral I/O interface

The contents of the access to internal peripheral I/O are not output to the external bus. Therefore, during instruction fetch access, internal peripheral I/O access can be performed in parallel.

Internal peripheral I/O access is basically 3-clock access. However, on some occasions, access to internal peripheral I/O registers with timer/counter functions also involves a wait.

Internal Peripheral I/O Register	Access	Waits	Clock Cycles
CC1n0 to CC1n3, TM1n (n = 0 to 5)	Read	1	4
	Write	0/1	3/4
CM40, CM41	Read	0	3
	Write	0/1	3/4
TM40, TM41	Read	0/1	3/4
	Write	0	3
Other	Read	0	3
	Write	0	3

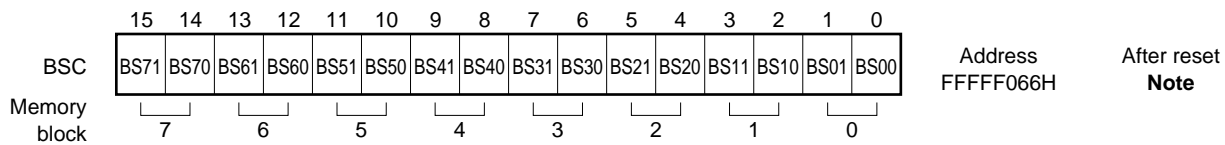
### 4.5.2 Bus sizing function

The V850E/MS2 is provided with a bus sizing function that is used to control the data bus width of each memory block.

The data bus width is specified by using the bus size configuration register (BSC).

#### (1) Bus size configuration register (BSC)

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



**Note** When in ROM-less mode 0: 5555H

When in ROM-less mode 1: 0000H

Bit Position	Bit Name	Function												
15 to 0	BSn1, BSn0 (n = 7 to 0)	Data Bus Width Sets the data bus width of memory block n. <table border="1" style="margin-top: 10px;"> <thead> <tr> <th>BSn1</th><th>BSn0</th><th>Data Bus Width of Memory Block n</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>8 bits</td></tr> <tr> <td>0</td><td>1</td><td>16 bits</td></tr> <tr> <td>1</td><td>Optional</td><td>RFU (Reserved)</td></tr> </tbody> </table>	BSn1	BSn0	Data Bus Width of Memory Block n	0	0	8 bits	0	1	16 bits	1	Optional	RFU (Reserved)
BSn1	BSn0	Data Bus Width of Memory Block n												
0	0	8 bits												
0	1	16 bits												
1	Optional	RFU (Reserved)												

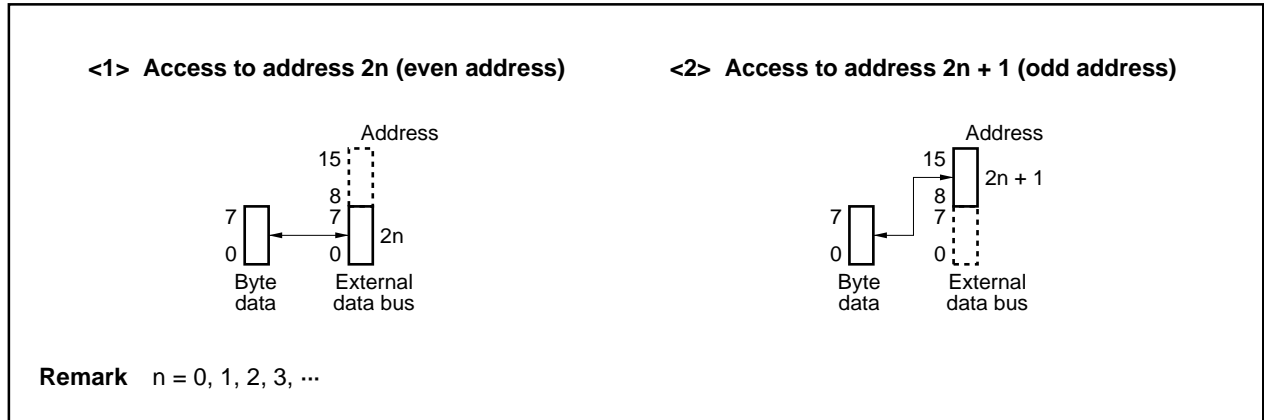
- Cautions**
1. Write to the BSC register after reset, and then do not change the set value. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the BSC register is complete. However, it is possible to access an external memory area whose initialization is complete.
  2. The in-circuit emulator (IE-703102-MC) for the V850E/MS2 does not support 8-bit width external ROM emulation.
  3. When 8-bit data bus width is selected, only the write signal  $\overline{\text{LWR}}$  becomes active,  $\overline{\text{UWR}}$  does not become active.

### 4.5.3 Bus width

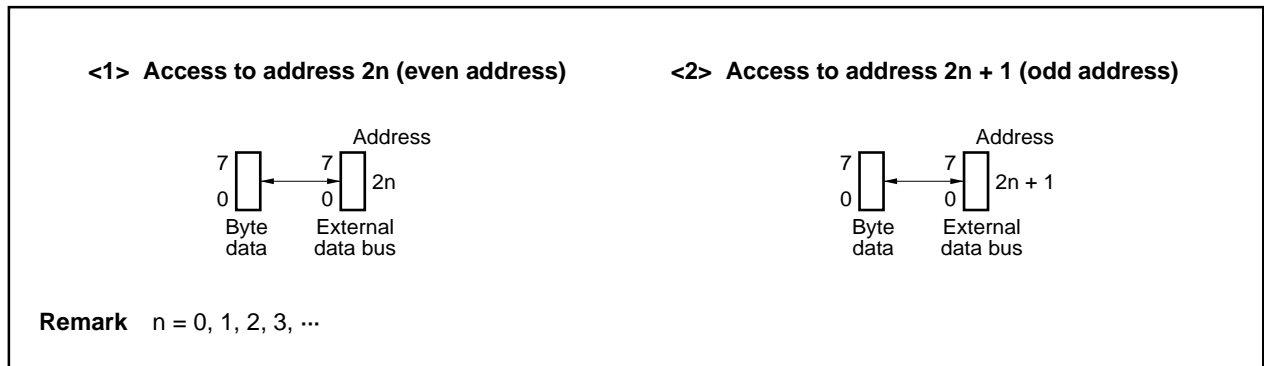
V850E/MS2 carries out peripheral I/O access and external memory access in 8, 16, or 32 bits. The following shows the operation for each access. All data is accessed in order from the lower side.

#### (1) Byte access (8 bits)

##### (a) When the data bus width is 16 bits



##### (b) When the data bus width is 8 bits

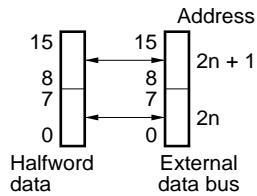


## (2) Halfword access (16 bits)

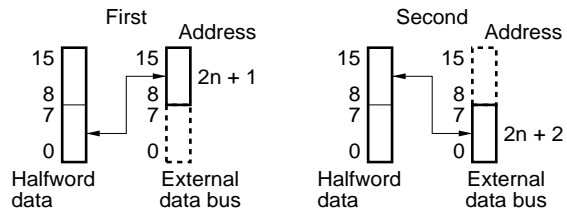
In halfword access to external memory, data is exchanged as is, or accessed in the order of lower byte, then higher byte.

### (a) When the data bus width is 16 bits

#### <1> Access to address $2n$ (even address)



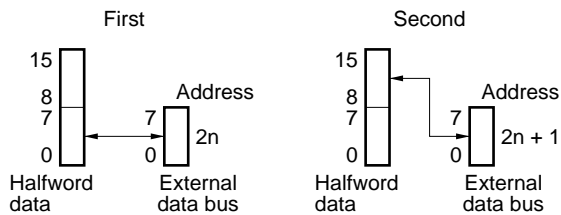
#### <2> Access to address $2n + 1$ (odd address)



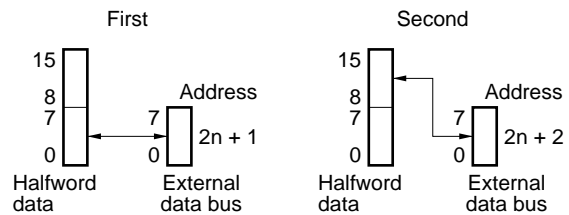
**Remark**  $n = 0, 1, 2, 3, \dots$

### (b) When the data bus width is 8 bits

#### <1> Access to address $2n$ (even address)



#### <2> Access to address $2n + 1$ (odd address)



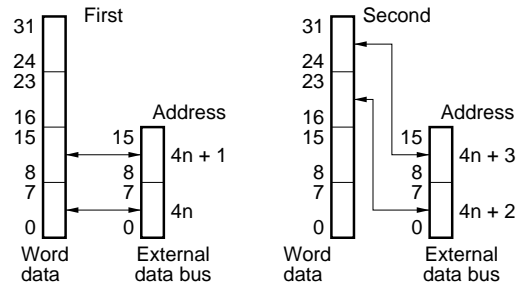
**Remark**  $n = 0, 1, 2, 3, \dots$

## (3) Word access (32 bits)

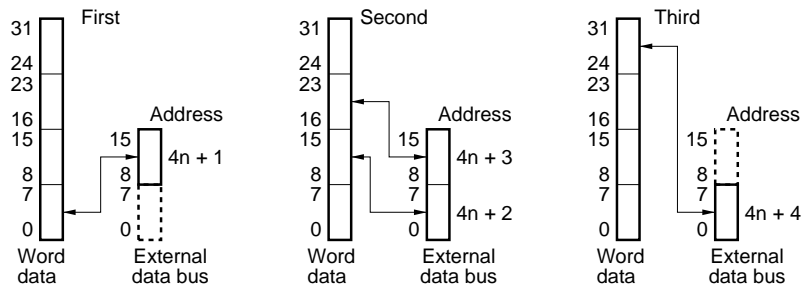
In word access to external memory, data is accessed in order from the lower halfword, then the higher halfword, or in order from the lowest byte to the highest byte.

(a) When the data bus width is 16 bits

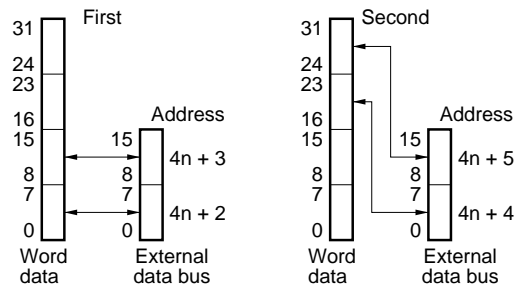
<1> Access to address  $4n$



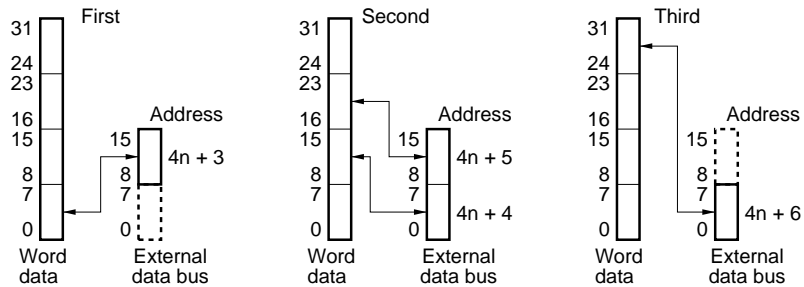
<2> Access to address  $4n + 1$



<3> Access to address  $4n + 2$



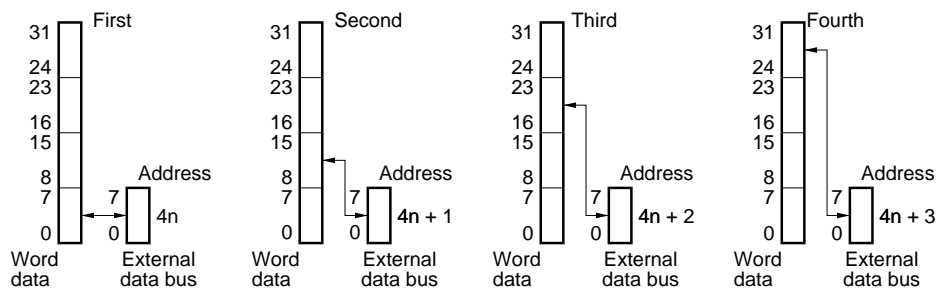
<4> Access to address  $4n + 3$



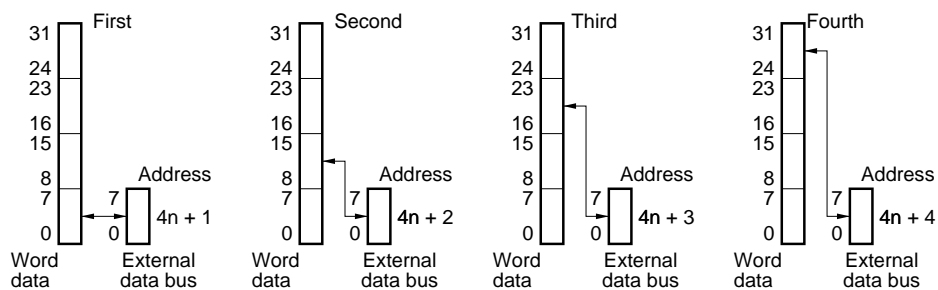
**Remark**  $n = 0, 1, 2, 3, \dots$

(b) When the data bus width is 8 bits

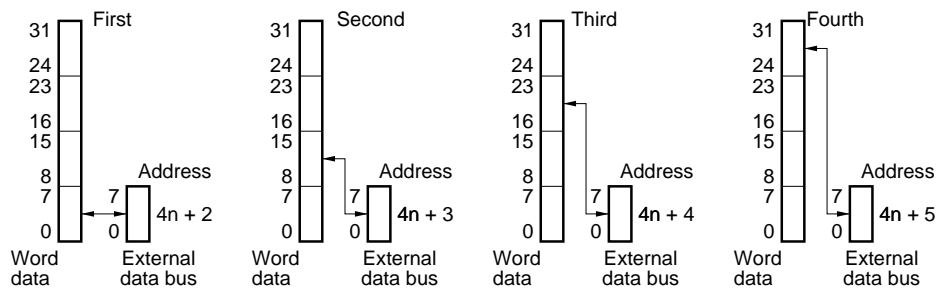
<1> Access to address  $4n$



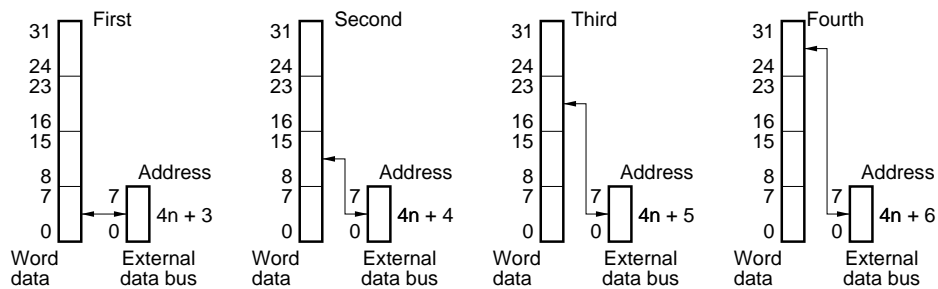
<2> Access to address  $4n + 1$



<3> Access to address  $4n + 2$



<4> Access to address  $4n + 3$



**Remark**  $n = 0, 1, 2, 3, \dots$



## 4.6 Wait Function

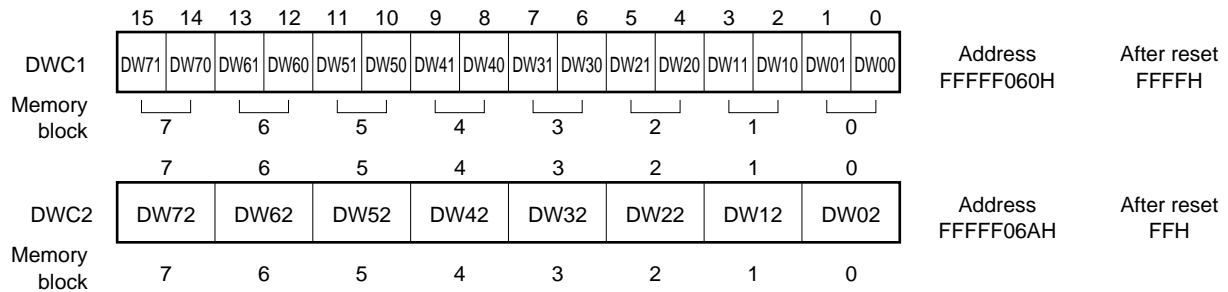
### 4.6.1 Programmable wait function

With the aim of realizing easy interfacing with low-speed memory or with I/Os, it is possible to insert up to 7 data wait states with respect to the starting bus cycle for each memory block.

The number of wait states can be set by data wait control registers 1 and 2 (DWC1, DWC2) and can be specified by program. Just after system reset, all blocks have 7 data wait states inserted.

#### (1) Data wait control registers 1, 2 (DWC1, DWC2)

It is possible to read/write the DWC1 register in 16-bit units and the DWC2 register in 8/1-bit units.



Register Name	Bit Position	Bit Name	Function																																				
DWC1	15 to 0	DWn1, DWn0 (n = 7 to 0)	Data Wait Specifies the number of wait states inserted in memory block n. Registers DWC1 and DWC2 are set in combination. <table border="1"> <tr> <th>DWn2</th><th>DWn1</th><th>DWn0</th><th>Number of Wait States Inserted in Memory Block n</th></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>2</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>3</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>4</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>5</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>6</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>7</td></tr> </table>	DWn2	DWn1	DWn0	Number of Wait States Inserted in Memory Block n	0	0	0	0	0	0	1	1	0	1	0	2	0	1	1	3	1	0	0	4	1	0	1	5	1	1	0	6	1	1	1	7
DWn2	DWn1	DWn0	Number of Wait States Inserted in Memory Block n																																				
0	0	0	0																																				
0	0	1	1																																				
0	1	0	2																																				
0	1	1	3																																				
1	0	0	4																																				
1	0	1	5																																				
1	1	0	6																																				
1	1	1	7																																				
DWC2	7 to 0	DWn2 (n = 7 to 0)																																					

- Cautions**
- The internal RAM area is not subject to programmable waits and ordinarily no wait access is carried out. Neither is the internal peripheral I/O area subject to programmable wait states, with wait control performed only by each peripheral function.
  - In the following cases, the settings of registers DWC1 and DWC2 are invalid (wait control is performed by each memory controller).
    - DRAM access
    - Page ROM on-page access
  - Write to the DWC1 and DWC2 registers after reset, and then do not change the set values. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the DWC1 and DWC2 registers is complete. However, it is possible to access an external memory area whose initialization is complete.

#### 4.6.2 External wait function

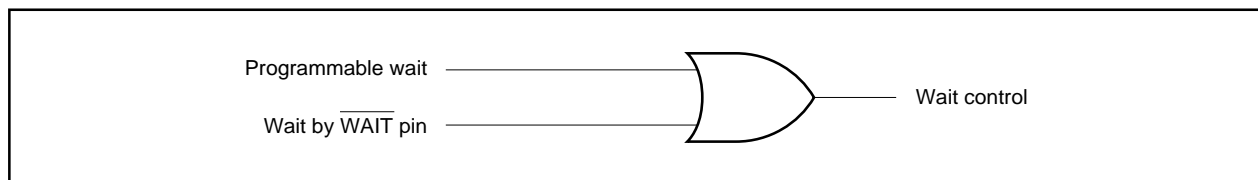
When an extremely slow device, I/O, or asynchronous system is connected, any number of wait states can be inserted in a bus cycle by the external wait pin ( $\overline{\text{WAIT}}$ ) to synchronize with the external device.

Just as with programmable waits, access to internal RAM and internal peripheral I/O areas cannot be controlled by external waits.

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 T1 and TW states of a bus cycle. If the setup/hold time in the sampling timing is not satisfied, a wait may or may not be inserted in the next state.

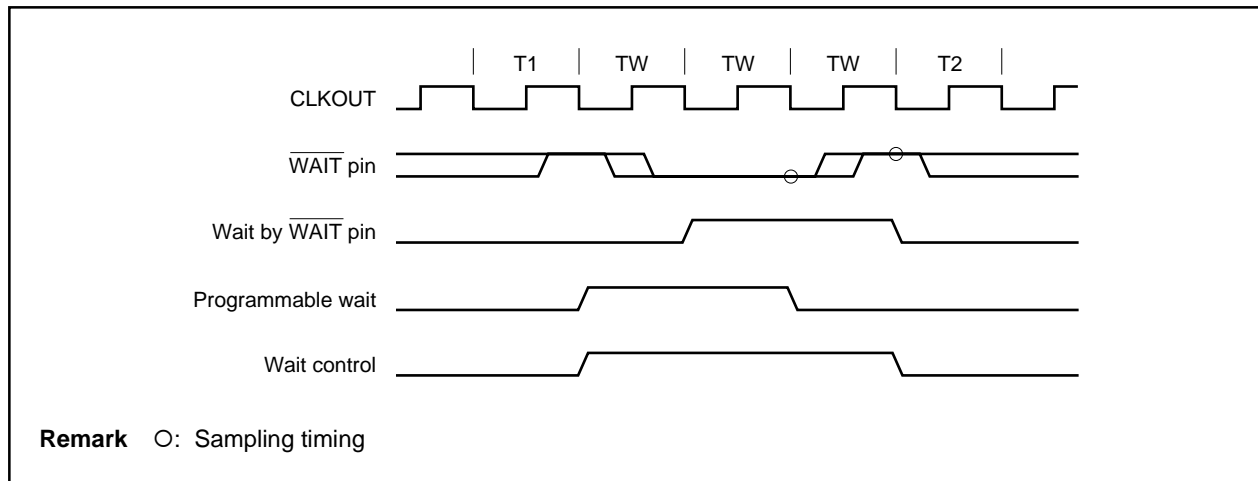
#### 4.6.3 Relationship between programmable wait and external wait

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 whichever has the most cycles.



For example, if the programmable wait is two waits, 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**



#### 4.6.4 Bus cycles in which the wait function is valid

In the V850E/MS2, the number of waits can be specified according to the type of memory specified for each memory block.

The registers which set the bus cycles and waits in which the wait function is valid are as shown below.

**Table 4-1. Bus Cycles in Which the Wait Function Is Valid (1/2)**

Bus Cycle			Type of Wait	Programmable Wait Setting		Wait by WAIT Pin		
				Higher Order: Register Lower Order: Bit	Number of Waits			
SRAM, external ROM, external I/O cycle			Data access wait	DWC1, DWC2	0 to 7	○		
				DWxx				
Page ROM cycle	Off-page		Data access wait	DWC1, DWC2	0 to 7	○		
				DWxx				
	On-page		Data access wait	PRC	0 to 7	○		
				PRW0 to PRW2				
EDO DRAM, high-speed page DRAM cycle	Read access	Off-page	RAS pre-charge	DRCn	0 to 3	×		
				RPC0n, RPC1n				
			Row address hold	DRCn	0 to 3	×		
				RHC0n, RHC1n				
			Data access wait	DRCn	0 to 3	Note		
				DAC0n, DAC1n				
		On-page	CAS pre-charge	DRCn	0 to 3	×		
				CPC0n, CPC1n				
			Data access wait	DRCn	0 to 3	×		
				DAC0n, DAC1n				
			Write access	Off-page	RAS pre-charge	DRCn	0 to 3	×
						RPC0n, RPC1n		
	Row address hold	DRCn			0 to 3	Note		
		RHC0n, RHC1n						
	Data access wait	DRCn			0 to 3	×		
		DAC0n, DAC1n						
	On-page	CAS pre-charge		DRCn	0 to 3	×		
				CPC0n, CPC1n				
		Data access wait	DRCn	0 to 3	×			
			DAC0n, DAC1n					
CBR refresh cycle			RAS pre-charge	RWC	0 to 3	×		
				RRW0, RRW1				
			RAS active width	RWC	0 to 7	×		
				RCW0 to RCW2				

**Note** EDO DRAM cycle: ×  
High-speed page DRAM cycle: ○

**Remarks** 1. ○: Valid ×: Invalid

2. n = 0 to 3

xx = 00 to 02, 10 to 12, 20 to 22, 30 to 32, 40 to 42, 50 to 52, 60 to 62, 70 to 72

Table 4-1. Bus Cycles in Which the Wait Function Is Valid (2/2)

Bus Cycle			Type of Wait		Programmable Wait Setting		Wait by $\overline{\text{WAIT}}$ Pin
					Higher Order: Register Lower Order: Bit	Number of Waits	
CBR self-refresh cycle			RAS pre-charge		RWC	0 to 3	×
					RRW0, RRW1		
			RAS active width		RWC	0 to 7	×
					RCW0 to RCW2		
			Self-refresh release width		RWC	0 to 14	×
					SRW0 to SRW2		
DMA flyby transfer cycle	External I/O ↔ SRAM		Data access wait	TW	DWC1, DWC2	0 to 7	○
					DWxx		
				TF	FDW	0, 1	×
					FDWm		
			RAS pre-charge		DRCn	0 to 3	×
					RPC0n, RPC1n		
	DRAM → External I/O	Off-page	Row address hold		DRCn	0 to 3	×
					RHC0n, RHC1n		
			Data access wait	TW	DRCn	0 to 3	○
					DAC0n, DAC1n		
				TF	FDW	0, 1	×
					FDWm		
		On-page	CAS pre-charge		DRCn	0 to 3	×
					CPC0n, CPC1n		
			Data access wait	TW	DRCn	0 to 3	○
					DAC0n, DAC1n		
				TF	FDW	0, 1	×
					FDWm		
	External I/O → DRAM	Off-page	RAS pre-charge		DRCn	0 to 3	×
					RPC0n, RPC1n		
			Row address hold		DRCn	0 to 3	○
					RHC0n, RHC1n		
			Data access wait	TW	DRCn	0 to 3	×
					DAC0n, DAC1n		
			TF		FDW	0, 1	×
					FDWm		
		On-page	CAS pre-charge		DRCn	1 to 3	○
					CPC0n, CPC1n		
			Data access wait	TW	DRCn	0 to 3	×
					DAC0n, DAC1n		
			TF		FDW	0, 1	×
					FDWm		

**Remarks** 1. ○: Valid    ×: Invalid

2. n = 0 to 3

m = 0 to 7

xx = 00 to 02, 10 to 12, 20 to 22, 30 to 32, 40 to 42, 50 to 52, 60 to 62, 70 to 72

## 4.7 Idle State Insertion Function

To facilitate interfacing with low-speed memory devices, an idle state (TI) can be inserted into the current bus cycle after the T2 state in order to meet the data output float delay time ( $t_{dF}$ ) on memory read accesses for each memory block. The bus cycle following the T2 state starts after the idle state is inserted.

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

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

The idle state is inserted only if the read cycle is followed by a write cycle.

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

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

BCC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BC71	BC70	BC61	BC60	BC51	BC50	BC41	BC40	BC31	BC30	BC21	BC20	BC11	BC10	BC01	BC00

Address FFFF062H

After reset 5555H

Memory block

7

6

5

4

3

2

1

0

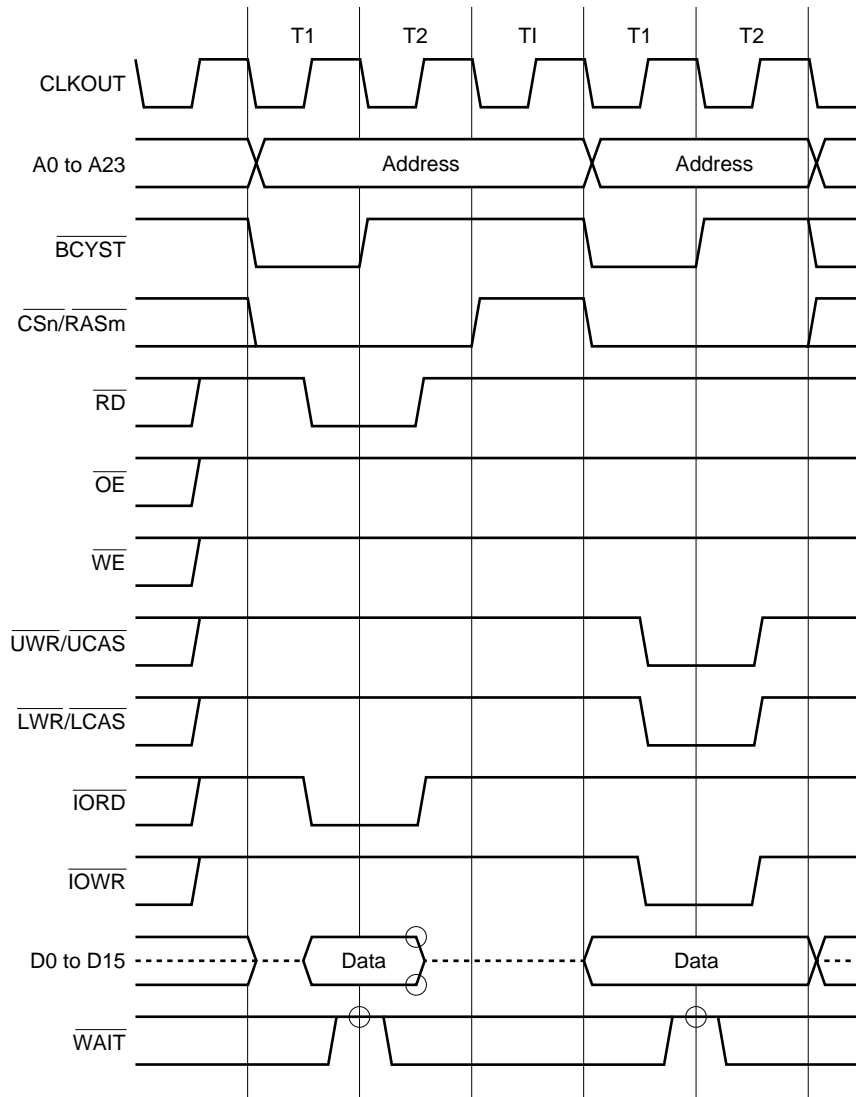
Bit Position	Bit Name	Function												
15 to 0	BCn1, BCn0 (n = 7 to 0)	<p>Bus Cycle</p> <p>Specifies insertion of an idle state in memory block n.</p> <table border="1" style="border-collapse: collapse; width: 100%; margin-top: 10px;"> <thead> <tr> <th style="width: 15%;">BCn1</th> <th style="width: 15%;">BCn0</th> <th style="width: 70%;">Idle State in Memory Block n</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Not inserted</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Inserted</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">Optional</td> <td>RFU (Reserved)</td> </tr> </tbody> </table>	BCn1	BCn0	Idle State in Memory Block n	0	0	Not inserted	0	1	Inserted	1	Optional	RFU (Reserved)
BCn1	BCn0	Idle State in Memory Block n												
0	0	Not inserted												
0	1	Inserted												
1	Optional	RFU (Reserved)												

**Cautions**

1. The internal RAM area and internal peripheral I/O area are not subject to insertion of an idle state.
2. Write to the BCC register after reset, and then do not change the set value. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the BCC register is complete. However, it is possible to access an external memory area whose initialization is complete.

## (2) Idle state insertion timing



- Remarks**
1. The circle indicates the sampling timing.
  2. The broken lines indicate high impedance.
  3.  $n = 0, 3 \text{ to } 5$   
 $m = 3 \text{ to } 5$

## 4.8 Bus Hold Function

### 4.8.1 Outline of function

If pins P96 and P97 are specified in the control mode, the  $\overline{\text{HLDAK}}$  and  $\overline{\text{HLDRQ}}$  functions become valid.

If it is determined that the  $\overline{\text{HLDRQ}}$  pin has become active (low level) as a bus acquisition request from another bus master, the external address/data bus and each strobe pin are shifted to high impedance and released (bus hold state). If the  $\overline{\text{HLDRQ}}$  pin becomes inactive (high level) and the bus acquisition request is canceled, driving of these pins begins again.

During the bus hold interval, internal operations in the V850E/MS2 continue until there is external memory access.

The bus hold state can be known by the  $\overline{\text{HLDAK}}$  pin becoming active (low level).

In a multiprocessor configuration, etc., a system that has multiple bus masters can be configured.

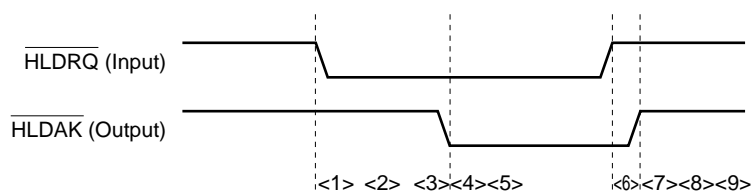
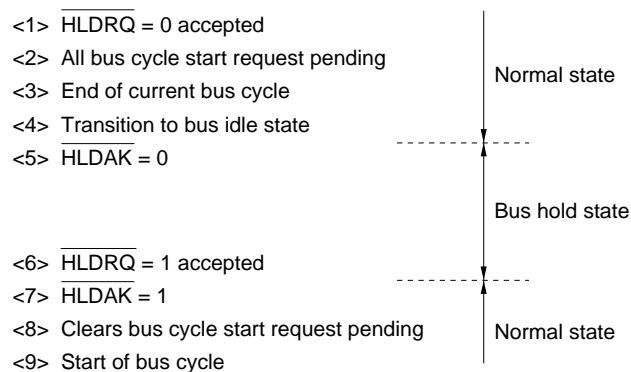
Note that bus hold requests are not received with the following timings.

**Caution** The  $\overline{\text{HLDRQ}}$  function is invalid during the reset period. When the  $\overline{\text{RESET}}$  pin and  $\overline{\text{HLDRQ}}$  pin are made active simultaneously, and then the  $\overline{\text{RESET}}$  pin is made inactive, the  $\overline{\text{HLDAK}}$  pin becomes active after a one-clock idle cycle has been inserted. Note that for a power-on reset, even if the  $\overline{\text{RESET}}$  pin and  $\overline{\text{HLDRQ}}$  pin are made active simultaneously, and then the  $\overline{\text{RESET}}$  pin is made inactive, the  $\overline{\text{HLDAK}}$  pin does not become active. When a bus master other than the V850E/MS2 is externally connected, execute arbitration at the moment of power-on using the  $\overline{\text{RESET}}$  signal.

State	Data Bus Width	Access Configuration	Timing in Which Bus Hold Request Will Not Be Received
CPU bus lock	16 bits	Word access to even address	Between 1st and 2nd times
		Word access to odd address	Between 1st and 2nd times
			Between 2nd and 3rd times
	8 bits	Halfword access to odd address	Between 1st and 2nd times
		Word access	Between 1st and 2nd times
			Between 2nd and 3rd times
			Between 3rd and 4th times
		Halfword access	Between 1st and 2nd times
Read modify write access to bit operation instruction	—	—	Between read access and write access

### 4.8.2 Bus hold procedure

The procedure of the bus hold function is illustrated below.



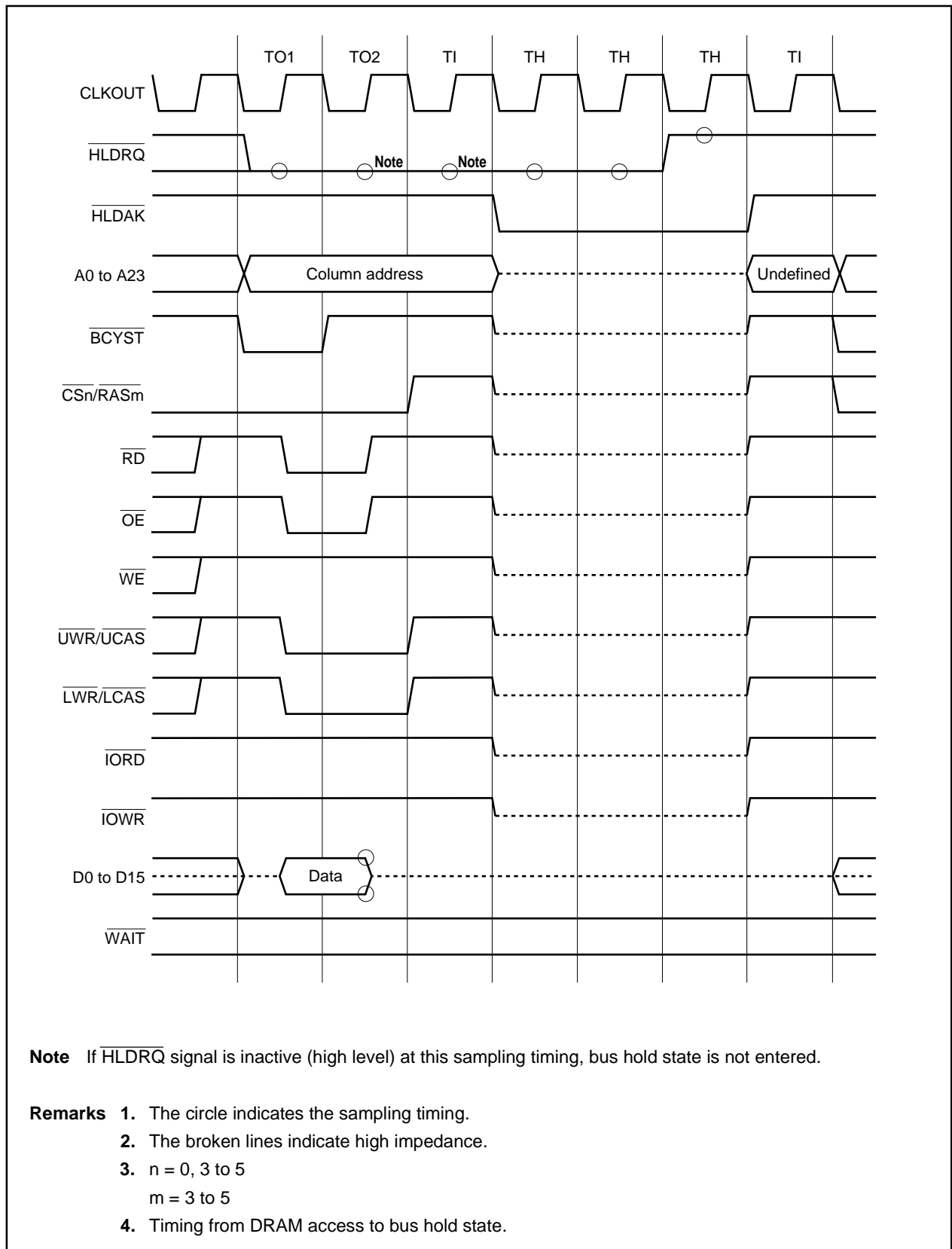
### 4.8.3 Operation in power save mode

In the STOP or IDLE mode, the internal system clock is stopped. Consequently, the bus hold state is not accepted and 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 state is set. When the  $\overline{\text{HLDRQ}}$  pin becomes inactive, the  $\overline{\text{HLDAK}}$  pin becomes inactive. As a result, the bus hold state is cleared, and the HALT mode is set again.



#### 4.8.4 Bus hold timing



## 4.9 Bus Priority Order

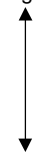
There are five external bus cycles: bus hold, instruction fetch, operand data access, DMA cycle and refresh cycle.

Bus hold has the highest priority, then the refresh cycle, DMA cycle, instruction fetch and operand data access, in descending order.

Between read access and write access in read modify write access, an instruction fetch may be inserted.

Also, between bus access and bus access during CPU bus lock, an instruction fetch may be inserted.

**Table 4-2. Bus Priority Order**

Priority Order	External Bus Cycle	Bus Master
High  Low	Bus hold	External device
	Refresh cycle	DRAM controller
	DMA cycle	DMA controller
	Instruction fetch	CPU
	Operand data access	CPU

## 4.10 Boundary Operation Conditions

### 4.10.1 Program space

- (1) Branching to the peripheral I/O area or successive fetch from the internal RAM area to the internal peripheral I/O area is prohibited. In terms of hardware, fetching the NOP op code continues, and fetching from the external memory is not performed.
- (2) If a branch instruction exists at the upper limit of the internal RAM area, a pre-fetch operation (invalid fetch) that straddles over the internal peripheral I/O area does not occur when instruction fetch is performed.
- (3) In burst fetch mode, if an instruction fetch is performed for contiguous memory blocks, the burst fetch is terminated at the upper limit of a block, and the start-up cycle is started at the lower limit of the next block.
- (4) Burst fetch is valid only in the external memory area. In memory block 7, it is terminated when the internal address count value has reached the upper limit of the external memory area.

**4.10.2 Data space**

The V850E/MS2 incorporates an address misalign function.

Through this function, regardless of the data format (word data, halfword data), data can be placed in all addresses. However, in the case of word data and halfword data, if data is not subject to boundary alignment, the bus cycle will be generated at least 2 times and bus efficiency will drop.

**(1) In the case of halfword length data access**

When the address's lowest bit is a 1, the byte length bus cycle will be generated 2 times.

**(2) In the case of word length data access**

- (a) When the address's lowest bit is a 1, bus cycles will be generated in the order of byte length bus cycle, halfword length bus cycle, and byte length bus cycle.
- (b) When the address's lower 2 bits are 10, the halfword length bus cycle will be generated 2 times.

[MEMO]

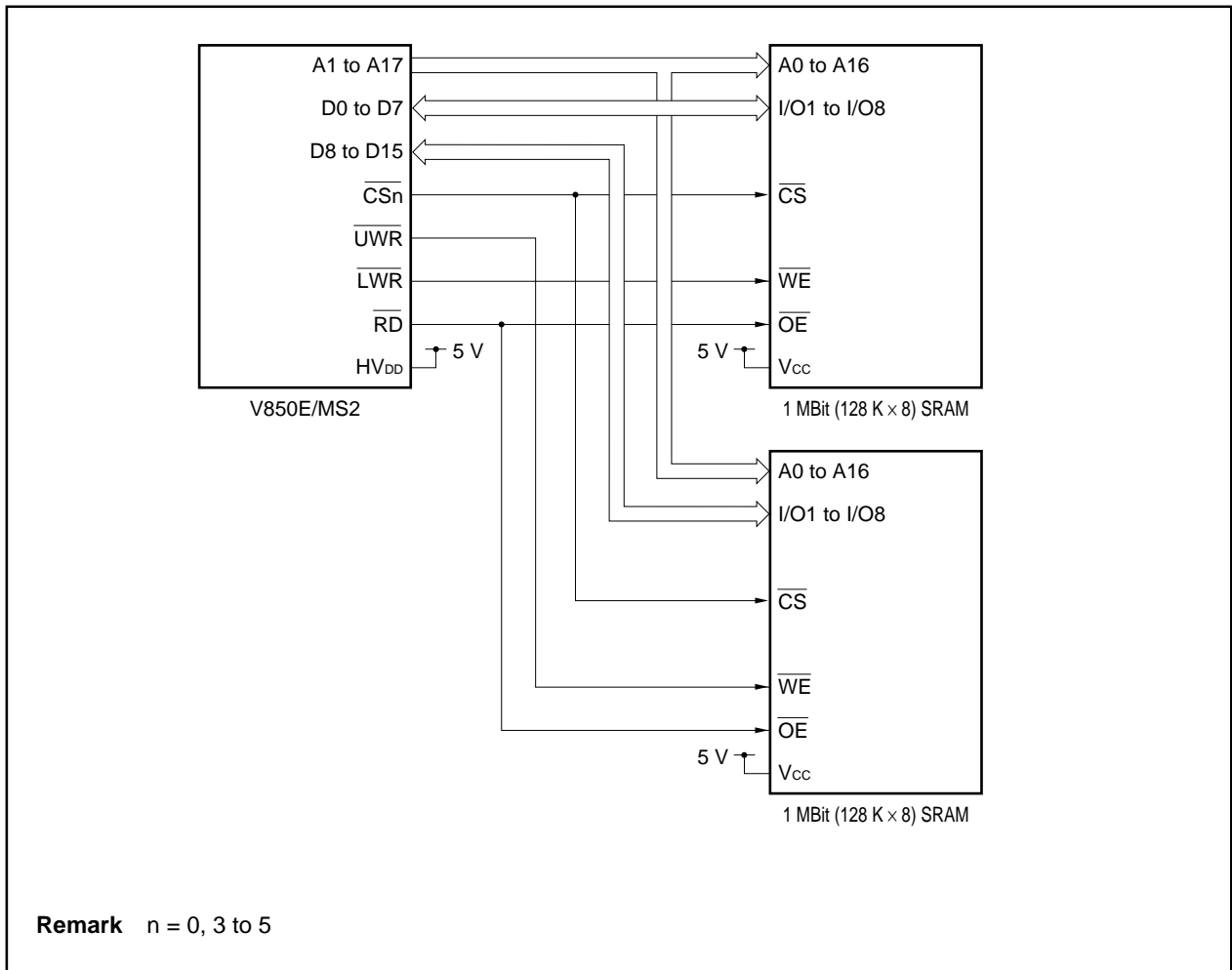
## CHAPTER 5 MEMORY ACCESS CONTROL FUNCTION

### 5.1 SRAM, External ROM, External I/O Interface

#### 5.1.1 SRAM connections

An example of connection to SRAM is shown below.

Figure 5-1. Example of Connection to SRAM



## 5.1.2 SRAM, external ROM, external I/O access

Figure 5-2. SRAM, External ROM, External I/O Access Timing (1/4)

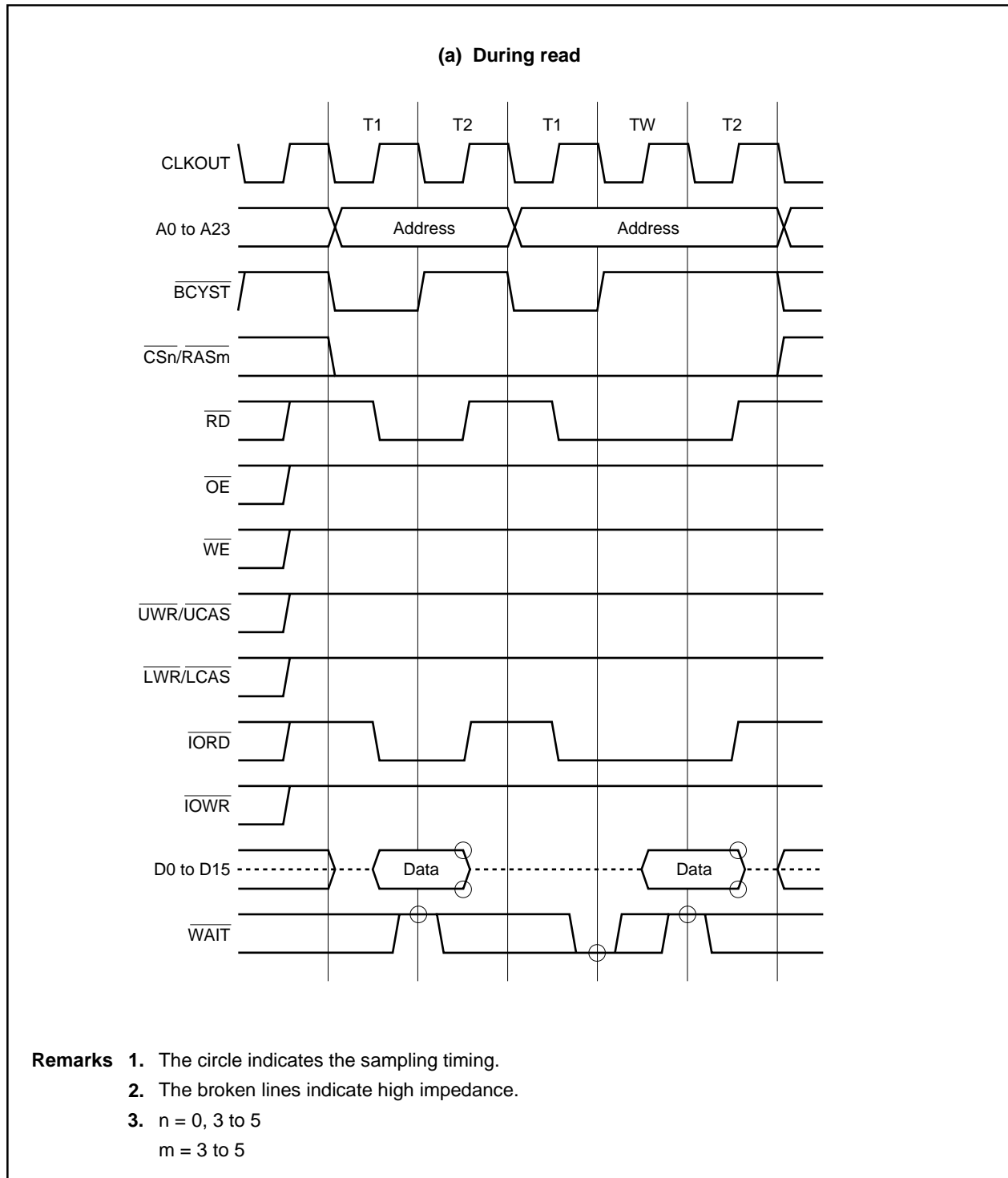
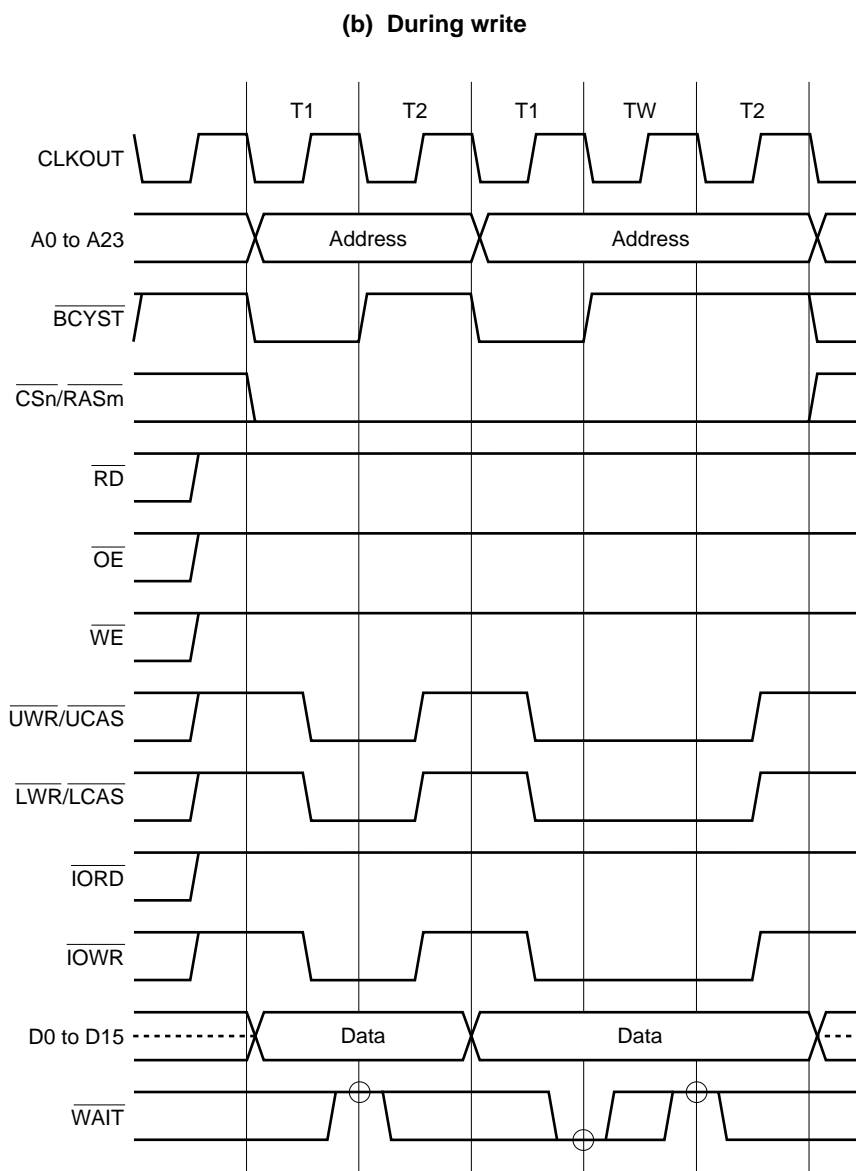
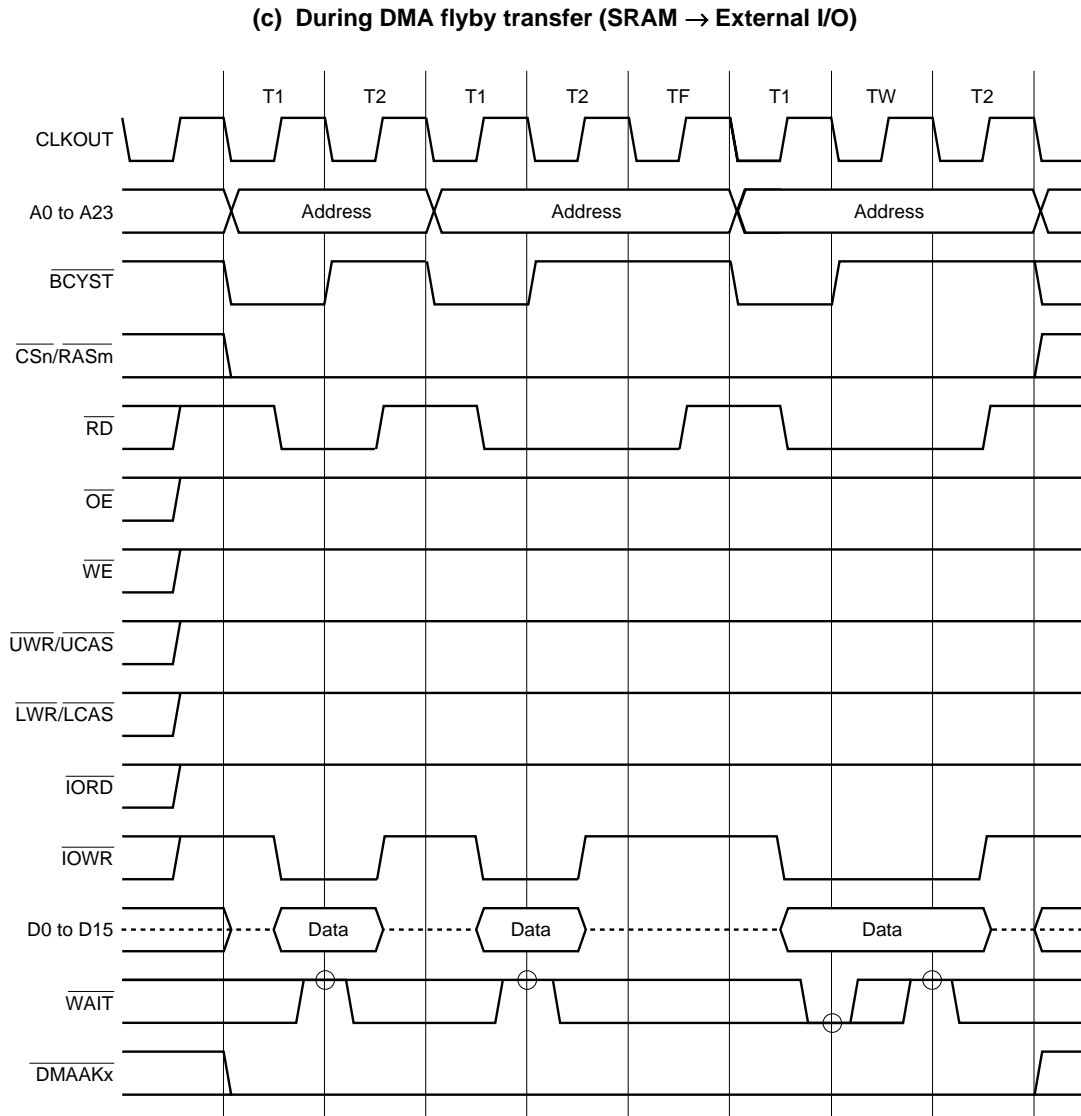


Figure 5-2. SRAM, External ROM, External I/O Access Timing (2/4)



- Remarks**
1. The circle indicates the sampling timing.
  2. The broken lines indicate high impedance.
  3.  $n = 0, 3$  to 5  
 $m = 3$  to 5

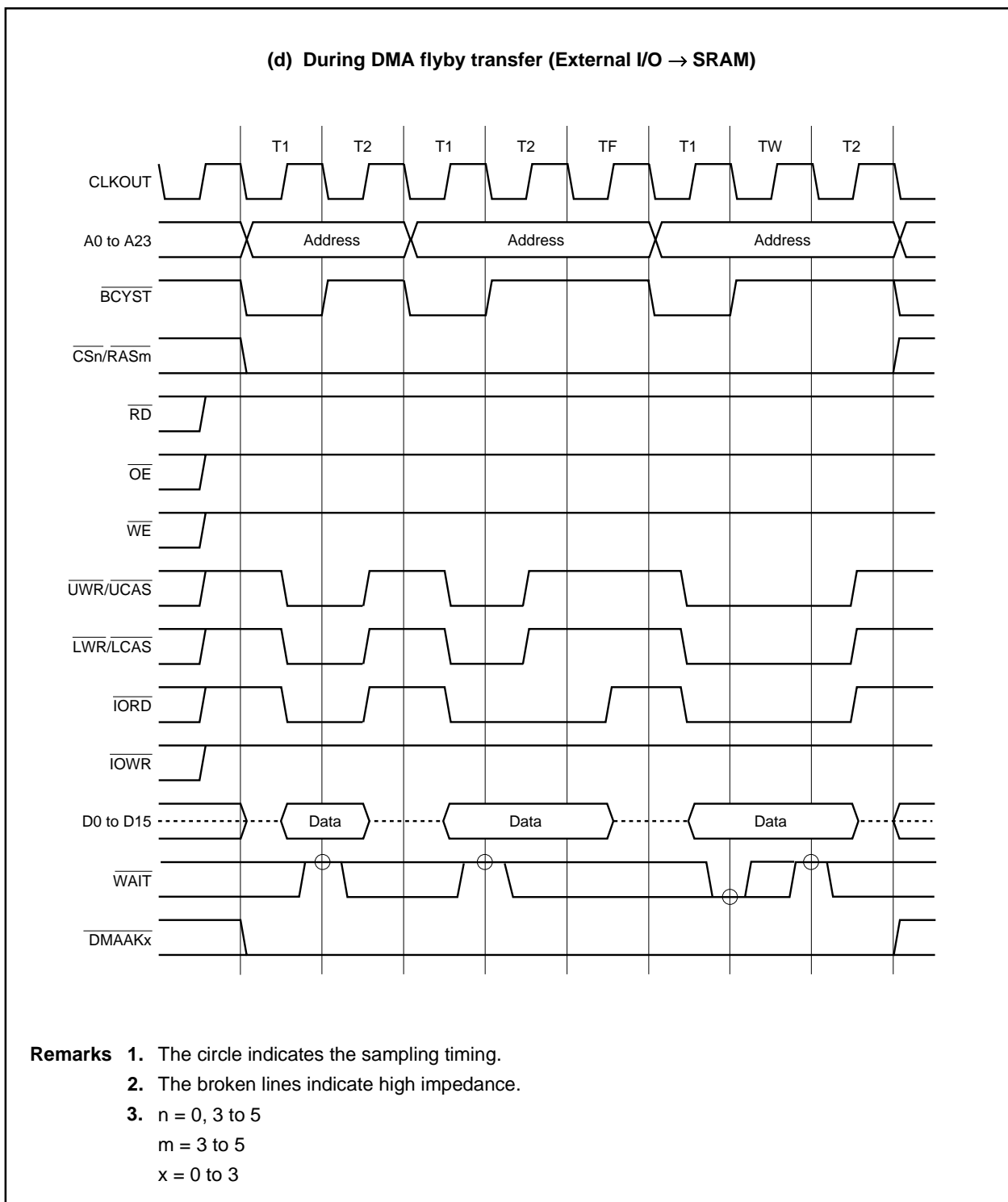
Figure 5-2. SRAM, External ROM, External I/O Access Timing (3/4)



- Remarks**
1. The circle indicates the sampling timing.
  2. The broken lines indicate high impedance.
  3.  $n = 0, 3$  to  $5$   
 $m = 3$  to  $5$   
 $x = 0$  to  $3$



Figure 5-2. SRAM, External ROM, External I/O Access Timing (4/4)



## 5.2 Page ROM Controller (ROMC)

The page ROM controller (ROMC) is for access to ROM (page ROM) with a page access function.

Comparison of addresses with the immediately previous bus cycle is carried out and wait control for normal access (off-page) and page access (on-page) is executed. This controller is capable of handling page widths of from 8 to 64 bytes.

### 5.2.1 Features

- It can connect directly to 8-bit/16-bit page ROM.
- When the bus width is 16 bits, it can handle 4/8/16/32-word page access.  
When the bus width is 8 bits, it can handle 8/16/32/64-word page access.
- Individual wait settings (0 to 7 waits) for off-page and on-page are possible.

### 5.2.2 Page ROM connections

Examples of page ROM connections are shown below.

**Figure 5-3. Example of Page ROM Connections (1/2)**

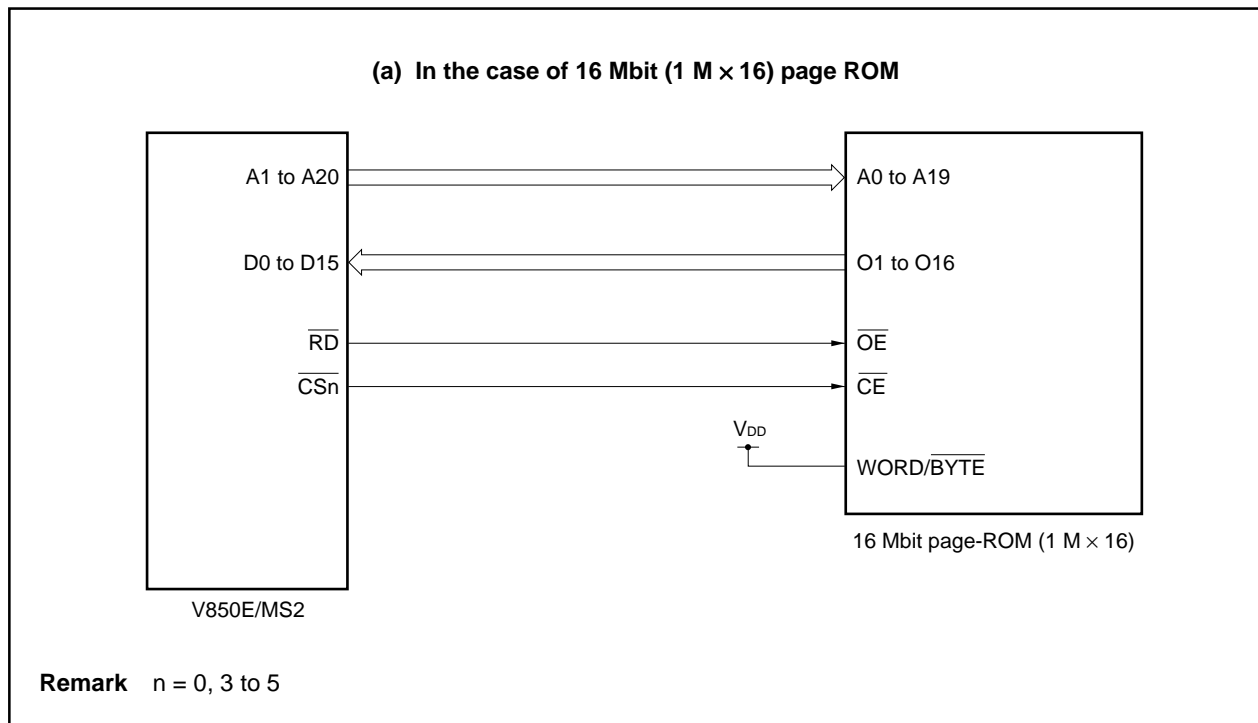
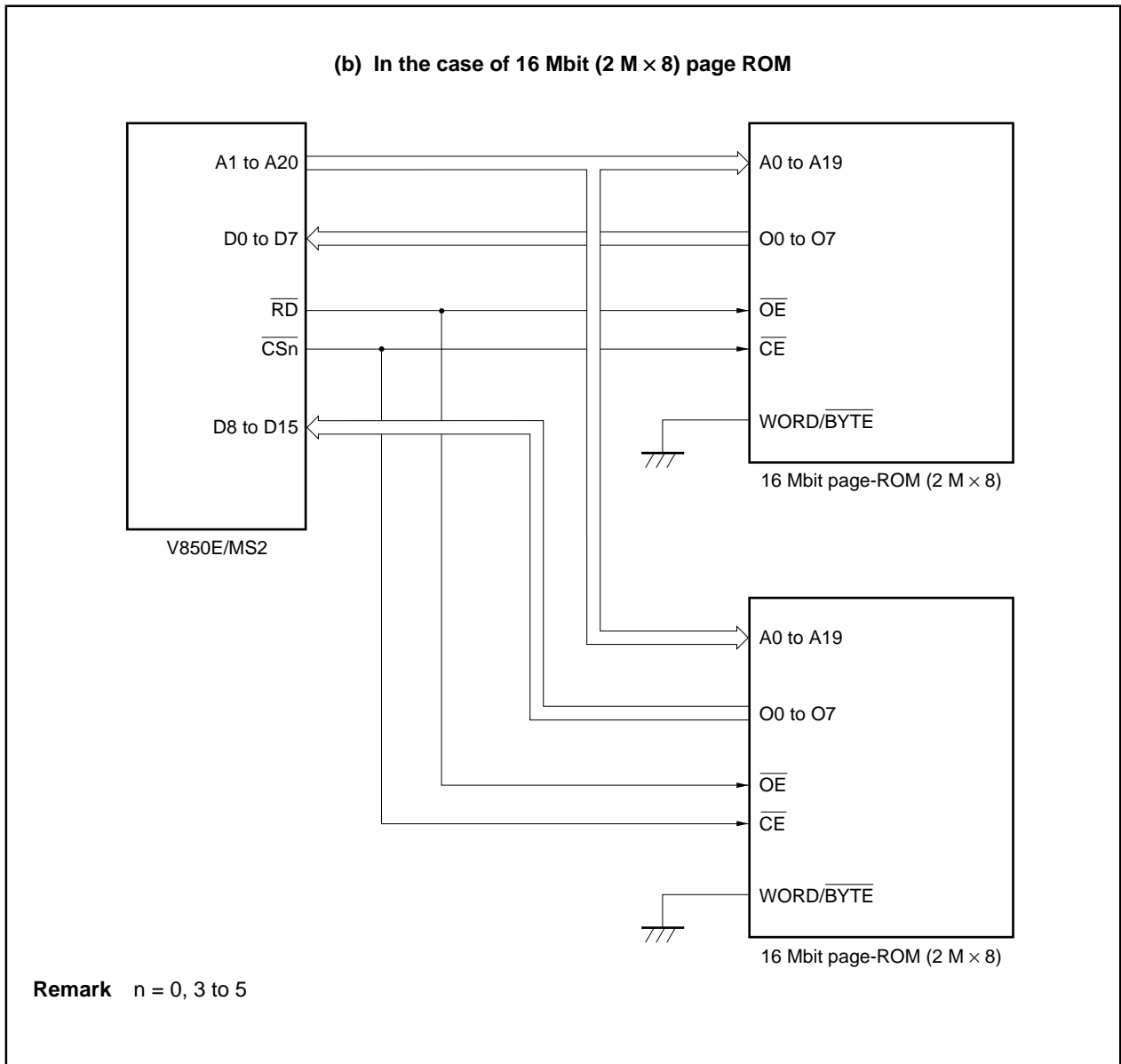


Figure 5-3. Example of Page ROM Connections (2/2)



### 5.2.3 On-page/off-page judgment

Whether a page ROM cycle is on-page or off-page is judged by latching the address of the previous cycle and comparing it with the address of the current cycle.

Using the page ROM configuration register (PRC), one of the addresses (A3 to A5) is set as the masking address (no comparison is made) according to the configuration of the connected page ROM and the number of continuously readable bits.

**Figure 5-4. On-Page/Off-Page Judgment for Page ROM Connection (1/2)**

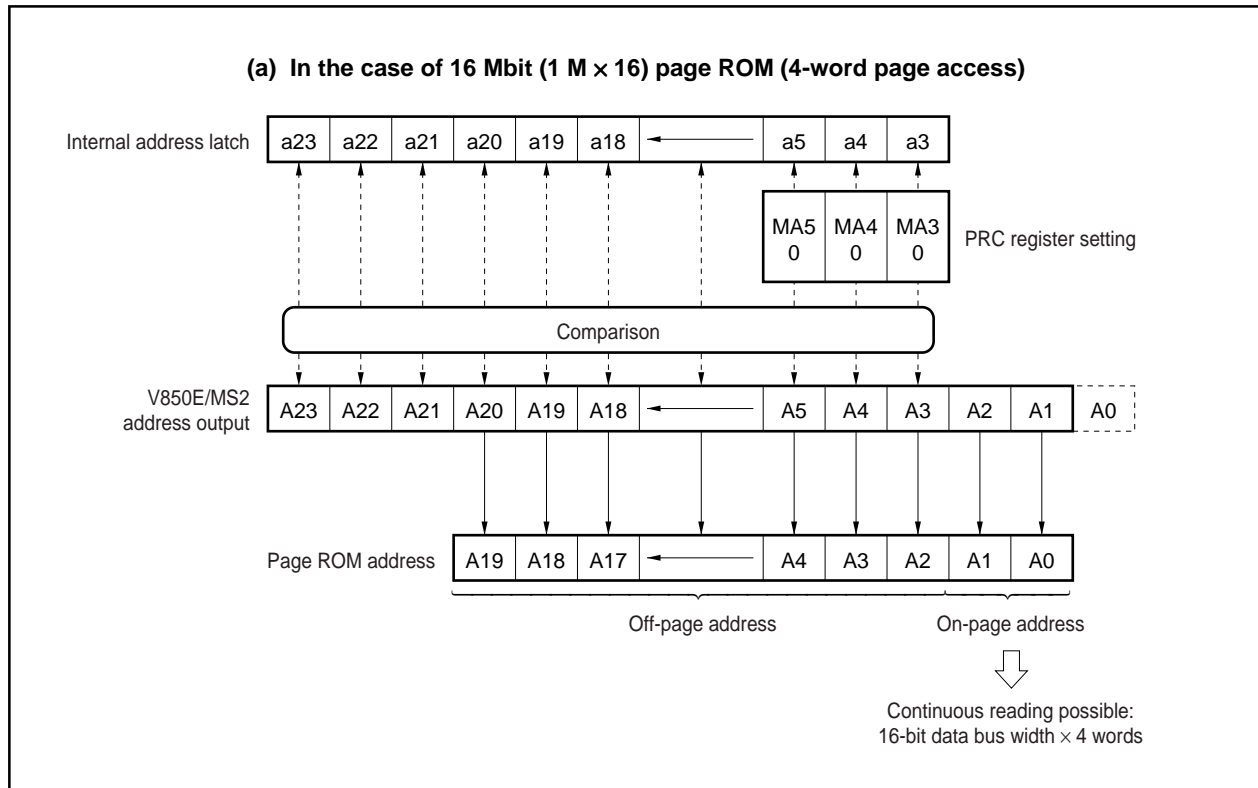
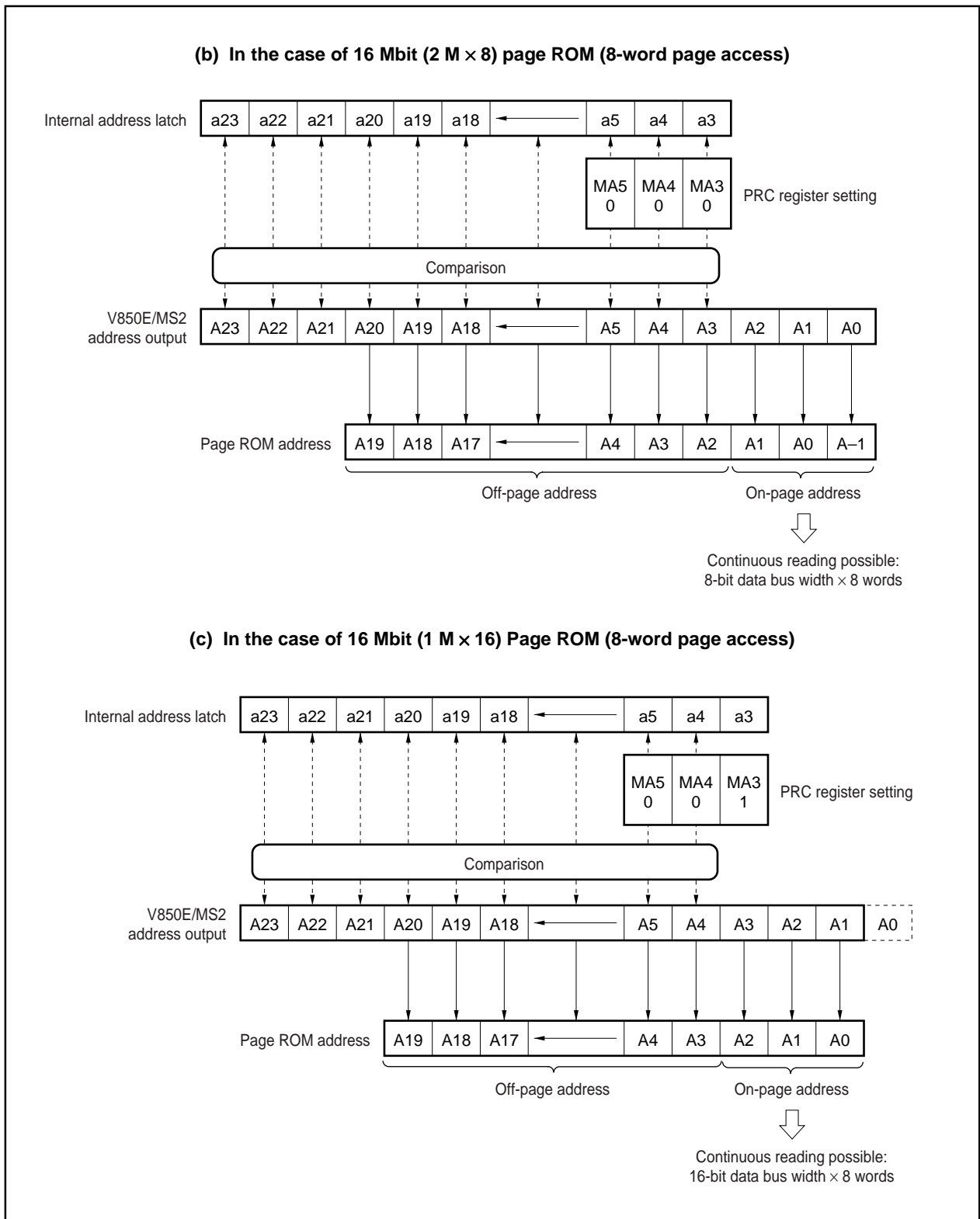


Figure 5-4. On-Page/Off-Page Judgment for Page ROM Connection (2/2)



### 5.2.4 Page ROM configuration register (PRC)

This specifies whether page ROM on-page access is enabled or disabled. Also, if on-page access is enabled, the masked addresses (no comparison is made) out of the addresses (A3 to A5) corresponding to the configuration of the connected page ROM and the number of bits that can be read continuously, as well as the number of waits corresponding to the internal system clock, are set.

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

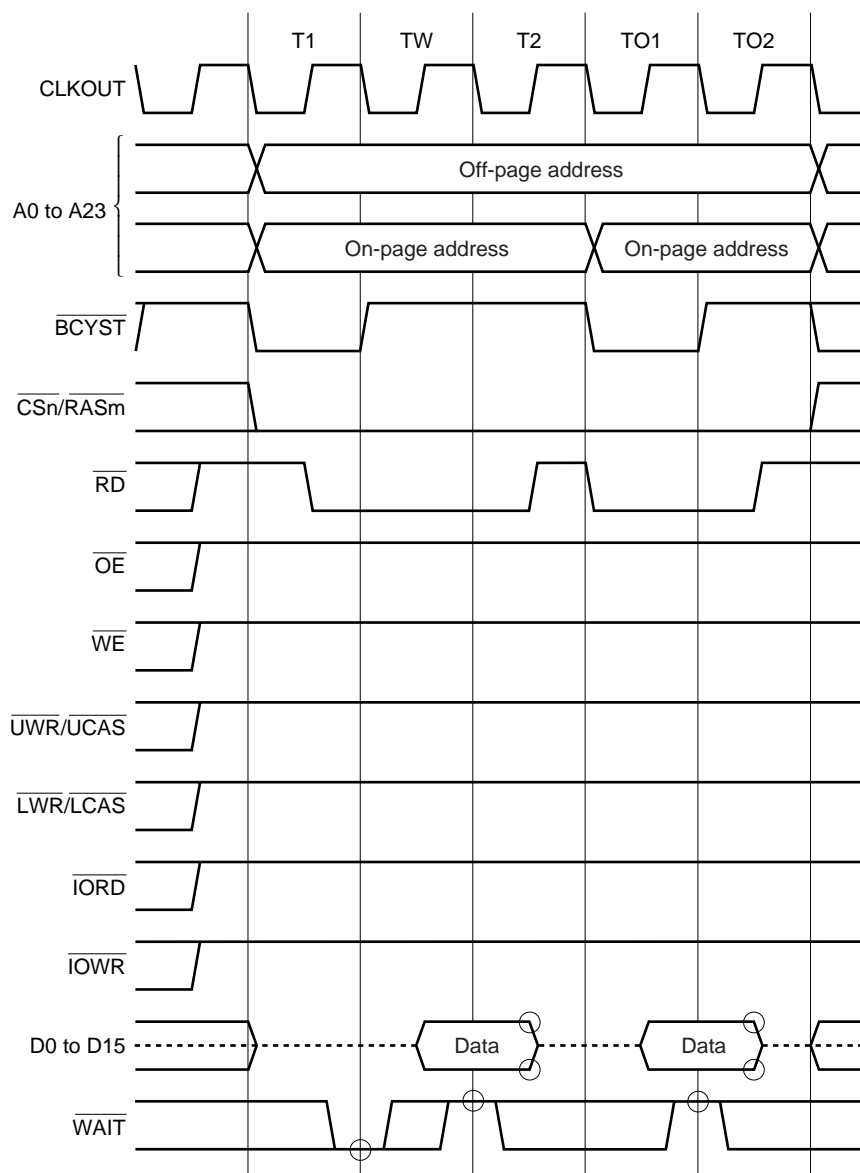
	7	6	5	4	3	2	1	0		
PRC	PAE	PRW2	PRW1	PRW0	0	MA5	MA4	MA3	Address FFFFFF24H	After reset 70H

Bit Position	Bit Name	Function																																				
7	PAE	Page ROM On-page Access Enable Specifies whether page ROM on-page access is enabled or disabled. 0: Disable 1: Enable																																				
6 to 4	PRW2 to PRW0	Page-ROM On-page Access Wait Control Sets the number of waits corresponding to the internal system clock. The waits set by this bit are inserted only for on-page access. For off-page access, the waits set by registers DWC1 and DWC2 are inserted (refer to <b>4.6 Wait Function</b> ). <table><tr><th>PRW2</th><th>PRW1</th><th>PRW0</th><th>Number of Inserted Wait Cycles</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>2</td></tr><tr><td>0</td><td>1</td><td>1</td><td>3</td></tr><tr><td>1</td><td>0</td><td>0</td><td>4</td></tr><tr><td>1</td><td>0</td><td>1</td><td>5</td></tr><tr><td>1</td><td>1</td><td>0</td><td>6</td></tr><tr><td>1</td><td>1</td><td>1</td><td>7</td></tr></table>	PRW2	PRW1	PRW0	Number of Inserted Wait Cycles	0	0	0	0	0	0	1	1	0	1	0	2	0	1	1	3	1	0	0	4	1	0	1	5	1	1	0	6	1	1	1	7
PRW2	PRW1	PRW0	Number of Inserted Wait Cycles																																			
0	0	0	0																																			
0	0	1	1																																			
0	1	0	2																																			
0	1	1	3																																			
1	0	0	4																																			
1	0	1	5																																			
1	1	0	6																																			
1	1	1	7																																			
2 to 0	MA5 to MA3	Mask Address Each address (A5 to A3) corresponding to MA5 to MA3 is masked (by 1). The masked address is not subject to comparison during on/off-page judgment. It is set according to the number of continuously readable bits. <table><tr><th>MA5</th><th>MA4</th><th>MA3</th><th>Number of Continuously Readable Bits</th></tr><tr><td>0</td><td>0</td><td>0</td><td>4 words × 16 bits (8 words × 8 bits)</td></tr><tr><td>0</td><td>0</td><td>1</td><td>8 words × 16 bits (16 words × 8 bits)</td></tr><tr><td>0</td><td>1</td><td>1</td><td>16 words × 16 bits (32 words × 8 bits)</td></tr><tr><td>1</td><td>1</td><td>1</td><td>32 words × 16 bits (64 words × 8 bits)</td></tr></table>	MA5	MA4	MA3	Number of Continuously Readable Bits	0	0	0	4 words × 16 bits (8 words × 8 bits)	0	0	1	8 words × 16 bits (16 words × 8 bits)	0	1	1	16 words × 16 bits (32 words × 8 bits)	1	1	1	32 words × 16 bits (64 words × 8 bits)																
MA5	MA4	MA3	Number of Continuously Readable Bits																																			
0	0	0	4 words × 16 bits (8 words × 8 bits)																																			
0	0	1	8 words × 16 bits (16 words × 8 bits)																																			
0	1	1	16 words × 16 bits (32 words × 8 bits)																																			
1	1	1	32 words × 16 bits (64 words × 8 bits)																																			

**Caution** Write to the PRC register after reset, and then do not change the set value. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the PRC register is complete. However, it is possible to access an external memory area whose initialization is complete.

## 5.2.5 Page ROM access

Figure 5-5. Page ROM Access Timing



- Remarks**
1. The circle indicates the sampling timing.
  2. The broken lines indicate high impedance.
  3.  $n = 0, 3$  to 5  
 $m = 3$  to 5

## 5.3 DRAM Controller

### 5.3.1 Features

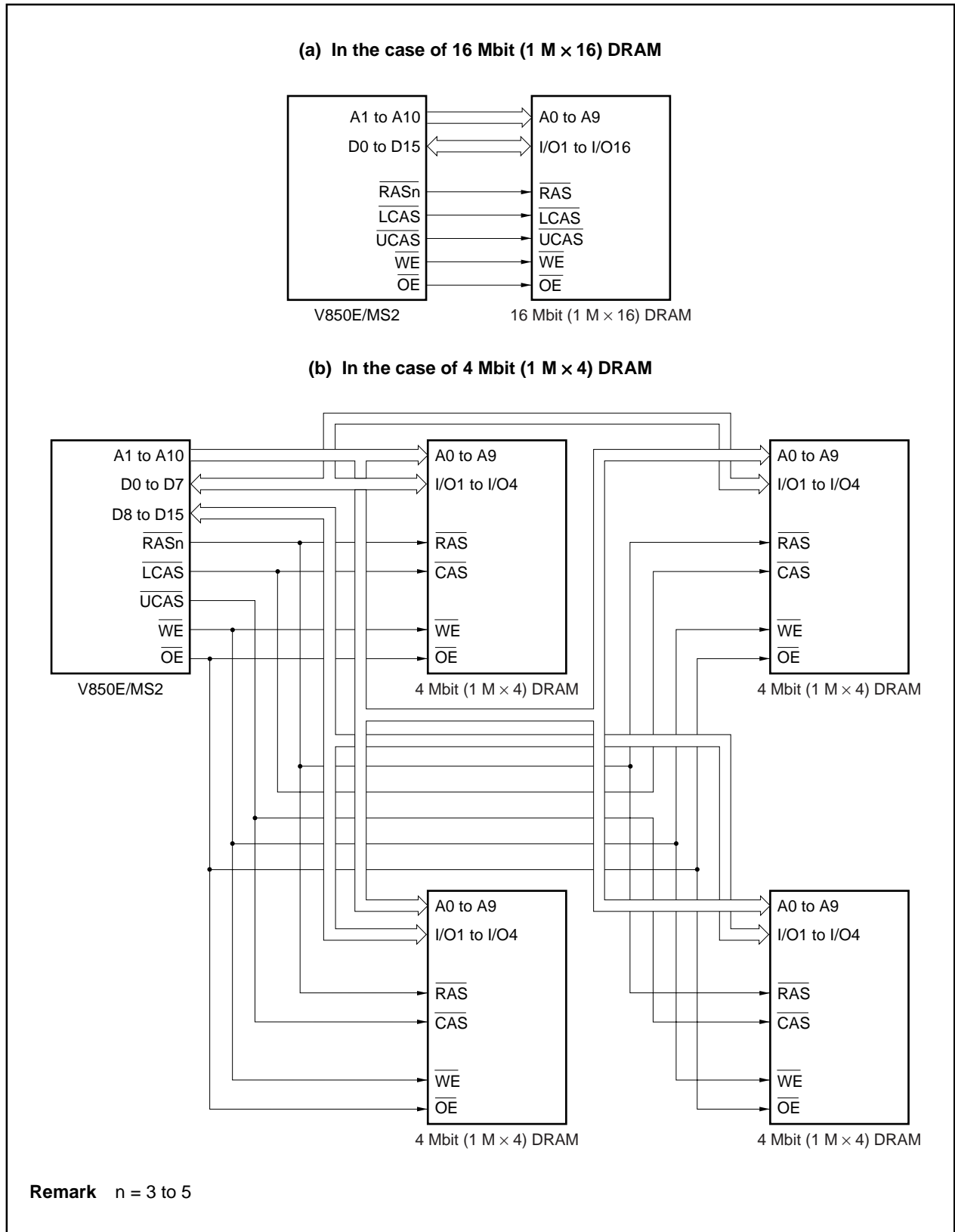
- Generates the  $\overline{\text{RAS}}$ ,  $\overline{\text{LCAS}}$  and  $\overline{\text{UCAS}}$  signals.
- Can be connected directly to high-speed page DRAM and EDO DRAM.
- Supports the RAS hold mode.
- 4 types of DRAM can be assigned to 8 memory block spaces.
- Can handle 2CAS type DRAM
- Can be switched between row and column address multiplex widths.
- Waits (0 to 3 waits) can be inserted at the following timings.
  - Row address precharge wait
  - Row address hold wait
  - Data access wait
  - Column address precharge wait
- Supports CBR refresh and CBR self-refresh.



### 5.3.2 DRAM connections

Examples of connections to DRAM are shown below.

Figure 5-6. Examples of Connections to DRAM



### 5.3.3 Address multiplex function

Depending on the value of the DAW0n and DAW1n bits in DRAM configuration register n (DRCn), the row address, column address output in the DRAM cycle is multiplexed as shown in Figure 5-7 (n = 0 to 3). In Figure 5-7, a0 to a23 show the addresses output from the CPU and A0 to A23 show the V850E/MS2's address pins. For example, when DAW0n and DAW1n = 11, it indicates that a12 to a22 are output from the address pins (A1 to A11) as row addresses and a1 to a11 are output as column addresses.

Table 5-1 shows the relationship between connectable DRAM and the address multiplex width. Depending on the DRAM being connected, DRAM space is from 128 KB to 8 MB.

**Figure 5-7. Row Address/Column Address Output**

Address pin	A23 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Row address (DAW1n, DAW0n = 11)	a23 to a18	a17	a16	a15	a25	a24	a23	a22	a21	a20	a19	a18	a17	a16	a15	a14	a13	a12	a11
Row address (DAW1n, DAW0n = 10)	a23 to a18	a17	a16	a25	a24	a23	a22	a21	a20	a19	a18	a17	a16	a15	a14	a13	a12	a11	a10
Row address (DAW1n, DAW0n = 01)	a23 to a18	a17	a25	a24	a23	a22	a21	a20	a19	a18	a17	a16	a15	a14	a13	a12	a11	a10	a9
Row address (DAW1n, DAW0n = 00)	a23 to a18	a25	a24	a23	a22	a21	a20	a19	a18	a17	a16	a15	a14	a13	a12	a11	a10	a9	a8
Column address	a23 to a18	a17	a16	a15	a14	a13	a12	a11	a10	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0

**Table 5-1. Example of DRAM and Address Multiplex Width**

Address Multiplex Width	DRAM Capacity (Bits) and Configuration					DRAM Space (Bytes)
	256 K	1 M	4 M	16 M	64 M	
8 bits	64 K × 4	—	—	—	—	128 K
9 bits	—	256 K × 4	256 K × 16	—	—	512 K
	—	—	512 K × 8	—	—	1 M
	—	—	—	—	4 M × 16	8 M
10 bits	—	—	1 M × 4	1 M × 16	—	2 M
	—	—	—	2 M × 8	—	4 M
	—	—	—	—	4 M × 16	8 M
11 bits	—	—	—	4 M × 4	—	8 M

### 5.3.4 DRAM configuration registers 0 to 3 (DRC0 to DRC3)

This sets the type of DRAM to be connected.

These registers can be read/written in 16-bit units.

**Caution** If the object of access is a DRAM area, the wait set in registers DWC1 and DWC2 becomes invalid.  
In this case, waits are controlled by registers DRC0 to DRC3.

(1/3)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DRC0	PAE 10	PAE 00	RPC 10	RPC 00	RHC 10	RHC 00	DAC 10	DAC 00	CPC 10	CPC 00	0	RHD 0	0	0	DAW 10	DAW 00	Address FFFFFF200H	After reset 3FC1H
DRC1	PAE 11	PAE 01	RPC 11	RPC 01	RHC 11	RHC 01	DAC 11	DAC 01	CPC 11	CPC 01	0	RHD 1	0	0	DAW 11	DAW 01	FFFFFF202H	3FC1H
DRC2	PAE 12	PAE 02	RPC 12	RPC 02	RHC 12	RHC 02	DAC 12	DAC 02	CPC 12	CPC 02	0	RHD 2	0	0	DAW 12	DAW 02	FFFFFF204H	3FC1H
DRC3	PAE 13	PAE 03	RPC 13	RPC 03	RHC 13	RHC 03	DAC 13	DAC 03	CPC 13	CPC 03	0	RHD 3	0	0	DAW 13	DAW 03	FFFFFF206H	3FC1H

Bit Position	Bit Name	Function															
15, 14	PAE1n, PAE0n	DRAM On-page Access Mode Control Controls the on-page access cycle. <table> <tr> <th>PAE1n</th><th>PAE0n</th><th>Access Mode</th></tr> <tr> <td>0</td><td>0</td><td>On-page access disabled.</td></tr> <tr> <td>0</td><td>1</td><td>High-speed page DRAM</td></tr> <tr> <td>1</td><td>0</td><td>EDO DRAM</td></tr> <tr> <td>1</td><td>1</td><td>Setting prohibited</td></tr> </table>	PAE1n	PAE0n	Access Mode	0	0	On-page access disabled.	0	1	High-speed page DRAM	1	0	EDO DRAM	1	1	Setting prohibited
PAE1n	PAE0n	Access Mode															
0	0	On-page access disabled.															
0	1	High-speed page DRAM															
1	0	EDO DRAM															
1	1	Setting prohibited															
13, 12	RPC1n, RPC0n	Row Address Precharge Control Specifies the number of wait states inserted as row address precharge time. <table> <tr> <th>RPC1n</th><th>RPC0n</th><th>Number of Wait States Inserted</th></tr> <tr> <td>0</td><td>0</td><td>0</td></tr> <tr> <td>0</td><td>1</td><td>1</td></tr> <tr> <td>1</td><td>0</td><td>2</td></tr> <tr> <td>1</td><td>1</td><td>3</td></tr> </table>	RPC1n	RPC0n	Number of Wait States Inserted	0	0	0	0	1	1	1	0	2	1	1	3
RPC1n	RPC0n	Number of Wait States Inserted															
0	0	0															
0	1	1															
1	0	2															
1	1	3															

**Remark** n = 0 to 3

Bit Position	Bit Name	Function															
11, 10	RHC1n, RHC0n	<p>Row Address Hold Wait Control Specifies the number of wait states inserted as row address hold time.</p> <table border="1"> <thead> <tr> <th>RHC1n</th><th>RHC0n</th><th>Number of Wait States Inserted</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>2</td></tr> <tr><td>1</td><td>1</td><td>3</td></tr> </tbody> </table>	RHC1n	RHC0n	Number of Wait States Inserted	0	0	0	0	1	1	1	0	2	1	1	3
RHC1n	RHC0n	Number of Wait States Inserted															
0	0	0															
0	1	1															
1	0	2															
1	1	3															
9, 8	DAC1n, DAC0n	<p>Data Access Programmable Wait Control Specifies the number of wait states inserted as data access time in DRAM access.</p> <table border="1"> <thead> <tr> <th>DAC1n</th><th>DAC0n</th><th>Number of Wait States Inserted</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>2</td></tr> <tr><td>1</td><td>1</td><td>3</td></tr> </tbody> </table>	DAC1n	DAC0n	Number of Wait States Inserted	0	0	0	0	1	1	1	0	2	1	1	3
DAC1n	DAC0n	Number of Wait States Inserted															
0	0	0															
0	1	1															
1	0	2															
1	1	3															
7, 6	CPC1n, CPC0n	<p>Column Address Pre-charge Control Specifies the number of wait states inserted as column address precharge time.</p> <table border="1"> <thead> <tr> <th>CPC1n</th><th>CPC0n</th><th>Number of Wait States Inserted</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0<sup>Note</sup></td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>2</td></tr> <tr><td>1</td><td>1</td><td>3</td></tr> </tbody> </table> <p><b>Note</b> 1 wait is inserted during DRAM write access in DMA flyby transfer.</p>	CPC1n	CPC0n	Number of Wait States Inserted	0	0	0 <sup>Note</sup>	0	1	1	1	0	2	1	1	3
CPC1n	CPC0n	Number of Wait States Inserted															
0	0	0 <sup>Note</sup>															
0	1	1															
1	0	2															
1	1	3															
4	RHDn	<p>RAS Hold Disable Sets the RAS hold mode. If access to DRAM during on-page operation is not continuous, and access enters another space midway, the <math>\overline{\text{RAS}}_m</math> signal (<math>m = 3</math> to <math>5</math>) is maintained in the active state (low level) during the time the other space is being accessed in the RAS hold mode state. In this way, if access continues in the same DRAM row address following access of the other space, on-page operation can be continued. 0: RAS hold mode enabled 1: RAS hold mode disabled</p>															

**Remark** n = 0 to 3

Bit Position	Bit Name	Function															
1, 0	DAW1n, DAW0n	<p>DRAM Address Multiplex Width Control This sets the address multiplex width (refer to <b>5.3.3 Address multiplex function</b>).</p> <table border="1"> <thead> <tr> <th>DAW1n</th><th>DAW0n</th><th>Address Multiplex Width</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>8 bits</td></tr> <tr> <td>0</td><td>1</td><td>9 bits</td></tr> <tr> <td>1</td><td>0</td><td>10 bits</td></tr> <tr> <td>1</td><td>1</td><td>11 bits</td></tr> </tbody> </table>	DAW1n	DAW0n	Address Multiplex Width	0	0	8 bits	0	1	9 bits	1	0	10 bits	1	1	11 bits
DAW1n	DAW0n	Address Multiplex Width															
0	0	8 bits															
0	1	9 bits															
1	0	10 bits															
1	1	11 bits															

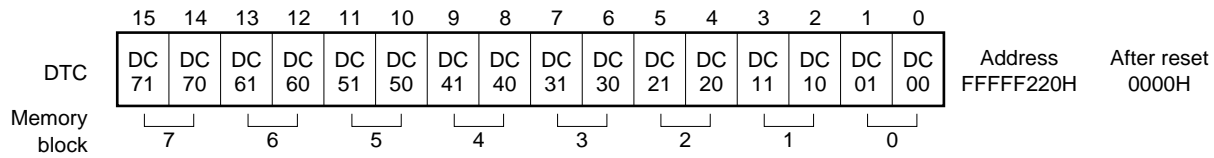
**Caution** Write to the DRCn register after reset, and then do not change the set value. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the DRCn register is complete. However, it is possible to access an external memory area whose initialization is complete.

**Remark** n = 0 to 3

### 5.3.5 DRAM type configuration register (DTC)

This controls the relationship between DRAM configuration register n (DRCn) and memory block m (n = 0 to 3, m = 0 to 7).

These registers can be read/written in 16-bit units.



Bit Position	Bit Name	Function															
15 to 0	DCm1, DCm0	<p>DRAM Type Configuration</p> <p>Specifies the DRAM configuration register n (DRCn) corresponding to memory block m. Furthermore, it has no meaning if the memory block m is not specified in the DRAM area.</p> <table border="1"> <tr> <th>DCm1</th><th>DCm0</th><th>DRAM Configuration Register n (DRCn) Corresponding to Memory Block m</th></tr> <tr> <td>0</td><td>0</td><td>DRC0</td></tr> <tr> <td>0</td><td>1</td><td>DRC1</td></tr> <tr> <td>1</td><td>0</td><td>DRC2</td></tr> <tr> <td>1</td><td>1</td><td>DRC3</td></tr> </table>	DCm1	DCm0	DRAM Configuration Register n (DRCn) Corresponding to Memory Block m	0	0	DRC0	0	1	DRC1	1	0	DRC2	1	1	DRC3
DCm1	DCm0	DRAM Configuration Register n (DRCn) Corresponding to Memory Block m															
0	0	DRC0															
0	1	DRC1															
1	0	DRC2															
1	1	DRC3															

**Caution** Write to the DTC register after reset, and then do not change the set value. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the DTC register is complete. However, it is possible to access an external memory area whose initialization is complete.

**Remark** n = 0 to 3  
m = 3 to 5

## 5.3.6 DRAM access

Figure 5-8. High-Speed Page DRAM Access Timing (1/4)

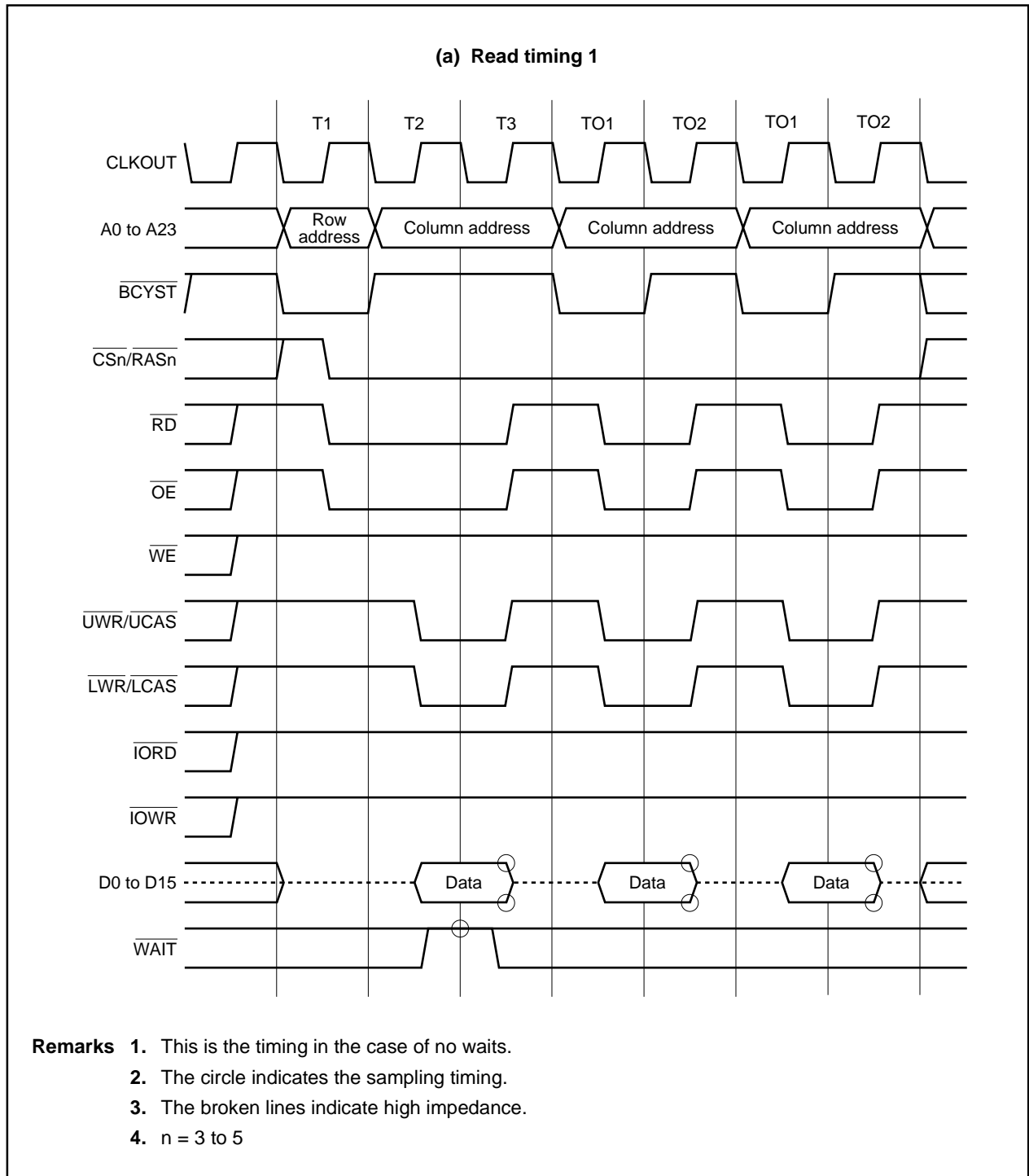


Figure 5-8. High-Speed Page DRAM Access Timing (2/4)

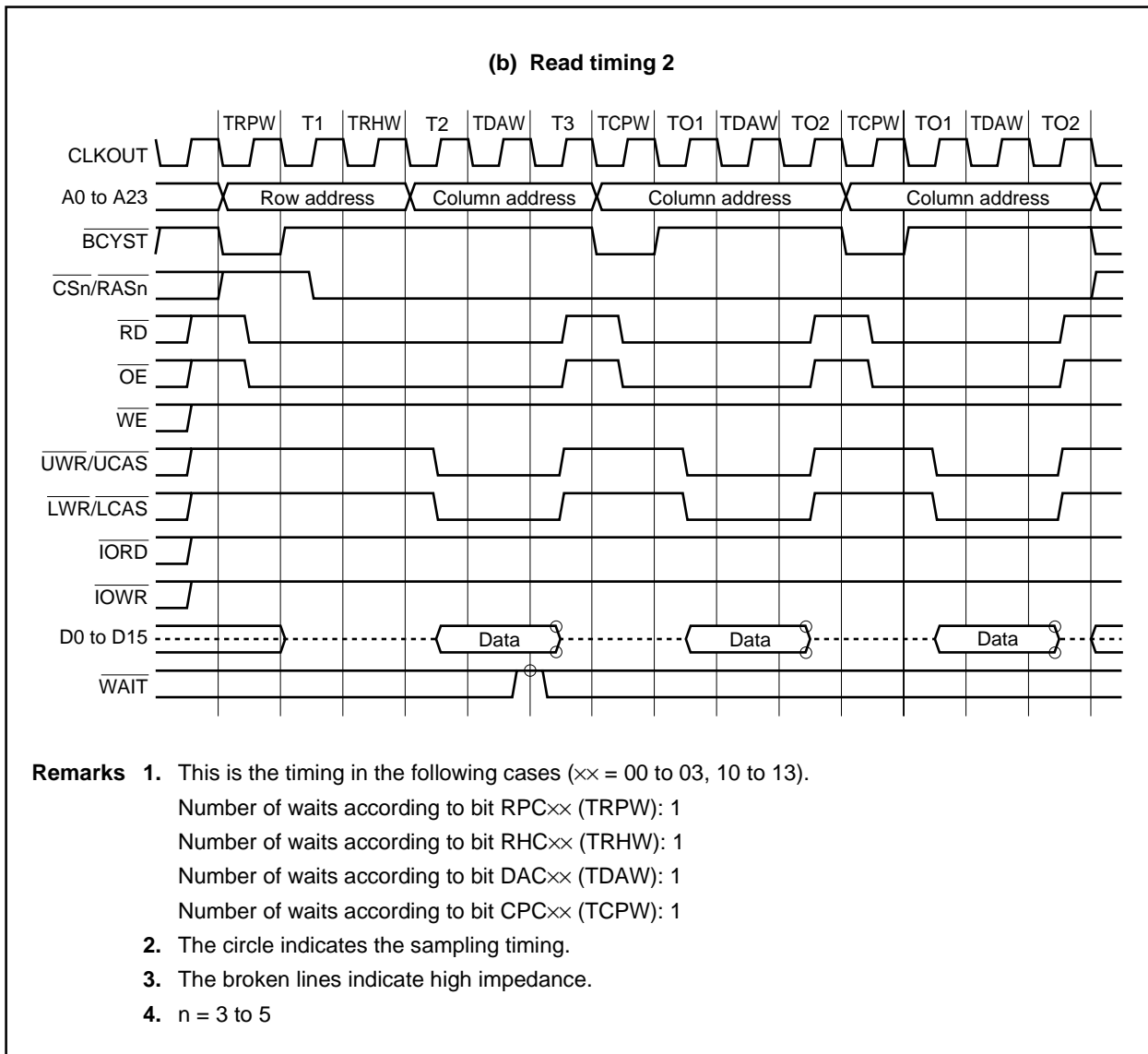




Figure 5-8. High-Speed Page DRAM Access Timing (3/4)

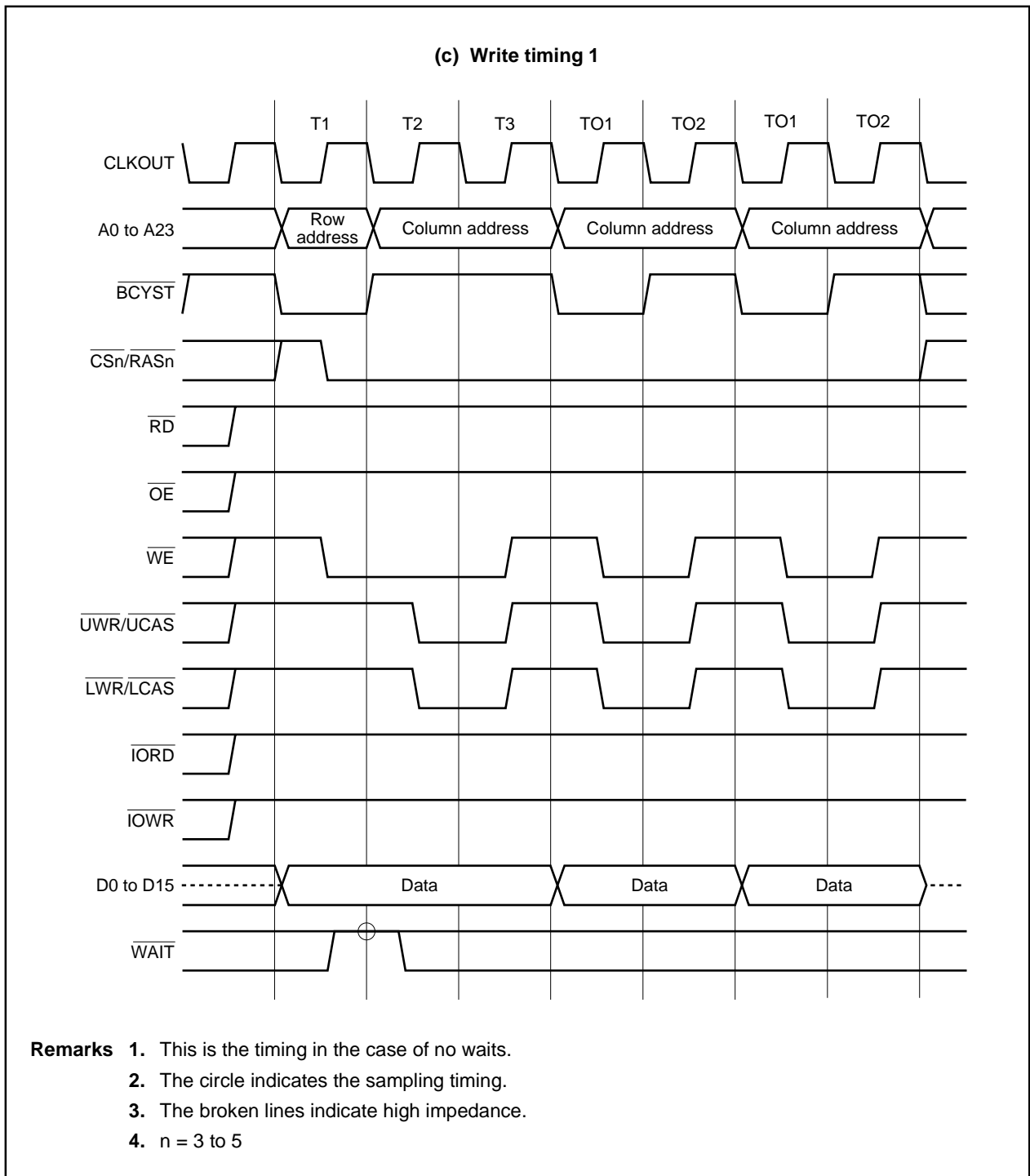


Figure 5-8. High-Speed Page DRAM Access Timing (4/4)

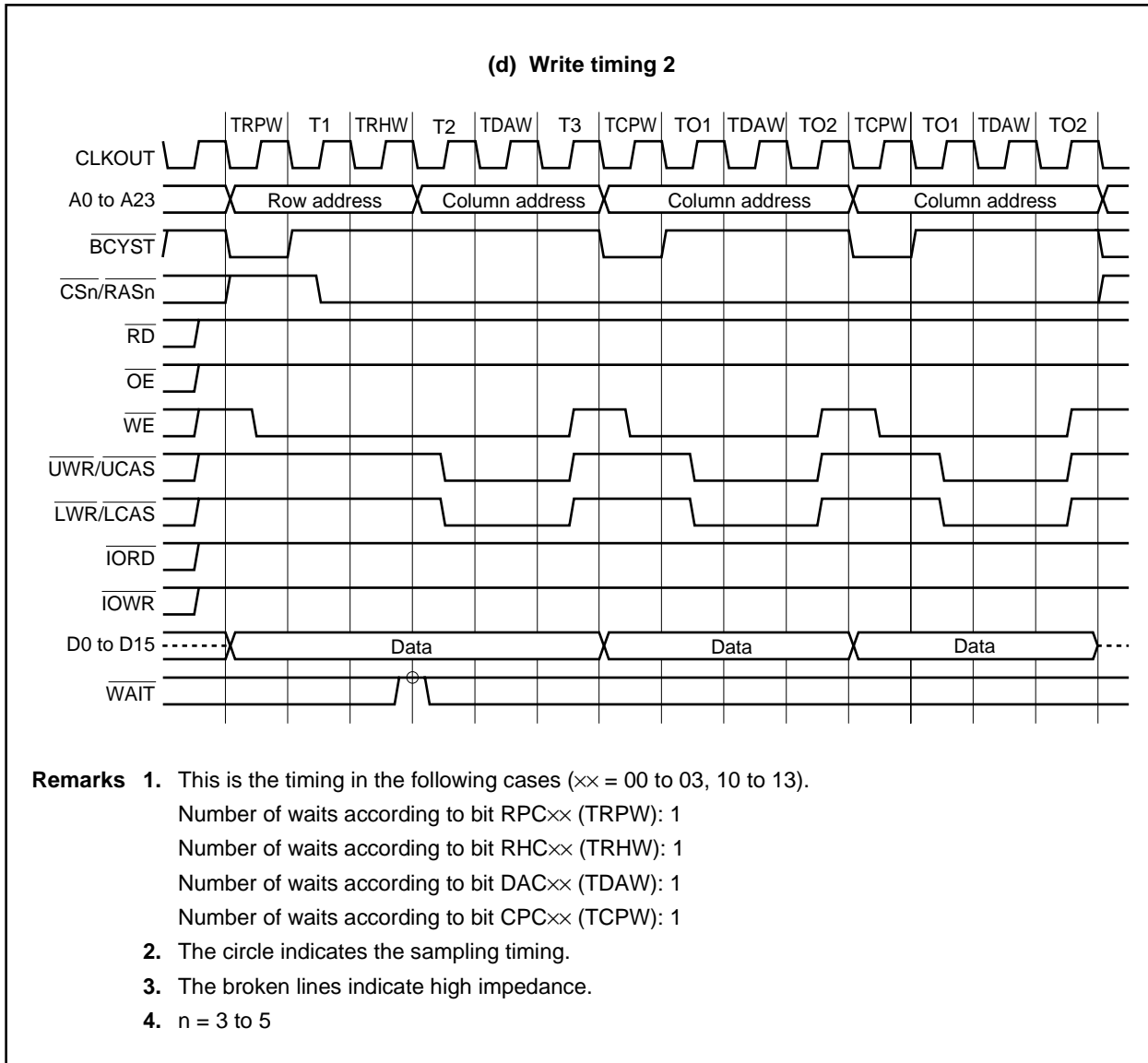


Figure 5-9. EDO DRAM Access Timing (1/4)

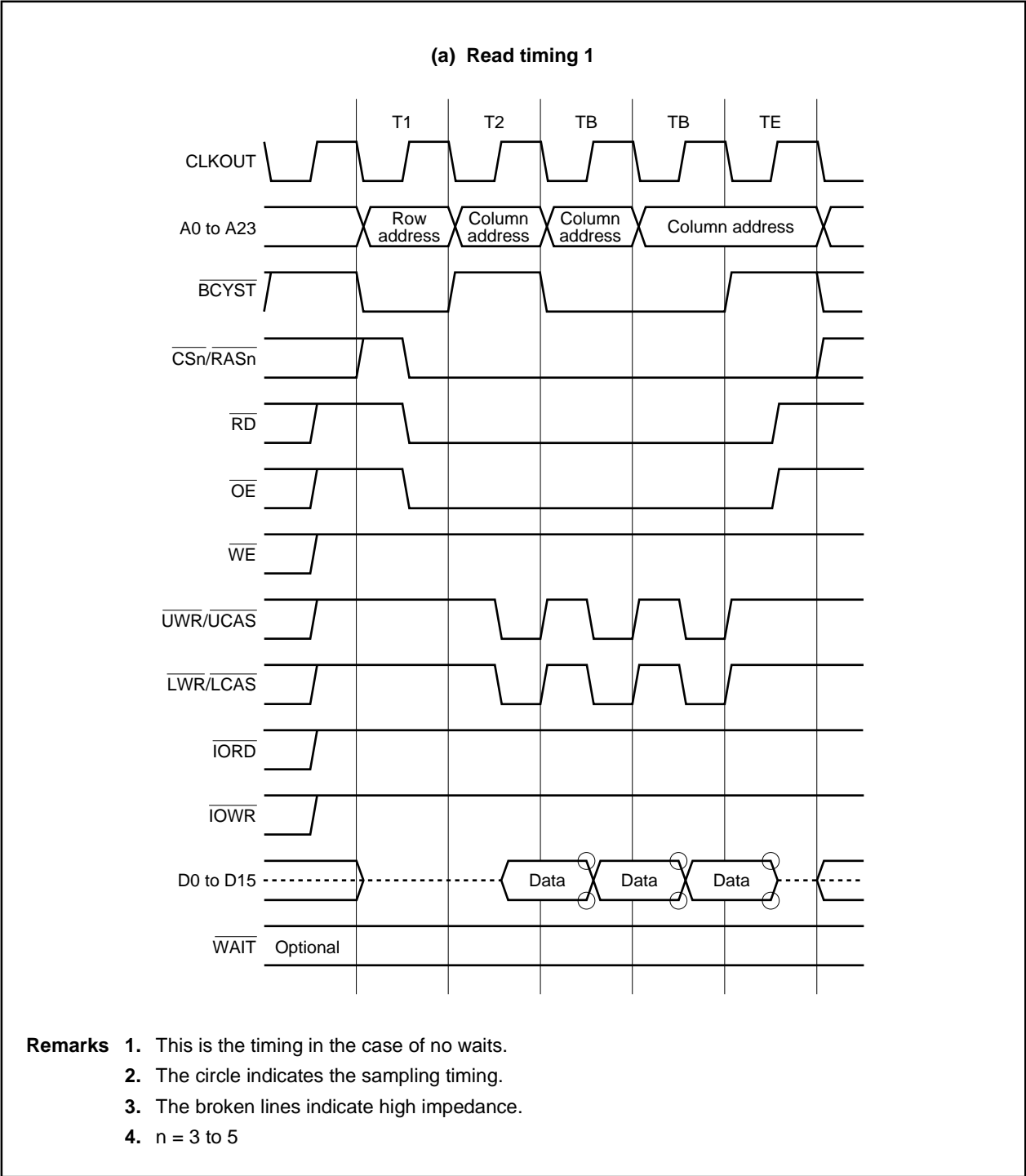
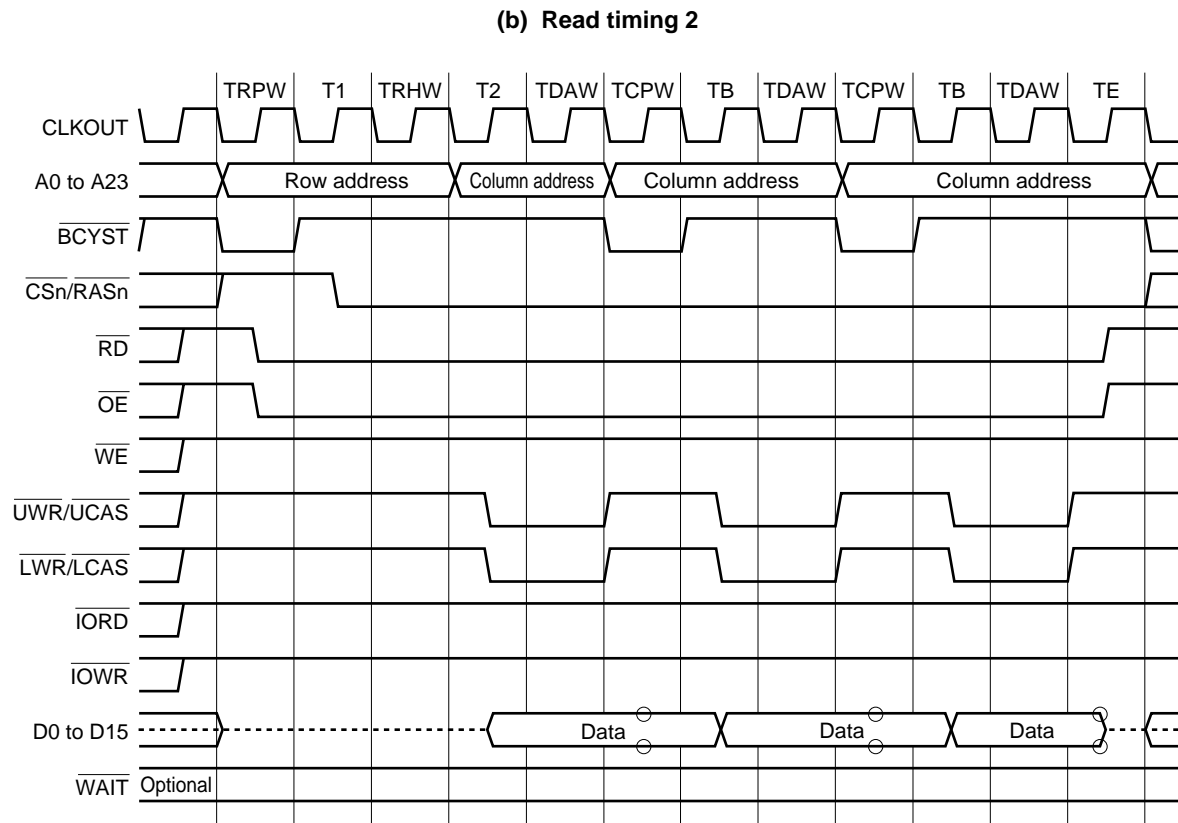


Figure 5-9. EDO DRAM Access Timing (2/4)



**Remarks 1.** This is the timing in the following cases ( $\times\times = 00$  to  $03$ ,  $10$  to  $13$ ).

Number of waits according to bit  $\text{RPC}\times\times$  (TRPW): 1

Number of waits according to bit  $\text{RHC}\times\times$  (TRHW): 1

Number of waits according to bit  $\text{DAC}\times\times$  (TDAW): 1

Number of waits according to bit  $\text{CPC}\times\times$  (TCPW): 1

**2.** The circle indicates the sampling timing.

**3.** The broken lines indicate high impedance.

**4.**  $n = 3$  to  $5$

Figure 5-9. EDO DRAM Access Timing (3/4)

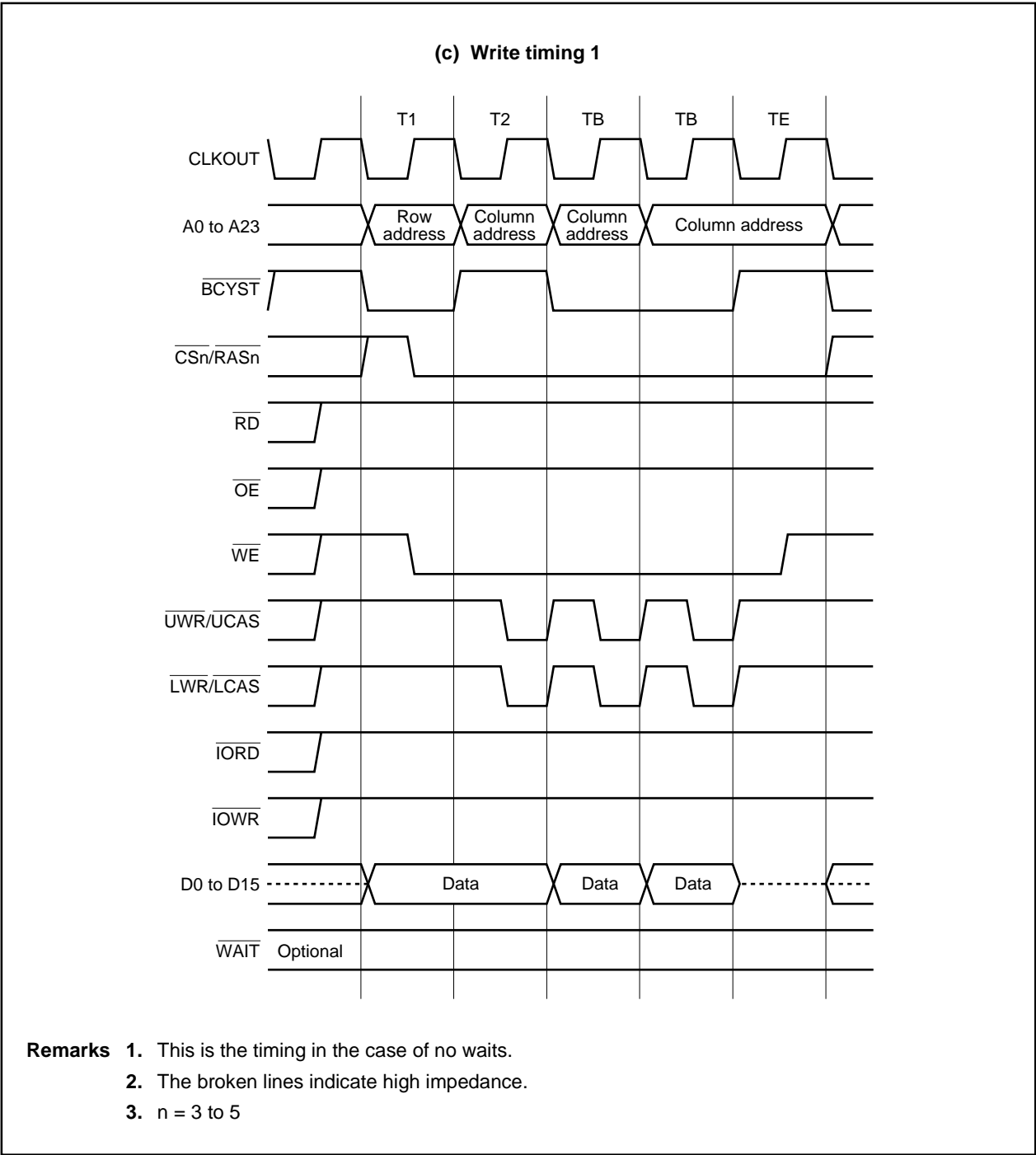
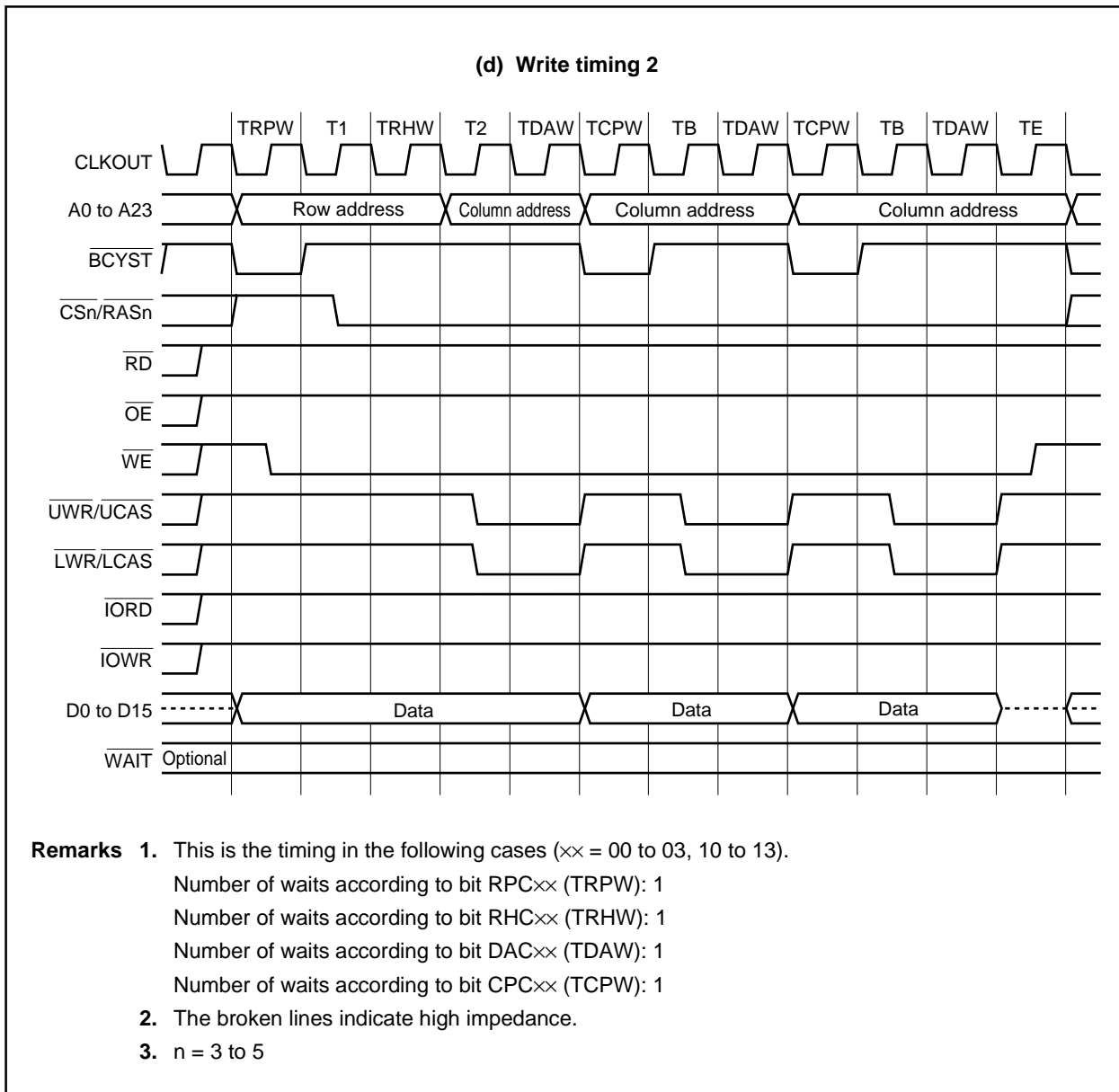


Figure 5-9. EDO DRAM Access Timing (4/4)



## 5.3.7 DRAM access during DMA flyby transfer

Figure 5-10. DRAM Access Timing During DMA Flyby Transfer (1/2)

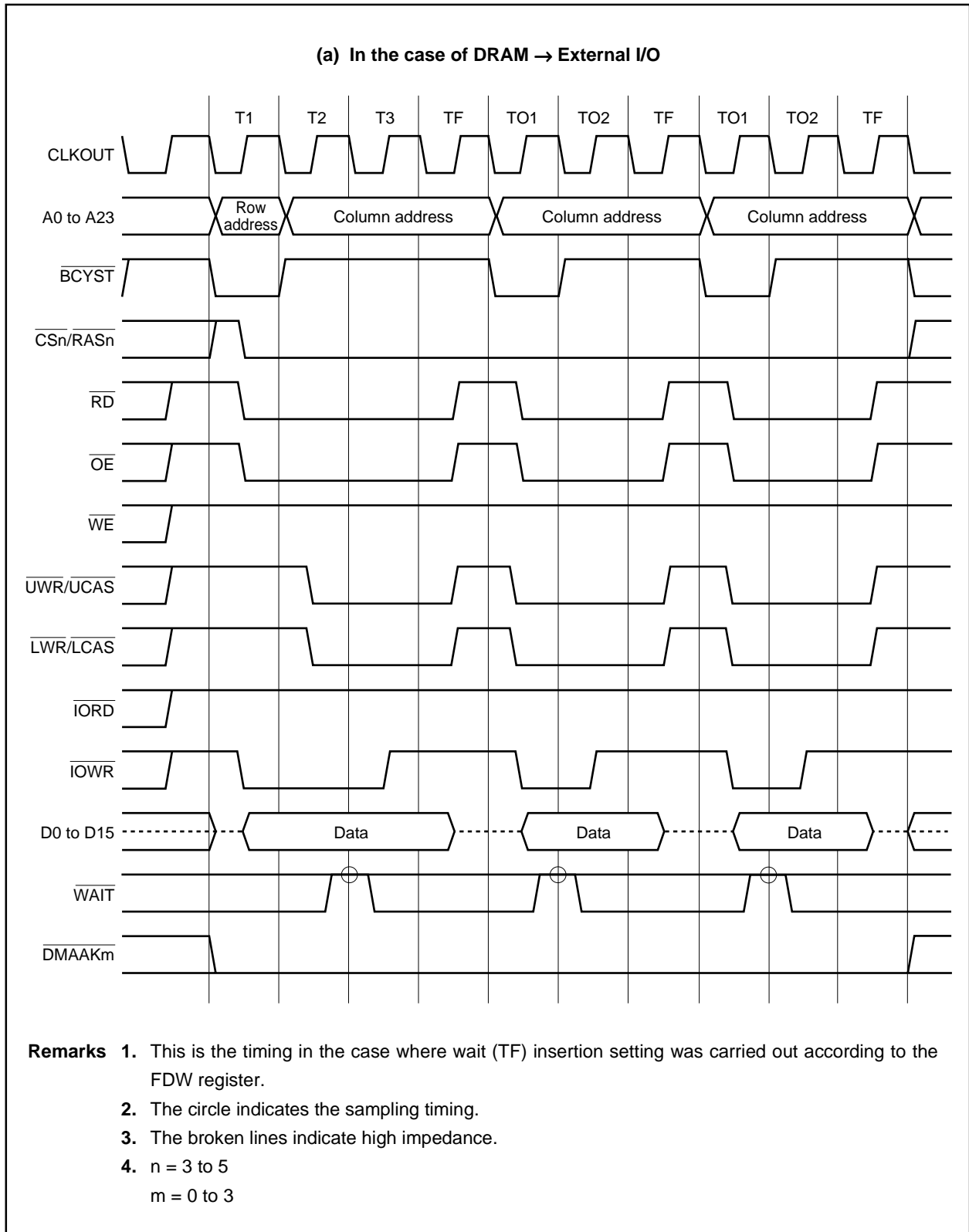
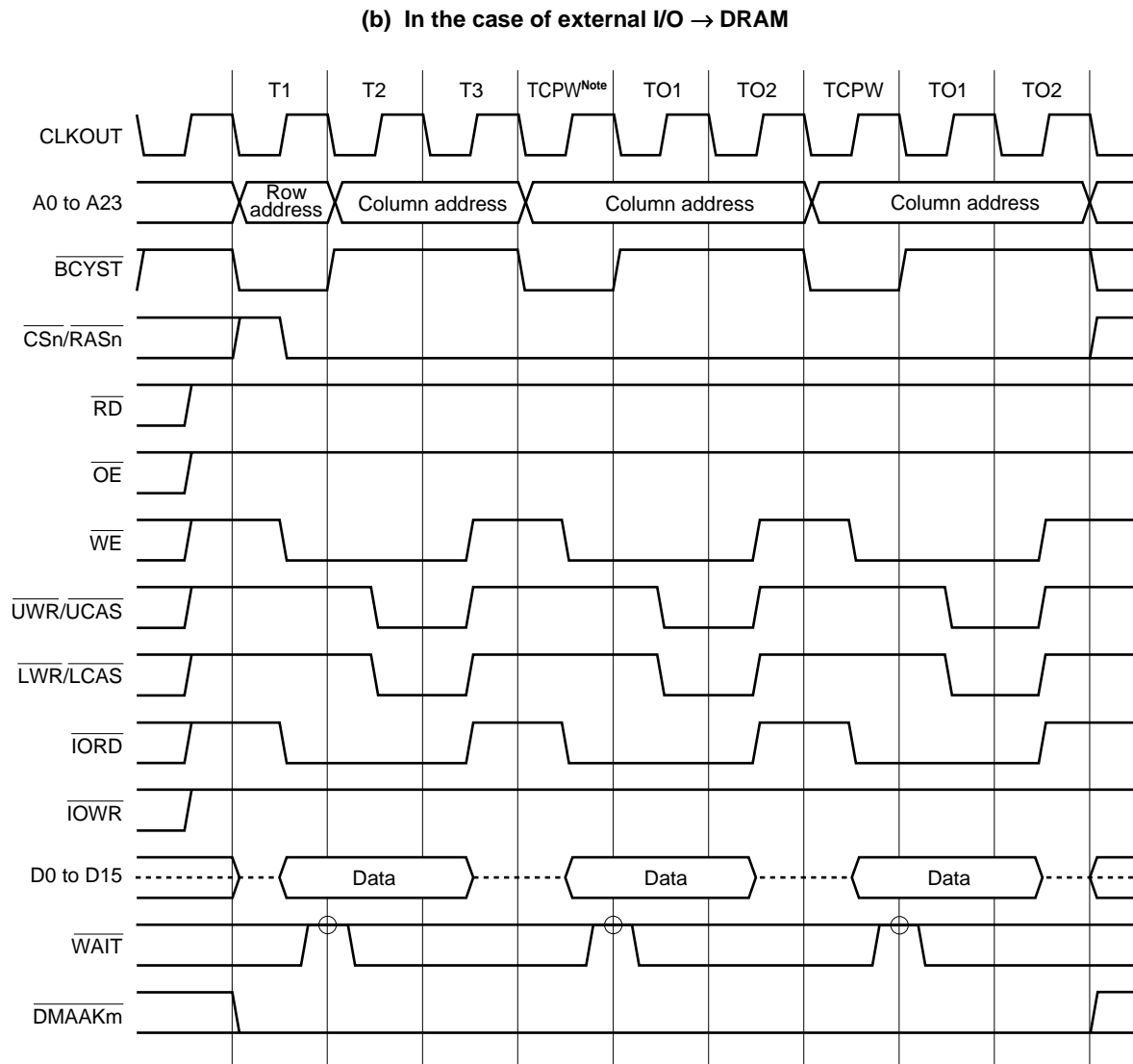


Figure 5-10. DRAM Access Timing During DMA Flyby Transfer (2/2)



**Note** A minimum of 1 clock cycle is inserted for the TCPW cycle regardless of the CPC0m and CPC1m bit settings in the DRCm register.

- Remarks**
1. This is the timing in the case where the number of waits according to the CPC $\times\times$  bit (TCPW) is 1 ( $\times\times = 00$  to  $03$ ,  $10$  to  $13$ ).
  2. In the case of external I/O → DRAM, the FDW register setting is invalid.
  3. The circle indicates the sampling timing.
  4. The broken lines indicate high impedance.
  5.  $n = 3$  to  $5$   
 $m = 0$  to  $3$



### 5.3.8 Refresh control function

V850E/MS2 can create a CBR (CAS-before-RAS) refresh cycle. The refresh cycle is set with the refresh control register (RFC).

When another bus master occupies the external bus, the DRAM controller cannot occupy the external bus. If another bus master occupies the external bus, therefore, release the bus in accordance with the refresh interval.

During the refresh interval, the address bus maintains the state it was in just before the refresh cycle.

#### (1) Refresh control registers 0 to 3 (RFC0 to RFC3)

These set whether refresh is enabled or disabled, and the refresh interval. The refresh interval is determined by the following calculation formula.

$$\text{Refresh interval } (\mu\text{s}) = \text{Refresh count clock } (T_{RCY}) \times \text{Interval factor}$$

The refresh count clock and interval factor are determined by the RENn bit and RIn bit, respectively, of the RFCn register.

Note that n corresponds to the register number (0 to 3) of DRAM configuration registers 0 to 3 (DRC0 to DRC3).

These registers can be read/written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
RFC0	REN 0	0	0	0	0	0	RCC 01	RCC 00	0	0	RI 05	RI 04	RI 03	RI 02	RI 01	RI 00	Address FFFFFF210H	After reset 0000H
RFC1	REN 1	0	0	0	0	0	RCC 11	RCC 10	0	0	RI 15	RI 14	RI 13	RI 12	RI 11	RI 10	FFFFFF212H	0000H
RFC2	REN 2	0	0	0	0	0	RCC 21	RCC 20	0	0	RI 25	RI 24	RI 23	RI 22	RI 21	RI 20	FFFFFF214H	0000H
RFC3	REN 3	0	0	0	0	0	RCC 31	RCC 30	0	0	RI 35	RI 34	RI 33	RI 32	RI 31	RI 30	FFFFFF216H	0000H

Bit Position	Bit Name	Function																																																	
15	RENn	Refresh Enable Specifies whether CBR refresh is enabled or disabled. 0: Refresh disabled 1: Refresh enabled																																																	
9, 8	RCCn1, RCCn0	Refresh Count Clock Specifies the refresh Count Clock ( $T_{RCY}$ ) <table><tr><th>RCCn1</th><th>RCCn0</th><th>Refresh Count Clock (<math>T_{RCY}</math>)</th></tr><tr><td>0</td><td>0</td><td><math>32/\phi</math></td></tr><tr><td>0</td><td>1</td><td><math>128/\phi</math></td></tr><tr><td>1</td><td>0</td><td><math>256/\phi</math></td></tr><tr><td>1</td><td>1</td><td>Setting prohibited</td></tr></table>	RCCn1	RCCn0	Refresh Count Clock ( $T_{RCY}$ )	0	0	$32/\phi$	0	1	$128/\phi$	1	0	$256/\phi$	1	1	Setting prohibited																																		
RCCn1	RCCn0	Refresh Count Clock ( $T_{RCY}$ )																																																	
0	0	$32/\phi$																																																	
0	1	$128/\phi$																																																	
1	0	$256/\phi$																																																	
1	1	Setting prohibited																																																	
5 to 0	RIn5 to RIn0	Refresh Interval Sets the interval factor of the interval timer for generation of refresh timing. <table><tr><th>RIn5</th><th>RIn4</th><th>RIn3</th><th>RIn2</th><th>RIn1</th><th>RIn0</th><th>Interval Factor</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>2</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>3</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>4</td></tr><tr><td>⋮</td><td>⋮</td><td>⋮</td><td>⋮</td><td>⋮</td><td>⋮</td><td>⋮</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>64</td></tr></table>	RIn5	RIn4	RIn3	RIn2	RIn1	RIn0	Interval Factor	0	0	0	0	0	0	1	0	0	0	0	0	1	2	0	0	0	0	1	0	3	0	0	0	0	1	1	4	⋮	⋮	⋮	⋮	⋮	⋮	⋮	1	1	1	1	1	1	64
RIn5	RIn4	RIn3	RIn2	RIn1	RIn0	Interval Factor																																													
0	0	0	0	0	0	1																																													
0	0	0	0	0	1	2																																													
0	0	0	0	1	0	3																																													
0	0	0	0	1	1	4																																													
⋮	⋮	⋮	⋮	⋮	⋮	⋮																																													
1	1	1	1	1	1	64																																													

**Caution** After refresh enable, if changing the refresh count clock or the interval factor, first clear the RENn bit (0) (refresh disable state), then perform reset.

**Remark** n = 0 to 3  
 $\phi$ : Internal system clock frequency

**Example** An example of the DRAM refresh interval and an example of setting the interval factor are shown below.

**Table 5-2. Example of DRAM Refresh Interval**

DRAM Capacity (bits)	Refresh Cycle (Cycles/ms)	Refresh Interval ( $\mu$ s)
256 K	256/4	15.6
1 M	512/8	15.6
	512/64	125
4 M	512/128	250
	1 K/16	15.6
	1 K/128	125
16 M	1 K/256	250
	2 K/256	125
	4 K/64	15.6
	4 K/256	62.5
64 M	4 K/64	15.6

**Table 5-3. Example of Interval Factor Settings**

Specified Refresh Interval Value ( $\mu$ s)	Refresh Count Clock ( $T_{RCY}$ )	Interval Factor Value <sup>Notes 1, 2</sup>		
		When $\phi = 16$ MHz	When $\phi = 20$ MHz	When $\phi = 30$ MHz
15.6	$32/\phi$	7 (14)	9 (14.4)	14 (14.9)
	$128/\phi$	1 (8)	2 (12.8)	3 (12.8)
	$256/\phi$	—	1 (12.8)	1 (8.5)
62.5	$32/\phi$	30 (60)	38 (60.8)	58 (61.9)
	$128/\phi$	7 (56)	9 (57.6)	14 (59.7)
	$256/\phi$	3 (48)	4 (51.2)	7 (59.7)
125	$32/\phi$	—	—	—
	$128/\phi$	15 (120)	19 (121.6)	29 (123.7)
	$256/\phi$	7 (112)	9 (115.2)	14 (119.5)
250	$32/\phi$	—	—	—
	$128/\phi$	31 (248)	38 (243.2)	58 (247.5)
	$256/\phi$	15 (240)	19 (243.2)	29 (247.5)

- Notes**
1. The interval factor is set by bits RIn0 to RIn5 of the RFCn register ( $n = 0$  to 3).
  2. The values in parentheses are the calculated value ( $\mu$ s) for the refresh interval.  
Refresh Interval ( $\mu$ s) = Refresh count clock ( $T_{RCY}$ )  $\times$  Interval factor

**Remark**  $\phi$ : Internal system clock frequency

**(2) Refresh wait control register (RWC)**

This specifies insertion of wait states during the refresh cycle. The register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
RWC	RRW1	RRW0	RCW2	RCW1	RCW0	SRW2	SRW1	SRW0	Address FFFFF218H	After reset 00H

Bit Position	Bit Name	Function																																				
7, 6	RRW1, RRW0	Refresh RAS Wait Control Specifies the number of wait states inserted as hold time for the $\overline{\text{RAS}}_m$ signal's high level width during CBR refresh.																																				
		<table><tr><th>RRW1</th><th>RRW0</th><th>Number of Insertion Wait States</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>2</td></tr><tr><td>1</td><td>1</td><td>3</td></tr></table>	RRW1	RRW0	Number of Insertion Wait States	0	0	0	0	1	1	1	0	2	1	1	3																					
		RRW1	RRW0	Number of Insertion Wait States																																		
		0	0	0																																		
		0	1	1																																		
		1	0	2																																		
1	1	3																																				
5 to 3	RCW2 to RCW0	Refresh Cycle Wait Control Specifies the number of wait states inserted as hold time for the $\overline{\text{RAS}}_m$ signal's low level width during CBR refresh.																																				
		<table><tr><th>RCW2</th><th>RCW1</th><th>RCW0</th><th>Number of Insertion Wait States</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>2</td></tr><tr><td>0</td><td>1</td><td>1</td><td>3</td></tr><tr><td>1</td><td>0</td><td>0</td><td>4</td></tr><tr><td>1</td><td>0</td><td>1</td><td>5</td></tr><tr><td>1</td><td>1</td><td>0</td><td>6</td></tr><tr><td>1</td><td>1</td><td>1</td><td>7</td></tr></table>	RCW2	RCW1	RCW0	Number of Insertion Wait States	0	0	0	0	0	0	1	1	0	1	0	2	0	1	1	3	1	0	0	4	1	0	1	5	1	1	0	6	1	1	1	7
		RCW2	RCW1	RCW0	Number of Insertion Wait States																																	
		0	0	0	0																																	
		0	0	1	1																																	
		0	1	0	2																																	
		0	1	1	3																																	
		1	0	0	4																																	
		1	0	1	5																																	
1	1	0	6																																			
1	1	1	7																																			
2 to 0	SRW2 to SRW0	Self-refresh Release Wait Control Specifies the number of wait states inserted as CBR self-refresh release time.																																				
		<table><tr><th>SRW2</th><th>SRW1</th><th>SRW0</th><th>Number of Insertion Wait States</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>2</td></tr><tr><td>0</td><td>1</td><td>1</td><td>3</td></tr><tr><td>1</td><td>0</td><td>0</td><td>4</td></tr><tr><td>1</td><td>0</td><td>1</td><td>5</td></tr><tr><td>1</td><td>1</td><td>0</td><td>6</td></tr><tr><td>1</td><td>1</td><td>1</td><td>7</td></tr></table>	SRW2	SRW1	SRW0	Number of Insertion Wait States	0	0	0	0	0	0	1	1	0	1	0	2	0	1	1	3	1	0	0	4	1	0	1	5	1	1	0	6	1	1	1	7
		SRW2	SRW1	SRW0	Number of Insertion Wait States																																	
		0	0	0	0																																	
		0	0	1	1																																	
		0	1	0	2																																	
		0	1	1	3																																	
		1	0	0	4																																	
		1	0	1	5																																	
1	1	0	6																																			
1	1	1	7																																			

**Caution** Write to the RWC register after reset, and then do not change the set value. Also, do not access an external memory area other than the one for this initialization routine until the initial setting of the RWC register is complete. However, it is possible to access an external memory area whose initialization is complete.

**Remark** m = 3 to 5

**Note** A minimum of 1 clock cycle is inserted for the TRCW cycle regardless of the RCW0 to RCW2 bit settings in the RWC register.

**Remarks**

1. This is the timing in the case where the number of waits (TRCW) according to the bits RCW0 to RCW2 is 1.
2.  $n = 3$  to 5

### 5.3.9 Self-refresh functions

In the case of IDLE mode and software STOP mode, the DRAM controller generates a CBR self-refresh cycle.

However, the  $\overline{\text{RASn}}$  pulse width of DRAM should meet the specifications to enter a self-refresh operation mode ( $n = 3$  to 5).

To release the self-refresh cycle, follow either of two methods below.

#### (1) Release by NMI input

##### (a) In the case of self-refresh cycle with IDLE mode

Set the  $\overline{\text{RASn}}$ ,  $\overline{\text{LCAS}}$ ,  $\overline{\text{UCAS}}$  signals to inactive (high level) immediately to release the self-refresh cycle.

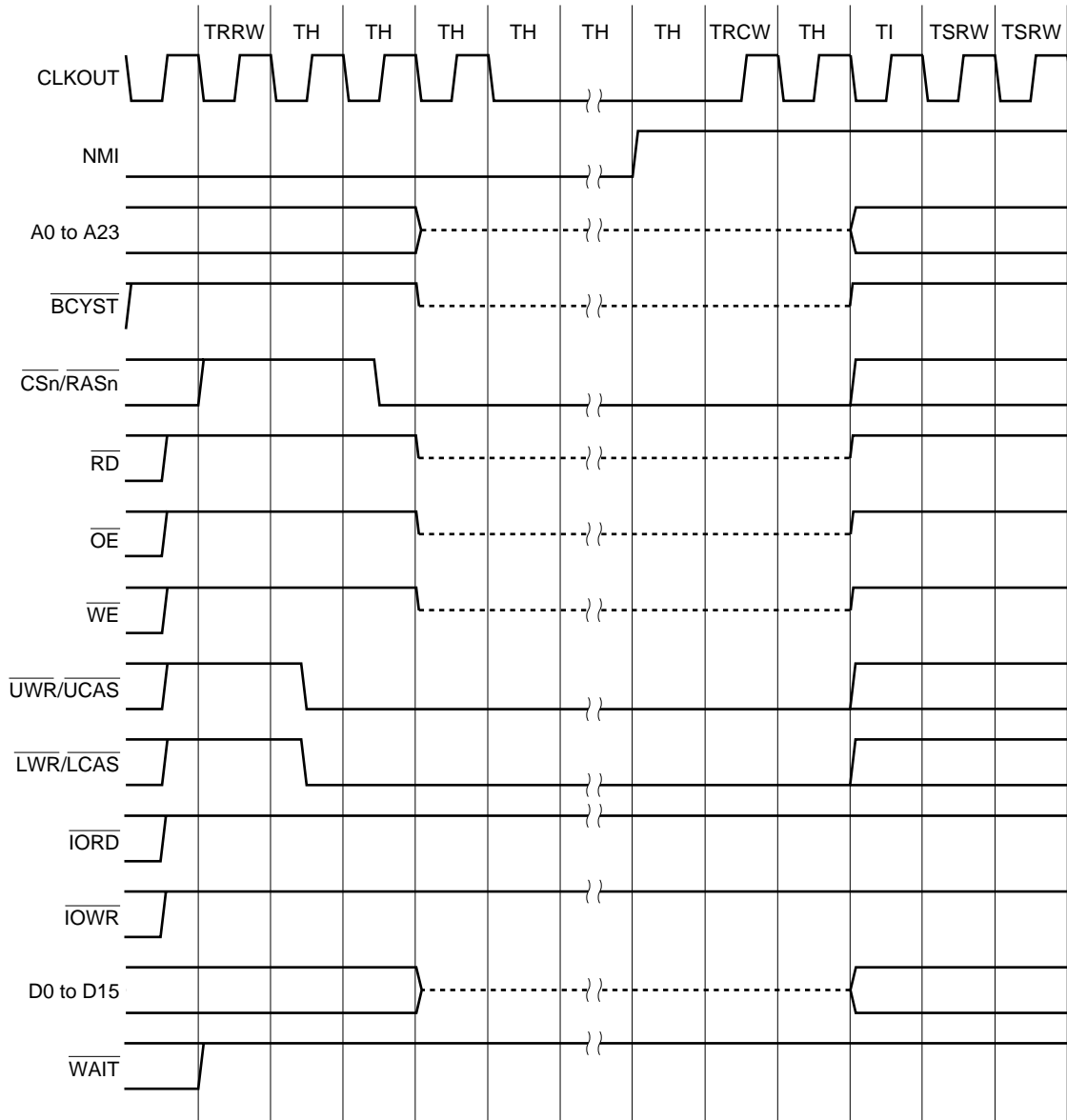
##### (b) In the case of self-refresh cycle with software STOP mode

Set the  $\overline{\text{RASn}}$ ,  $\overline{\text{LCAS}}$ ,  $\overline{\text{UCAS}}$  signals to inactive (high level) after stabilizing oscillation to release the self-refresh cycle.

#### (2) Release by $\overline{\text{RESET}}$ input

Figure 5-12. CBR Self-Refresh Timing (1/2)

(a) In the case of release according to the NMI input (in the IDLE Mode)



**Remarks 1.** This is the timing in the following cases.

Number of waits according to bits RRW0 and RRW1 (TRRW): 1

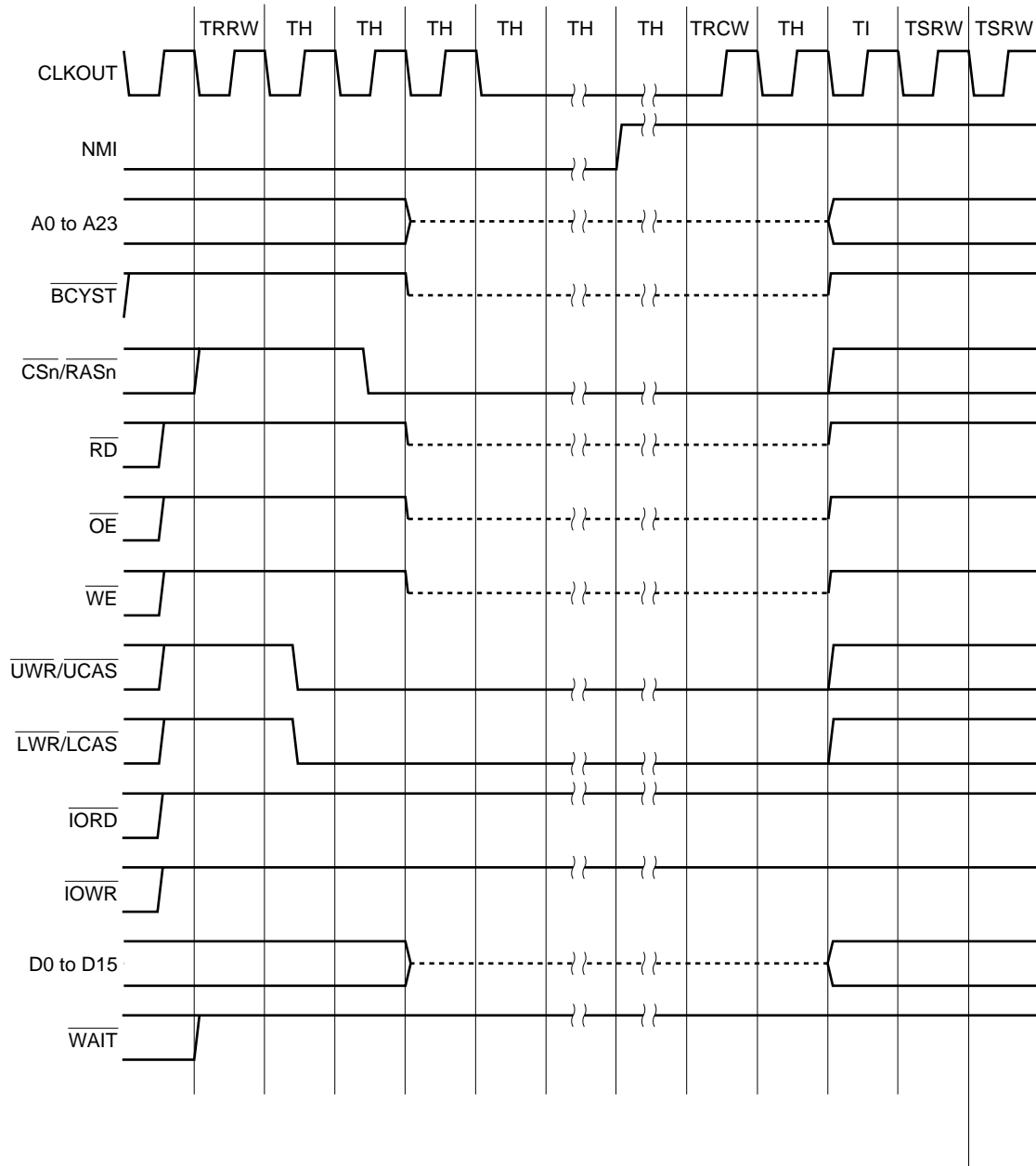
Number of waits according to bits RCW0 to RCW2 (TRCW): 1

Number of waits according to bits SRW0 to SRW2 (TSRW): 2

**2.** n = 3 to 5

Figure 5-12. CBR Self-Refresh Timing (2/2)

(b) In the case of release according to the NMI input (in the software STOP Mode)



- Remarks**
- This is the timing in the following cases.
    - Number of waits according to bits RRW0 and RRW1 (TRRW): 1
    - Number of waits according to bits RCW0 to RCW2 (TRCW): 1
    - Number of waits according to bits SRW0 to SRW2 (TSRW): 2
  - $n = 3$  to 5



## CHAPTER 6 DMA FUNCTIONS (DMA CONTROLLER)

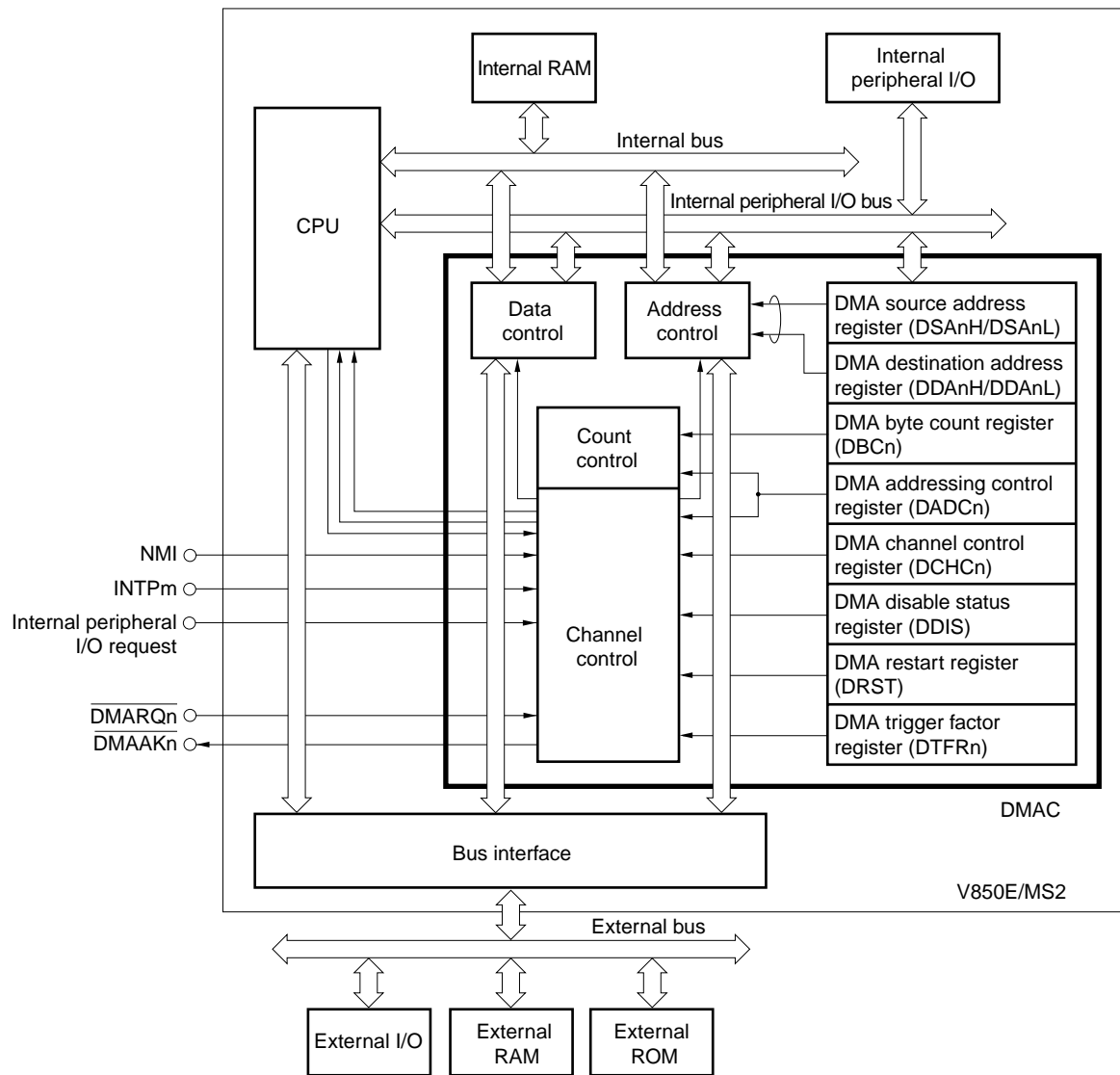
The V850E/MS2 includes a DMA (Direct Memory Access) controller (DMAC), which executes and controls DMA transfer.

The DMAC (DMA controller) transfers data between memory and I/O, or within memory, based on DMA requests issued by the internal peripheral I/O (serial interface and real-time pulse unit),  $\overline{\text{DMARQ0}}$  to  $\overline{\text{DMARQ3}}$  pins, or software triggers.

### 6.1 Features

- 4 independent DMA channels
- Transfer unit: 8/16 bits
- Maximum transfer count: 65,536 ( $2^{16}$ )
- Two types of transfer
  - Flyby (one-cycle) transfer
  - Two-cycle transfer
- Three transfer modes
  - Single transfer mode
  - Single-step transfer mode
  - Block transfer mode
- Transfer requests
  - $\overline{\text{DMARQ0}}$  to  $\overline{\text{DMARQ3}}$  pin ( $\times 4$ )
  - Requests from internal peripheral I/O (serial interface and real-time pulse unit)
  - Requests from software
- Transfer objects
  - Memory to I/O and vice versa
  - Memory to memory

## 6.2 Configuration



**Remark** m = 100 to 103, 110 to 113, 130  
n = 0 to 3

## 6.3 Control Registers

### 6.3.1 DMA source address registers 0 to 3 (DSA0 to DSA3)

These registers are used to set the DMA source addresses (26 bits each) for DMA channel  $n$  ( $n = 0$  to  $3$ ). They are divided into two 16-bit registers, DSA $n$ H and DSA $n$ L.

During DMA transfer, the registers store the next DMA source addresses.

When flyby transfer between external memory and external I/O is specified with the TTYP bits of DMA addressing control register  $n$  (DADC $n$ ), the external memory addresses are set with the DSA $n$  register. The setting made with DMA destination address register  $n$  (DDA $n$ ) is ignored.

#### (1) DMA source address registers 0H to 3H (DSA0H to DSA3H)

These registers can be read/written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DSA0H	0	0	0	0	0	0	SA 25	SA 24	SA 23	SA 22	SA 21	SA 20	SA 19	SA 18	SA 17	SA 16	Address FFFFFF1A0H	After reset Undefined
DSA1H	0	0	0	0	0	0	SA 25	SA 24	SA 23	SA 22	SA 21	SA 20	SA 19	SA 18	SA 17	SA 16	FFFFFF1A8H	Undefined
DSA2H	0	0	0	0	0	0	SA 25	SA 24	SA 23	SA 22	SA 21	SA 20	SA 19	SA 18	SA 17	SA 16	FFFFFF1B0H	Undefined
DSA3H	0	0	0	0	0	0	SA 25	SA 24	SA 23	SA 22	SA 21	SA 20	SA 19	SA 18	SA 17	SA 16	FFFFFF1B8H	Undefined

Bit Position	Bit Name	Function
9 to 0	SA25 to SA16	Source Address Sets the DMA source address (A25 to A16). During DMA transfer, it stores the next DMA source address. During flyby transfer between external memory and external I/O, it stores a memory address.

**(2) DMA source address registers 0L to 3L (DSA0L to DSA3L)**

These registers can be read/written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DSA0L	SA 15	SA 14	SA 13	SA 12	SA 11	SA 10	SA 9	SA 8	SA 7	SA 6	SA 5	SA 4	SA 3	SA 2	SA 1	SA 0	Address FFFFFF1A2H	After reset Undefined
DSA1L	SA 15	SA 14	SA 13	SA 12	SA 11	SA 10	SA 9	SA 8	SA 7	SA 6	SA 5	SA 4	SA 3	SA 2	SA 1	SA 0	FFFFFF1AAH	Undefined
DSA2L	SA 15	SA 14	SA 13	SA 12	SA 11	SA 10	SA 9	SA 8	SA 7	SA 6	SA 5	SA 4	SA 3	SA 2	SA 1	SA 0	FFFFFF1B2H	Undefined
DSA3L	SA 15	SA 14	SA 13	SA 12	SA 11	SA 10	SA 9	SA 8	SA 7	SA 6	SA 5	SA 4	SA 3	SA 2	SA 1	SA 0	FFFFFF1BAH	Undefined

Bit Position	Bit Name	Function
15 to 0	SA15 to SA0	Source Address Sets the DMA source address (A15 to A0). During DMA transfer, it stores the next DMA source address. During flyby transfer between external memory and external I/O, it stores a memory address.

**6.3.2 DMA destination address registers 0 to 3 (DDA0 to DDA3)**

These registers are used to set the DMA destination addresses (26 bits each) for DMA channel  $n$  ( $n = 0$  to  $3$ ). They are divided into two 16-bit registers, DDAnH and DDAnL.

During DMA transfer, the registers store the next DMA destination addresses.

When flyby transfer between external memory and external I/O is specified with the TTyp bits of DMA addressing control register  $n$  (DADCn), the setting of these registers are ignored. But when flyby transfer between internal RAM and internal peripheral I/O has been set, the DMA destination address registers (DDA0 to DDA3) must be set.

**(1) DMA destination address registers 0H to 3H (DDA0H to DDA3H)**

These registers can be read/written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DDA0H	0	0	0	0	0	0	DA 25	DA 24	DA 23	DA 22	DA 21	DA 20	DA 19	DA 18	DA 17	DA 16	Address FFFFFF1A4H	After reset Undefined
DDA1H	0	0	0	0	0	0	DA 25	DA 24	DA 23	DA 22	DA 21	DA 20	DA 19	DA 18	DA 17	DA 16	FFFFFF1ACH	Undefined
DDA2H	0	0	0	0	0	0	DA 25	DA 24	DA 23	DA 22	DA 21	DA 20	DA 19	DA 18	DA 17	DA 16	FFFFFF1B4H	Undefined
DDA3H	0	0	0	0	0	0	DA 25	DA 24	DA 23	DA 22	DA 21	DA 20	DA 19	DA 18	DA 17	DA 16	FFFFFF1BCH	Undefined

Bit Position	Bit Name	Function
9 to 0	DA25 to DA16	Destination Address Sets the DMA destination address (A25 to A16). During DMA transfer, it stores the next DMA destination address. This is disregarded during flyby transfer between external memory and external I/O, but be sure to set this register during flyby transfer between internal RAM and internal peripheral I/O.

**(2) DMA destination address registers 0L to 3L (DDA0L to DDA3L)**

These registers can be read/written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DDA0L	DA 15	DA 14	DA 13	DA 12	DA 11	DA 10	DA 9	DA 8	DA 7	DA 6	DA 5	DA 4	DA 3	DA 2	DA 1	DA 0	Address FFFFFF1A6H	After reset Undefined
DDA1L	DA 15	DA 14	DA 13	DA 12	DA 11	DA 10	DA 9	DA 8	DA 7	DA 6	DA 5	DA 4	DA 3	DA 2	DA 1	DA 0	FFFFFF1AEH	Undefined
DDA2L	DA 15	DA 14	DA 13	DA 12	DA 11	DA 10	DA 9	DA 8	DA 7	DA 6	DA 5	DA 4	DA 3	DA 2	DA 1	DA 0	FFFFFF1B6H	Undefined
DDA3L	DA 15	DA 14	DA 13	DA 12	DA 11	DA 10	DA 9	DA 8	DA 7	DA 6	DA 5	DA 4	DA 3	DA 2	DA 1	DA 0	FFFFFF1BEH	Undefined

Bit Position	Bit Name	Function
15 to 0	DA15 to DA0	<b>Destination Address</b> Sets the DMA destination address (A15 to A0). During DMA transfer, it stores the next DMA destination address. This is disregarded during flyby transfer between external memory and external I/O, but be sure to set this register during flyby transfer between internal RAM and internal peripheral I/O.

### 6.3.3 DMA byte count registers 0 to 3 (DBC0 to DBC3)

These 16-bit registers are used to set the byte transfer counts for DMA channel n (n = 0 to 3).

They store the remaining transfer counts during DMA transfer.

These registers are decremented by 1 for byte transfer and by two for 16-bit transfer. Transfer ends when a borrow occurs. Thus, “transfer count –1” should be set for byte transfer and “(transfer count –1) × 2” for 16-bit transfer.

These registers can be read/written in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DBC0	BC 15	BC 14	BC 13	BC 12	BC 11	BC 10	BC 9	BC 8	BC 7	BC 6	BC 5	BC 4	BC 3	BC 2	BC 1	BC 0	Address FFFFFF1E0H	After reset Undefined
DBC1	BC 15	BC 14	BC 13	BC 12	BC 11	BC 10	BC 9	BC 8	BC 7	BC 6	BC 5	BC 4	BC 3	BC 2	BC 1	BC 0	FFFFFF1E2H	Undefined
DBC2	BC 15	BC 14	BC 13	BC 12	BC 11	BC 10	BC 9	BC 8	BC 7	BC 6	BC 5	BC 4	BC 3	BC 2	BC 1	BC 0	FFFFFF1E4H	Undefined
DBC3	BC 15	BC 14	BC 13	BC 12	BC 11	BC 10	BC 9	BC 8	BC 7	BC 6	BC 5	BC 4	BC 3	BC 2	BC 1	BC 0	FFFFFF1E6H	Undefined

Bit Position	Bit Name	Function									
15 to 0	BC15 to BC0	Byte Count Sets the byte transfer count. During DMA transfer, it stores the remaining byte transfer count.									
		DBCn	States	0000H	Byte transfer count 1 or the remaining byte transfer count	0001H	Byte transfer count 2 or the remaining byte transfer count	:	:	FFFFH	Byte transfer count 65,536 (2 <sup>16</sup> ) or the remaining byte transfer count
		DBCn	States								
		0000H	Byte transfer count 1 or the remaining byte transfer count								
		0001H	Byte transfer count 2 or the remaining byte transfer count								
		:	:								
		FFFFH	Byte transfer count 65,536 (2 <sup>16</sup> ) or the remaining byte transfer count								

**Remark** n = 0 to 3

### 6.3.4 DMA addressing control registers 0 to 3 (DADC0 to DADC3)

These 16-bit registers are used to control the DMA transfer operation modes for DMA channel n (n = 0 to 3).

These registers can be read/written in 16-bit units.

**Caution** During DMA transfer, do not perform writing to these registers.

(1/2)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DADC0	0	0	0	0	0	0	0	DS	SAD <sub>1</sub>	SAD <sub>0</sub>	DAD <sub>1</sub>	DAD <sub>0</sub>	TM <sub>1</sub>	TM <sub>0</sub>	TTYP	TDIR	Address FFFFFF1F0H	After reset 0000H
DADC1	0	0	0	0	0	0	0	DS	SAD <sub>1</sub>	SAD <sub>0</sub>	DAD <sub>1</sub>	DAD <sub>0</sub>	TM <sub>1</sub>	TM <sub>0</sub>	TTYP	TDIR	FFFFFF1F2H	0000H
DADC2	0	0	0	0	0	0	0	DS	SAD <sub>1</sub>	SAD <sub>0</sub>	DAD <sub>1</sub>	DAD <sub>0</sub>	TM <sub>1</sub>	TM <sub>0</sub>	TTYP	TDIR	FFFFFF1F4H	0000H
DADC3	0	0	0	0	0	0	0	DS	SAD <sub>1</sub>	SAD <sub>0</sub>	DAD <sub>1</sub>	DAD <sub>0</sub>	TM <sub>1</sub>	TM <sub>0</sub>	TTYP	TDIR	FFFFFF1F6H	0000H

Bit Position	Bit Name	Function															
8	DS	Data Size Sets the transfer data size for DMA transfer. 0: 8 bits 1: 16 bits															
7, 6	SAD1, SAD0	Source Address count Direction Sets the count direction of the source address for DMA channel n. <table border="1"> <tr> <th>SAD1</th><th>SAD0</th><th>Count Direction</th></tr> <tr> <td>0</td><td>0</td><td>Increment</td></tr> <tr> <td>0</td><td>1</td><td>Decrement</td></tr> <tr> <td>1</td><td>0</td><td>Fixed</td></tr> <tr> <td>1</td><td>1</td><td>Setting prohibited</td></tr> </table>	SAD1	SAD0	Count Direction	0	0	Increment	0	1	Decrement	1	0	Fixed	1	1	Setting prohibited
SAD1	SAD0	Count Direction															
0	0	Increment															
0	1	Decrement															
1	0	Fixed															
1	1	Setting prohibited															

**Remark** n = 0 to 3



Bit Position	Bit Name	Function															
5, 4	DAD1, DAD0	Destination Address count Direction Sets the count direction of the destination address for DMA channel n. <table border="1"> <thead> <tr> <th>DAD1</th><th>DAD0</th><th>Count Direction</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>Increment</td></tr> <tr> <td>0</td><td>1</td><td>Decrement</td></tr> <tr> <td>1</td><td>0</td><td>Fixed</td></tr> <tr> <td>1</td><td>1</td><td>Setting prohibited</td></tr> </tbody> </table>	DAD1	DAD0	Count Direction	0	0	Increment	0	1	Decrement	1	0	Fixed	1	1	Setting prohibited
DAD1	DAD0	Count Direction															
0	0	Increment															
0	1	Decrement															
1	0	Fixed															
1	1	Setting prohibited															
3, 2	TM1, TM0	Transfer Mode Sets the transfer mode during DMA transfer. <table border="1"> <thead> <tr> <th>TM1</th><th>TM0</th><th>Transfer Mode</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>Single transfer mode</td></tr> <tr> <td>0</td><td>1</td><td>Single-step transfer mode</td></tr> <tr> <td>1</td><td>0</td><td>Block transfer mode</td></tr> <tr> <td>1</td><td>1</td><td>Setting prohibited</td></tr> </tbody> </table>	TM1	TM0	Transfer Mode	0	0	Single transfer mode	0	1	Single-step transfer mode	1	0	Block transfer mode	1	1	Setting prohibited
TM1	TM0	Transfer Mode															
0	0	Single transfer mode															
0	1	Single-step transfer mode															
1	0	Block transfer mode															
1	1	Setting prohibited															
1	TTYP	Transfer Type Sets the DMA transfer type. 0: Two-cycle transfer 1: Flyby transfer															
0	TDIR	Transfer Direction Sets the transfer direction during transfer between I/O and memory. The setting is valid during flyby transfer only and ignored during two-cycle transfer. 0: Memory → I/O (read) 1: I/O → memory (write)															

**Remark** n = 0 to 3

**6.3.5 DMA channel control registers 0 to 3 (DCHC0 to DCHC3)**

These 8-bit registers are used to control the DMA transfer operation mode for DMA channel  $n$  ( $n = 0$  to  $3$ ).

These registers can be read/written in 8-bit units. (However, bit 7 is read-only and bits 2 and 1 are write-only. When the DMA channel control registers are read, bits 2 and 1 are always 0.)

	7	6	5	4	3	2	1	0		
DCHC0	TC0	0	0	0	0	INIT0	STG0	EN0	Address FFFFF5F0H	After reset 00H
DCHC1	TC1	0	0	0	0	INIT1	STG1	EN1	FFFFF5F2H	00H
DCHC2	TC2	0	0	0	0	INIT2	STG2	EN2	FFFFF5F4H	00H
DCHC3	TC3	0	0	0	0	INIT3	STG3	EN3	FFFFF5F6H	00H

Bit Position	Bit Name	Function
7	TC $n$	Terminal Count This status bit indicates whether DMA transfer through DMA channel $n$ has ended or not. This bit can only be read. It is set (1) when DMA transfer ends with a terminal count and reset (0) when it is read. 0: DMA transfer has not ended. 1: DMA transfer has ended.
2	INIT $n$	Initialize If this bit is set (1), the DMA transfer is forcibly terminated.
1	STG $n$	Software Trigger In DMA transfer enable state (TC $n$ bit = 0, EN $n$ bit = 1), if this bit is set (1), DMA transfer can be started by software.
0	EN $n$	Enable Specifies whether DMA transfer through DMA channel $n$ is to be enabled or disabled. It is reset (0) when DMA transfer ends with a terminal count. It is also reset (0) when transfer is forcibly ended by means of setting (1) NMI input or INIT $n$ bit. 0: DMA transfer disabled. 1: DMA transfer enabled.

**Remark**  $n = 0$  to  $3$

**6.3.6 DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)**

These 8-bit registers are used to control the DMA transfer start trigger through interrupt requests from peripheral I/O.

The interrupt requests that are set with these registers start DMA transfer.

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

(1/2)

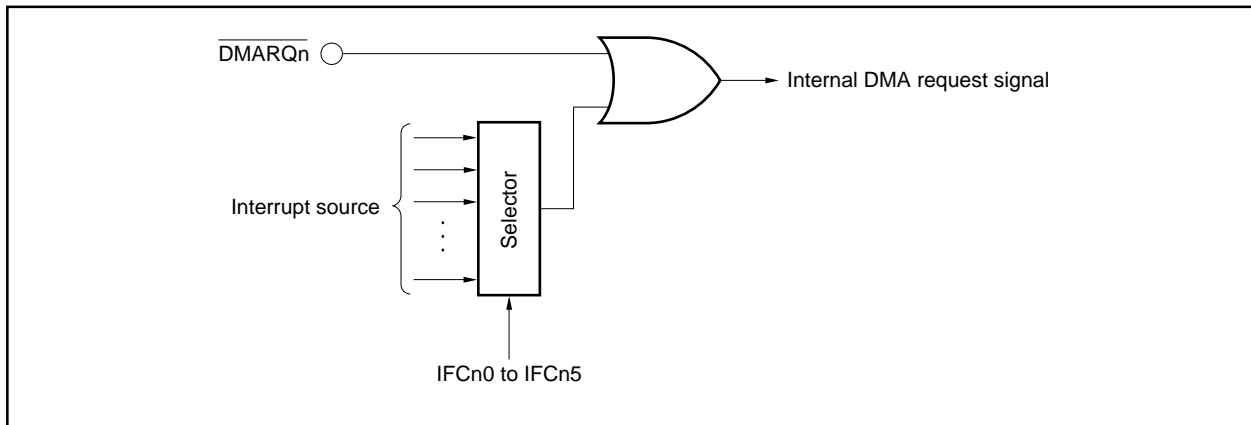
	7	6	5	4	3	2	1	0		
DTFR0	0	0	IFC05	IFC04	IFC03	IFC02	IFC01	IFC00	Address FFFFF5E0H	After reset 00H
DTFR1	0	0	IFC15	IFC14	IFC13	IFC12	IFC11	IFC10	FFFFF5E2H	00H
DTFR2	0	0	IFC25	IFC24	IFC23	IFC22	IFC21	IFC20	FFFFF5E4H	00H
DTFR3	0	0	IFC35	IFC34	IFC33	IFC32	IFC31	IFC30	FFFFF5E6H	00H

Bit Position	Bit Name	Function																																																																																																																													
5 to 0	IFCn5 to IFCn0	Interrupt Factor Code This code indicates the source of the DMA transfer trigger.																																																																																																																													
		IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt Source	0	0	0	0	0	0	DMA request from internal peripheral I/O disabled.	0	0	0	0	0	1	INTCM40	0	0	0	0	1	0	INTCM41	0	0	0	0	1	1	INTCSI0	0	0	0	1	0	0	INTSR0	0	0	0	1	0	1	INTST0	0	0	0	1	1	0	INTCSI1	0	0	0	1	1	1	INTSR1	0	0	1	0	0	0	INTST1	0	0	1	0	1	1	INTP100/INTCC100	0	0	1	1	0	0	INTP101/INTCC101	0	0	1	1	0	1	INTP102/INTCC102	0	0	1	1	1	0	INTP103/INTCC103	0	0	1	1	1	1	INTP110/INTCC110	0	1	0	0	0	0	INTP111/INTCC111	0	1	0	0	0	1	INTP112/INTCC112	0	1	0	0	1	0	INTP113/INTCC113
		IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt Source																																																																																																																							
		0	0	0	0	0	0	DMA request from internal peripheral I/O disabled.																																																																																																																							
		0	0	0	0	0	1	INTCM40																																																																																																																							
		0	0	0	0	1	0	INTCM41																																																																																																																							
		0	0	0	0	1	1	INTCSI0																																																																																																																							
		0	0	0	1	0	0	INTSR0																																																																																																																							
		0	0	0	1	0	1	INTST0																																																																																																																							
		0	0	0	1	1	0	INTCSI1																																																																																																																							
		0	0	0	1	1	1	INTSR1																																																																																																																							
		0	0	1	0	0	0	INTST1																																																																																																																							
		0	0	1	0	1	1	INTP100/INTCC100																																																																																																																							
		0	0	1	1	0	0	INTP101/INTCC101																																																																																																																							
		0	0	1	1	0	1	INTP102/INTCC102																																																																																																																							
		0	0	1	1	1	0	INTP103/INTCC103																																																																																																																							
		0	0	1	1	1	1	INTP110/INTCC110																																																																																																																							
		0	1	0	0	0	0	INTP111/INTCC111																																																																																																																							
		0	1	0	0	0	1	INTP112/INTCC112																																																																																																																							
		0	1	0	0	1	0	INTP113/INTCC113																																																																																																																							

Bit Position	Bit Name	Function						
5 to 0	IFCn5 to IFCn0							
		IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt Source
		0	1	0	0	1	1	INTCC120
		0	1	0	1	0	0	INTCC121
		0	1	0	1	0	1	INTCC122
		0	1	0	1	1	0	INTCC123
		0	1	0	1	1	1	INTP130/INTCC130
		0	1	1	0	0	0	INTCC131
		0	1	1	0	0	1	INTCC132
		0	1	1	0	1	0	INTCC133
		1	0	0	0	1	1	INTAD
		Other than above						Setting prohibited

**Remark** n = 0 to 3

**Remark** The relationship between the  $\overline{\text{DMARQn}}$  signal and the interrupt source which becomes the DMA transfer start trigger is as follows (n = 0 to 3).



### 6.3.7 DMA disable status register (DDIS)

This register holds the contents of the ENn bit of the DCHCn register during NMI input (n = 0 to 3). It is read-only, in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
DDIS	0	0	0	0	CH3	CH2	CH1	CH0	Address FFFFF5D0H	After reset 00H

Bit Position	Bit Name	Function
3 to 0	CHn (n = 3 to 0)	NMI Interruption Status Reflects the contents of the ENn bit of the DCHCn register during NMI input. The contents of this register are held until the next NMI input or until the next system reset.

### 6.3.8 DMA restart register (DRST)

This register is used to restart DMA transfer that was forcibly interrupted during NMI input. The RENn bit of this register and the ENn bit of the DCHCn register are linked to each other (n = 0 to 3). After NMI is completed, the DDIS register is referred to and the DMA channel that was interrupted is confirmed, then by setting the RENn bit in the corresponding channel (1), DMA transfer can be restarted. The register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
DRST	0	0	0	0	REN3	REN2	REN1	REN0	Address FFFFF5D2H	After reset 00H

Bit Position	Bit Name	Function
3 to 0	RENn (n = 3 to 0)	Restart Enable This sets DMA transfer enable/disable in DMA channel n. If DMA transfer is completed in accordance with the terminal count, it is reset (0). It is also reset (0) when DMA is forcibly terminated by NMI input or by setting of the INITn bit (1) in the DCHCn register. 0: DMA transfer disabled. 1: DMA transfer enabled.

### 6.3.9 Flyby transfer data wait control register (FDW)

To prevent illegal writing during flyby transfer, this register sets the insertion of wait states (TF) for securing the time from when the write signal ( $\overline{\text{IOWR}}$ ,  $\overline{\text{UWR}}$ ,  $\overline{\text{LWR}}$ ,  $\overline{\text{WE}}$ ) becomes inactive until the read signal ( $\overline{\text{RD}}$ ,  $\overline{\text{IORD}}$ ,  $\overline{\text{OE}}$ ) becomes inactive. This register can be read/written in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
FDW	FDW7	FDW6	FDW5	FDW4	FDW3	FDW2	FDW1	FDW0	Address FFFF06CH	After reset 00H
Memory Block	7	6	5	4	3	2	1	0		

Bit Position	Bit Name	Function
7 to 0	FDWn (n = 7 to 0)	Flyby Data Wait Sets wait state insertion for memory block n. 0: Wait state not inserted. 1: Wait state inserted.

**Caution** Write to the FDW register after reset, and then do not change the value. Also, do not access an external memory area until the initial setting of the FDW register is complete. (However, the memory area 0000000H to 01FFFFFFH is excluded.)

**Remark** Setting of the FDW register is valid during the DMA transfers shown below.

Type of Memory Object of Transfer	SRAM, Page ROM	DRAM
Memory → I/O	Valid	Valid
I/O → Memory	Valid	Invalid

## 6.4 DMA Bus States

### 6.4.1 Types of bus states

The DMAC bus cycle consists of the following 25 states:

**(1) TI state**

The TI state is idle state, during which no access request is issued.

The  $\overline{\text{DMARQ0}}$  to  $\overline{\text{DMARQ3}}$  signals are sampled at the falling edge of the CLKOUT signal.

**(2) T0 state**

DMA transfer ready state. (A DMA transfer request has been issued, causing bus mastership to be acquired for the first DMA transfer).

**(3) T1R state**

The bus enters the T1R state at the beginning of a read operation in two-cycle transfer mode. Address driving starts. After entering the T1R state, the bus invariably enters the T2R state.

**(4) T1RI state**

T1RI is a state in which the bus is waiting for the acknowledge in response to an external memory read request. After entering the last T1RI state, the bus invariably enters the T2R state.

**(5) T2R state**

The T2R state corresponds to the last state of a read operation in two-cycle transfer mode, or to a wait state. In the last T2R state, read data is sampled. After entering the last T2R state, the bus invariably enters the T1W state.

**(6) T2RI state**

Internal peripheral I/O or internal RAM DMA transfer ready state (Bus mastership is acquired for DMA transfer to internal peripheral I/O or internal RAM). After entering the last T2RI state, the bus invariably enters the T1W state.

**(7) T1W state**

The bus enters the T1W state at the beginning of a write operation in two-cycle transfer mode. Address driving starts. After entering the T1W state, the bus invariably enters the T2W state.

**(8) T1WI state**

T1WI is a state in which the bus is waiting for the acknowledge signal in response to an external memory write request. After entering the last T1WI state, the bus invariably enters the T2W state.

**(9) T2W state**

The T2W state corresponds to the last state of a write operation in two-cycle transfer mode, or to a wait state. In the last T2W state, the write strobe signal is made inactive.

**(10) T1F state**

The bus enters the T1F state at the beginning of a flyby transfer from internal peripheral I/O to internal RAM. The read cycle from internal peripheral I/O is started. After entering the T1F state, the bus invariably enters the T2F state.

**(11) T2F state**

The T2F state corresponds to the middle state of a flyby transfer from internal peripheral I/O to internal RAM. The write cycle to internal RAM is started. After entering the T2F state, the bus invariably enters the T3F state.

**(12) T3F state**

The T3F state corresponds to the last state of a flyby transfer from internal peripheral I/O to internal RAM, or a wait state. In the last T3F state, the write strobe signal is made inactive.

**(13) T1FR state**

The bus enters the T1FR state at the beginning of a flyby transfer from internal RAM to internal peripheral I/O. The read cycle from internal RAM is started. After entering the T1FR state, the bus invariably enters the T2FR state.

**(14) T2FR state**

The T2FR state corresponds to the middle state of a flyby transfer from internal RAM to internal peripheral I/O. The write cycle to internal peripheral I/O is started. After entering the T2FR state, the bus invariably enters the T3FR state.

**(15) T3FR state**

T3FR is a state in which it is judged whether a flyby transfer from internal RAM to internal peripheral I/O is continued or not. If the next transfer is executed in block transfer mode, the bus enters the T1FRB state after the T3FR state, otherwise, the bus enters the T4 state.

**(16) T1FRB state**

The bus enters the T1FRB state at the beginning of a flyby block transfer from internal RAM to internal peripheral I/O. The read cycle from internal RAM is started.

**(17) T1FRBI state**

The T1FRBI state corresponds to a wait state of a flyby block transfer from internal RAM to internal peripheral I/O.

A wait state requested by peripheral hardware is generated, and the bus enters the T2FRB state.

**(18) T2FRB state**

The T2FRB state corresponds to the middle state of a flyby block transfer from internal RAM to internal peripheral I/O. The write cycle to internal peripheral I/O is started. After entering the T2FRB state, the bus invariably enters the T3FRB state.

**(19) T3FRB state**

T3FRB is a state in which it is judged whether a flyby transfer from internal RAM to internal peripheral I/O is continued or not. If the next transfer is executed in block transfer mode, the bus enters the T1FRB state after the T3FRB state, otherwise, the bus enters the T4 state.

**(20) T4 state**

The T4 state corresponds to a wait state of a flyby transfer from internal RAM to internal peripheral I/O. A wait state requested by peripheral hardware is generated, and the bus enters the T3 state.



**(21) T1FH state**

The T1FH state corresponds to the standard state of a flyby transfer between external memory and external I/O, and is the executing cycle of this transfer. After entering the T1FH state, the bus enters the T2FH state.

**(22) T1FHI state**

The T1FHI state corresponds to the last state of a flyby transfer between external memory and external I/O, and is a state in which the bus is waiting for end of DMA flyby transfer. After entering the T1FHI state, the bus is released, and enters the TE state.

**(23) T2FH state**

T2FH is a state in which it is judged whether a flyby transfer between external memory and external I/O is continued or not. If the next transfer is executed in block transfer mode, the bus enters the T1FH state after the T2FH state, otherwise, when a wait is issued, the bus enters the T1FHI state. When a wait is not issued, the bus is released, and enters the TE state.

**(24) T3 state**

The bus enters the T3 state when a DMA transfer has been completed, and the bus has been released. After entering the T3 state, the bus invariably enters the TE state.

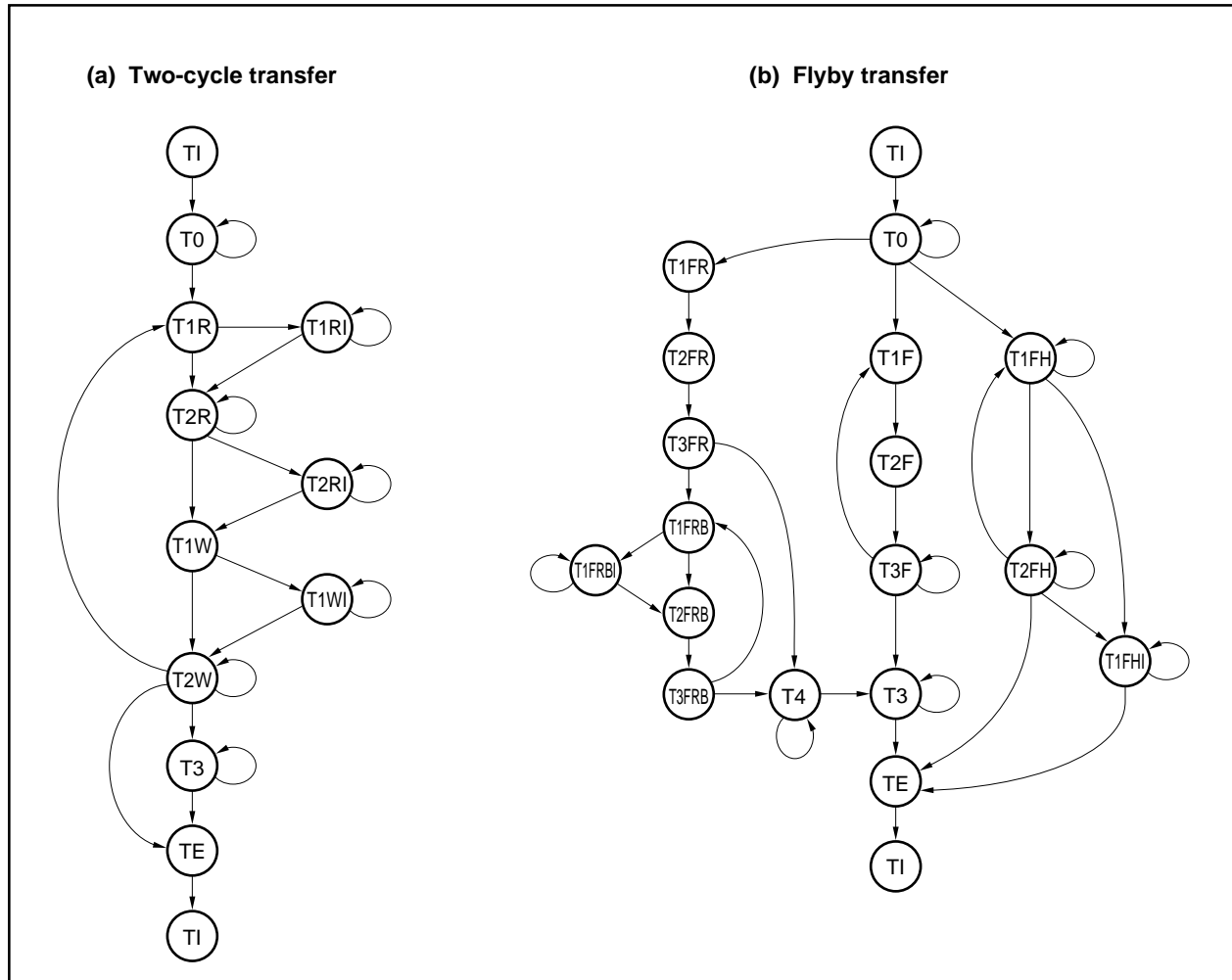
**(25) TE state**

The TE state corresponds to the output state. In the TE state, the DMAC generates an internal signal to indicate DMA transfer end ( $TC_n = 1$ ), and initializes miscellaneous internal signals ( $n = 0$  to 3). After entering the TE state, the bus invariably enters the TI state.

### 6.4.2 DMAC state transition

Except block transfer mode, each time the processing for a DMA service is completed, the bus is released (the bus enters bus release mode).

Figure 6-1. DMAC Bus Cycle State Transition Diagram



## 6.5 Transfer Mode

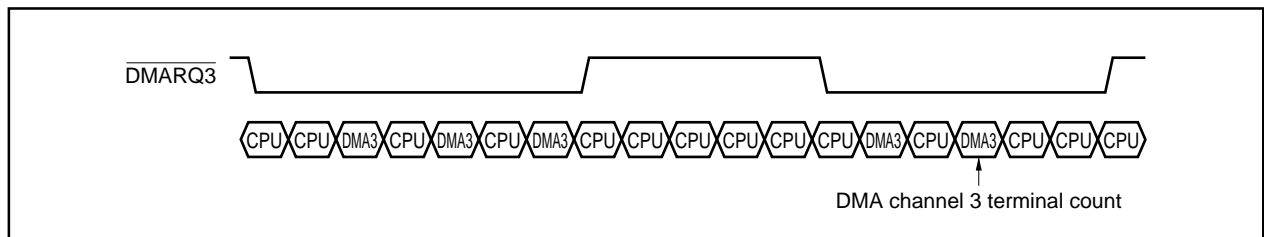
### 6.5.1 Single transfer mode

In single transfer mode, the DMAC releases the bus at each byte/halfword transfer. If there is a subsequent DMA transfer request, transfer is performed again. This operation continues until a terminal count occurs.

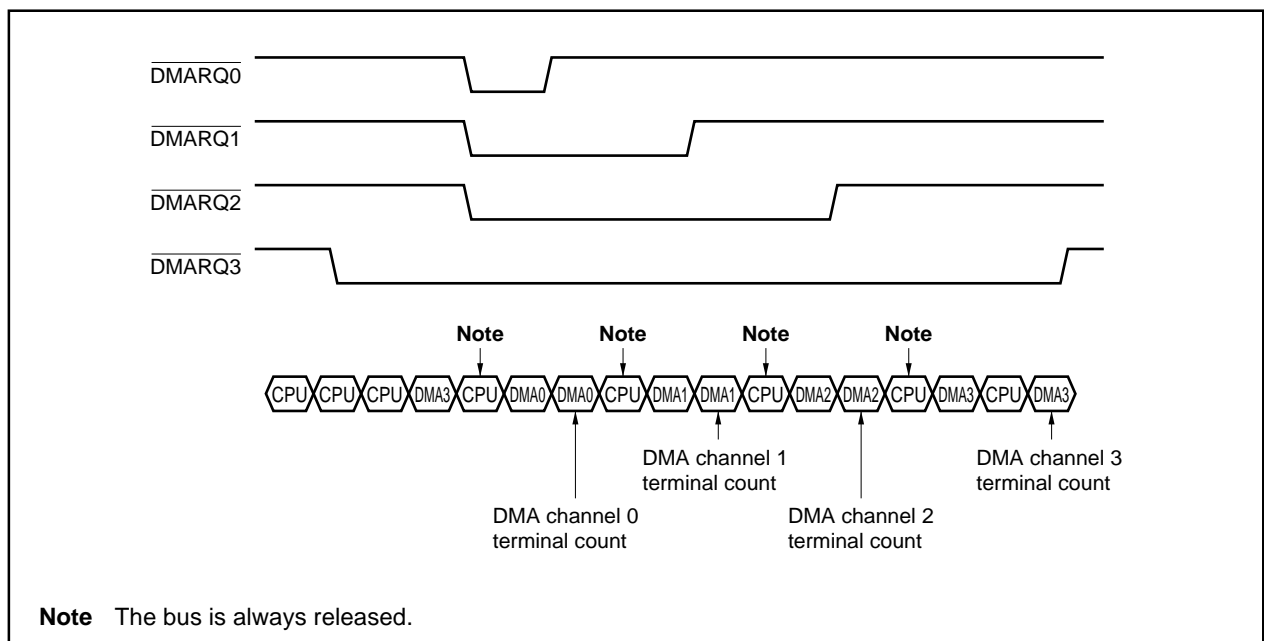
When the DMAC has released the bus, if another higher priority DMA transfer request is issued, the higher priority DMA request always takes precedence. If a single transfer is executed, the internal DMA request is cleared each time one DMA cycle has been completed. If any other channel requests DMA after completion of one DMA cycle, therefore, the DMA transfer request with the highest priority is selected from the channels other than the one for which the DMA cycle has just been completed.

Figures 6-2 and 6-3 show examples of single transfer. Figure 6-3 shows an example of single transfer in which a higher priority DMA request is issued. DMA channels 0 to 2 are in block transfer mode and channel 3 is in single transfer mode.

**Figure 6-2. Single Transfer Example 1**



**Figure 6-3. Single Transfer Example 2**



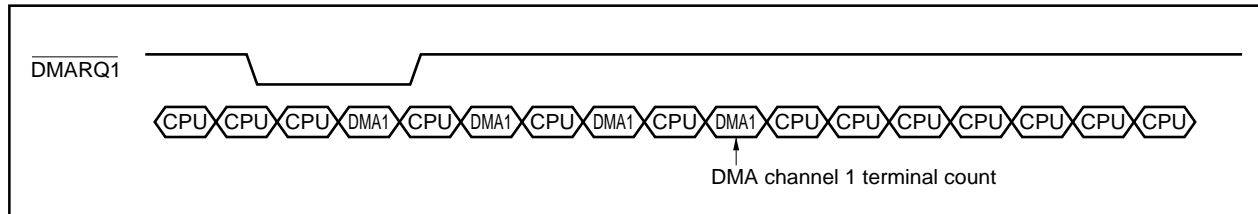
### 6.5.2 Single-step transfer mode

In single-step transfer mode, DMAC releases the bus at each byte/halfword transfer. Once a request signal ( $\overline{\text{DMARQ0}}$  to  $\overline{\text{DMARQ3}}$ ) is received, this operation continues until a terminal count occurs.

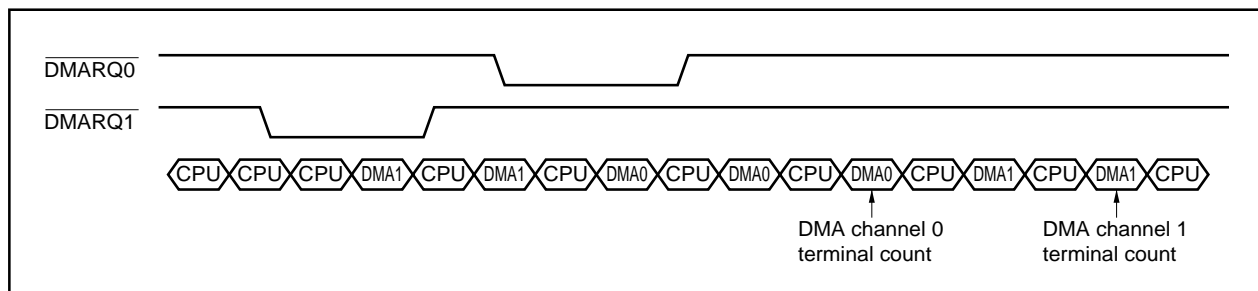
When the DMAC has released the bus, if another higher priority DMA transfer request is issued, the higher priority DMA request always takes precedence.

Figures 6-4 and 6-5 show examples of single-step transfer.

**Figure 6-4. Single-Step Transfer Example 1**



**Figure 6-5. Single-Step Transfer Example 2**



### 6.5.3 Block transfer mode

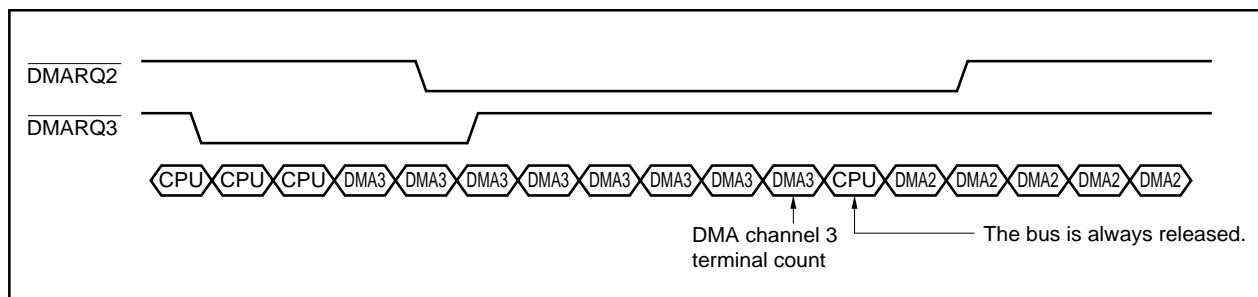
In block transfer mode, once transfer starts, the transfer continues without the bus being released, until a terminal count occurs. No other DMA requests are accepted during block transfer.

After the block transfer ends and DMAC releases the bus, another DMA transfer can be accepted.

Figures 6-6 shows an example of block transfer. In this block transfer example, a high priority DMA request is issued. DMA channels 2 and 3 are in block transfer mode.

Note that caution is required when in block transfer mode. For details, refer to **6.19 Precautions**.

**Figure 6-6. Block Transfer Example**



## 6.6 Transfer Types

### 6.6.1 Two-cycle transfer

In two-cycle transfer, data transfer is performed in two-cycles, source to DMAC then DMAC to destination.

In the first cycle, the source address is output to perform reading from the source to DMAC. In the second cycle, the destination address is output to perform writing from DMAC to the destination.

Figure 6-7 shows examples of two-cycle transfer.

Note that caution is required when in two-cycle transfer. For details, refer to **6.19 Precautions**.

Figure 6-7. Timing of Two-Cycle Transfer (1/4)

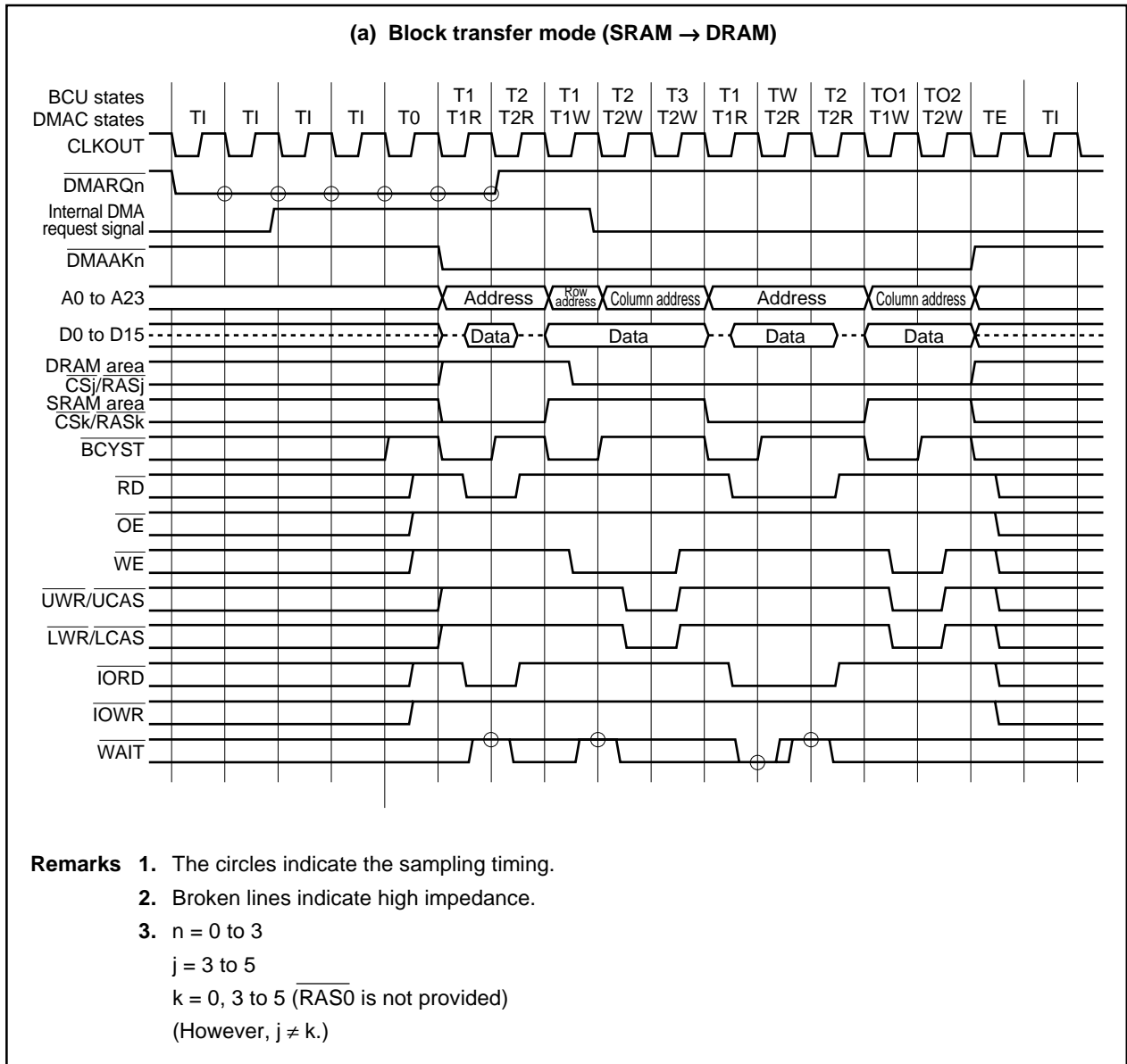
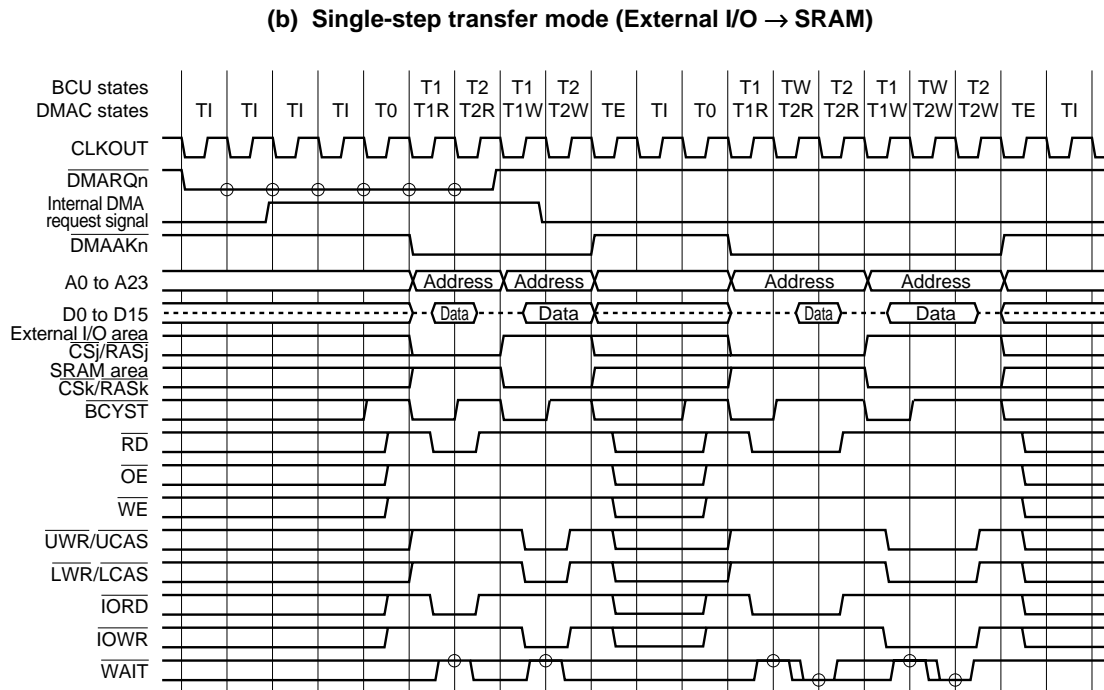


Figure 6-7. Timing of Two-Cycle Transfer (2/4)



- Remarks**
1. The circles indicate the sampling timing.
  2. Broken lines indicate high impedance.
  3.  $n = 0$  to  $3$   
 $j = 3$  to  $5$   
 $k = 0, 3$  to  $5$  ( $\overline{RAS0}$  is not provided)  
 (However,  $j \neq k$ .)

Figure 6-7. Timing of Two-Cycle Transfer (3/4)

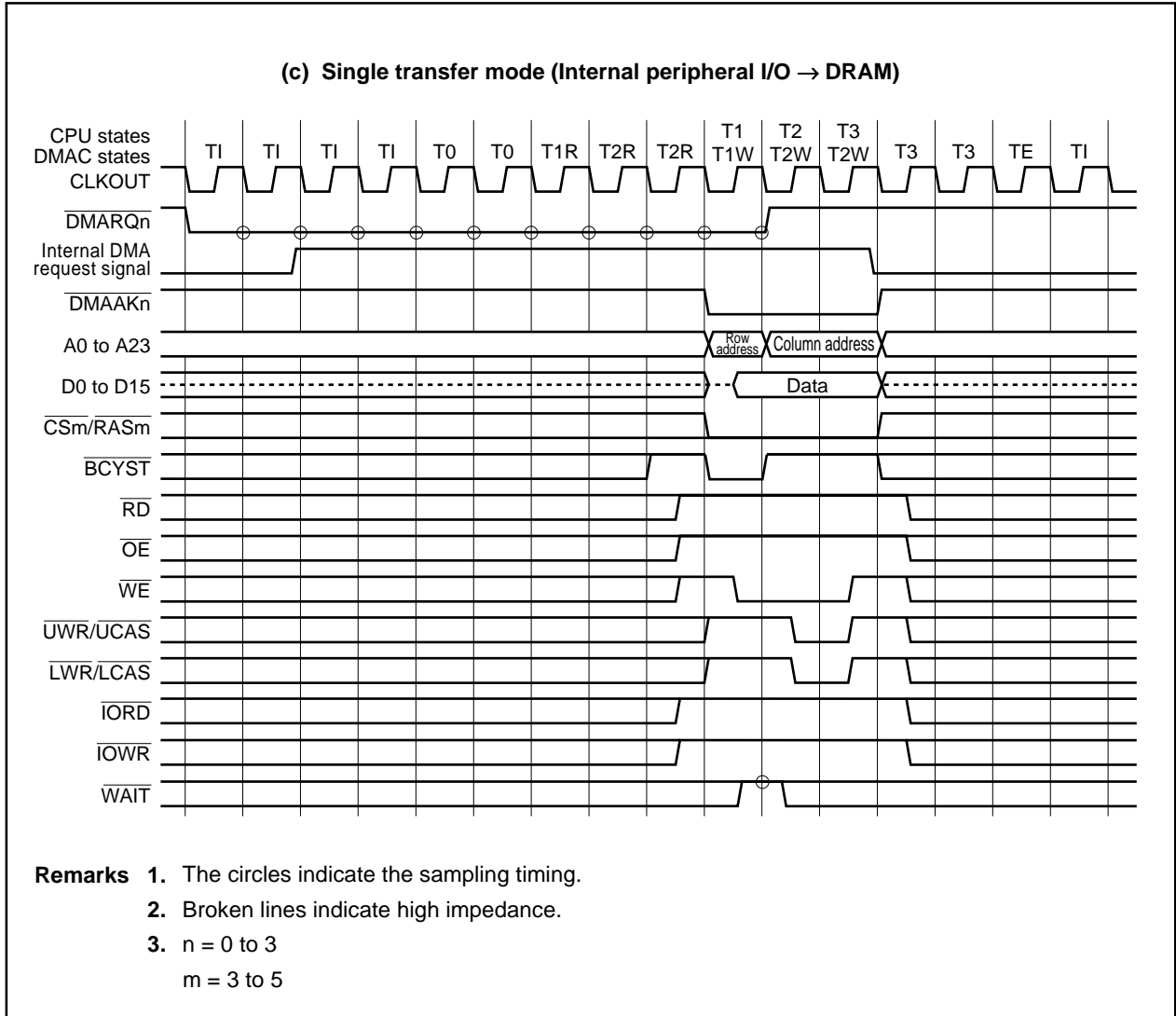
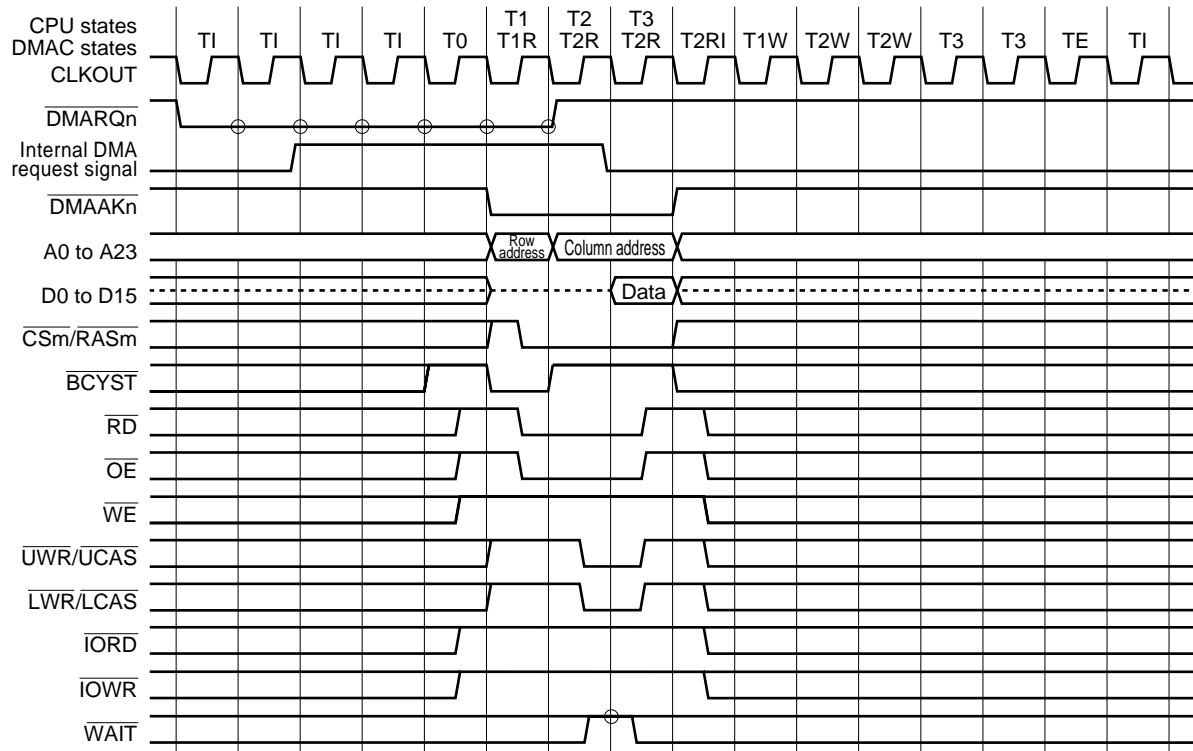


Figure 6-7. Timing of Two-Cycle Transfer (4/4)

## (d) Single transfer mode (DRAM → Internal peripheral I/O)



- Remarks**
1. The circles indicate the sampling timing.
  2. Broken lines indicate high impedance.
  3. n = 0 to 3  
m = 3 to 5



### 6.6.2 Flyby transfer

The V850E/MS2 supports flyby transfer between external memory and external I/O, and internal RAM and internal peripheral I/O.

#### (1) Flyby transfer between external memory and external I/O

This data transfer between memory and I/O is performed in one cycle. To achieve single-cycle transfer, the memory address is always output irrespective of whether it is that of the source or the destination, and the read/write strobe signals for the memory and I/O are made active at the same time.

The external I/O is selected with the  $\overline{\text{DMAAK0}}$  to  $\overline{\text{DMAAK3}}$  signal.

Figure 6-8 shows examples of flyby DMA transfer for an external device.

Figure 6-8. Timing of Flyby Transfer (DRAM → External I/O) (1/3)

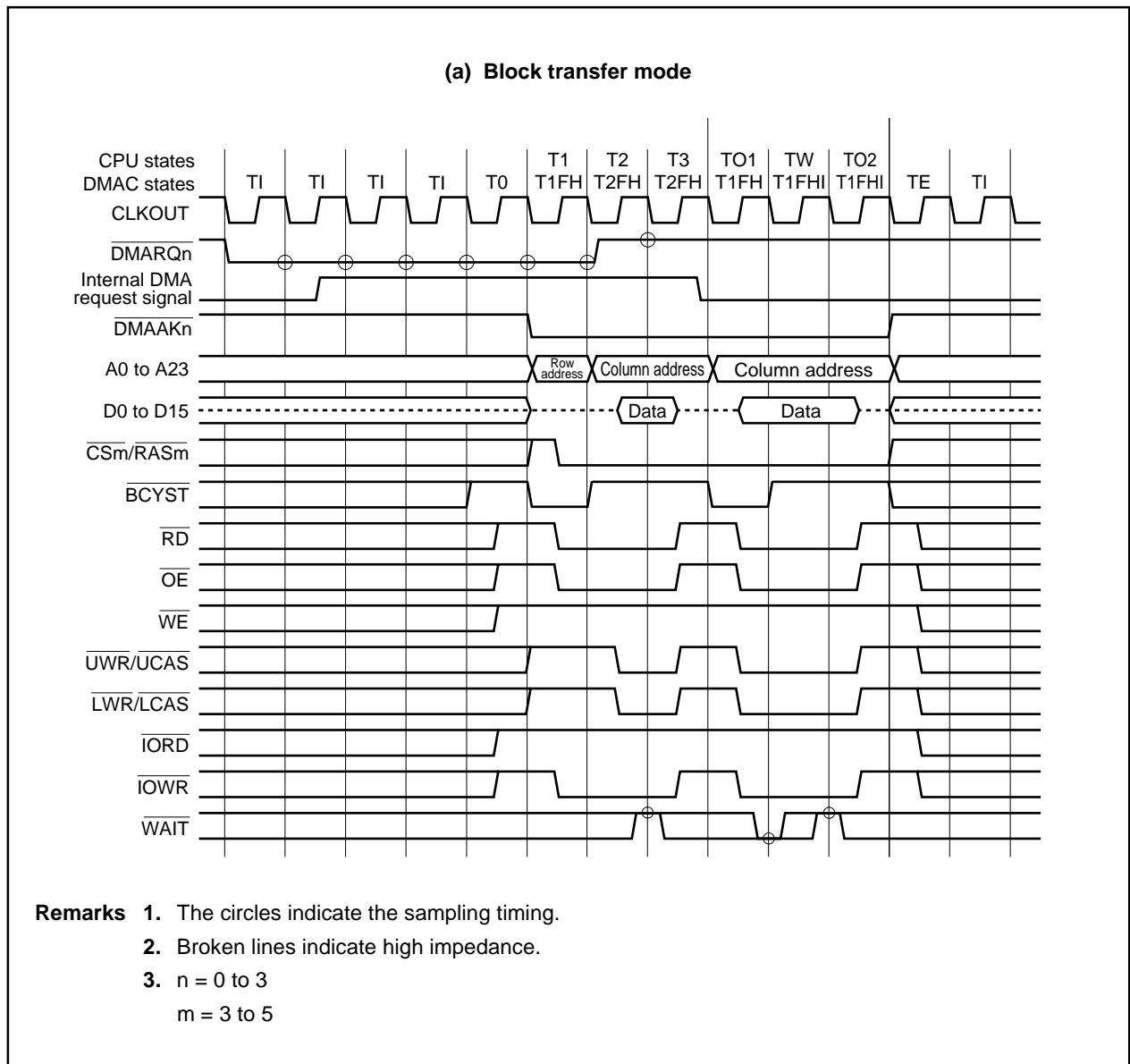


Figure 6-8. Timing of Flyby Transfer (DRAM → External I/O) (2/3)

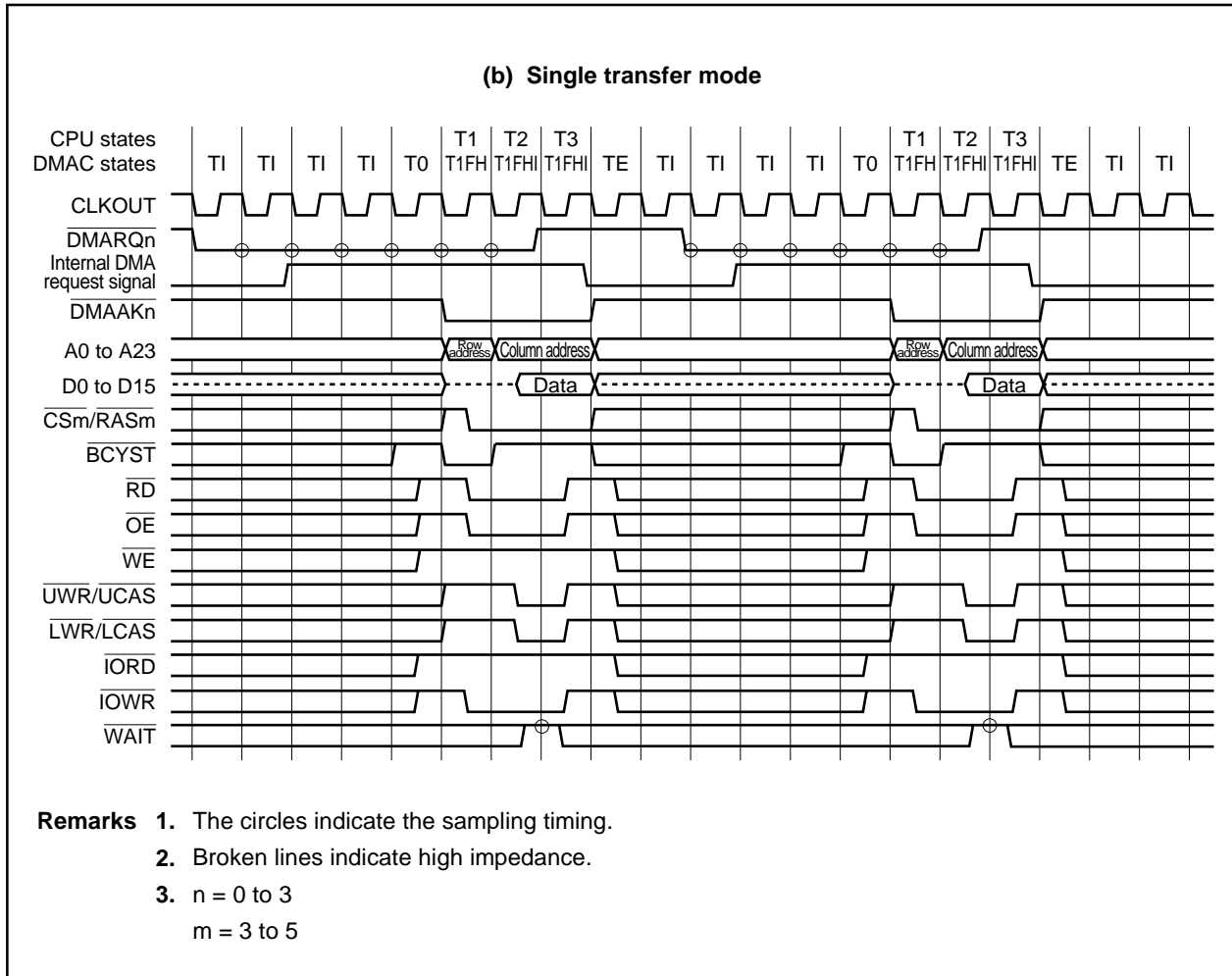
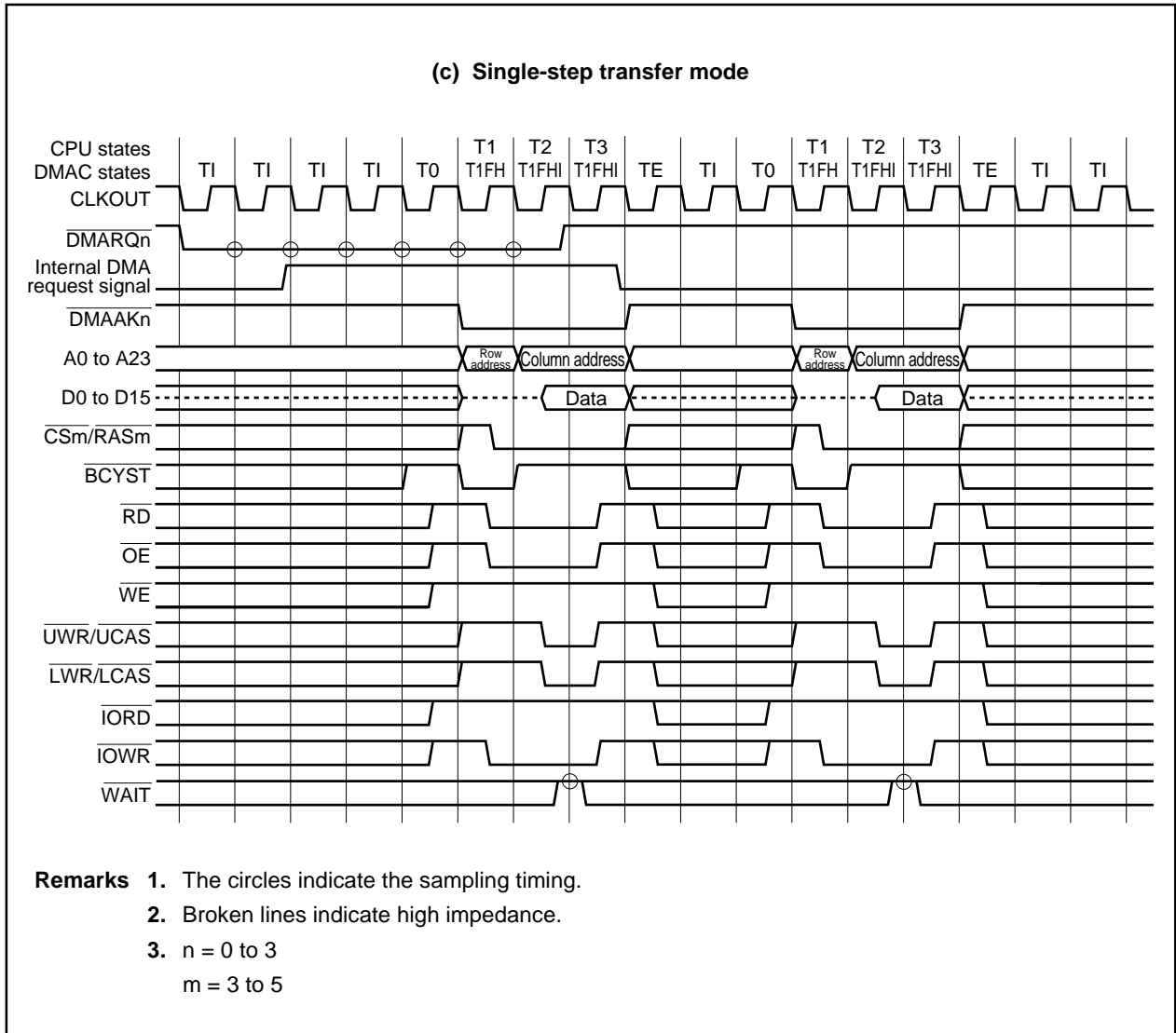


Figure 6-8. Timing of Flyby Transfer (DRAM → External I/O) (3/3)

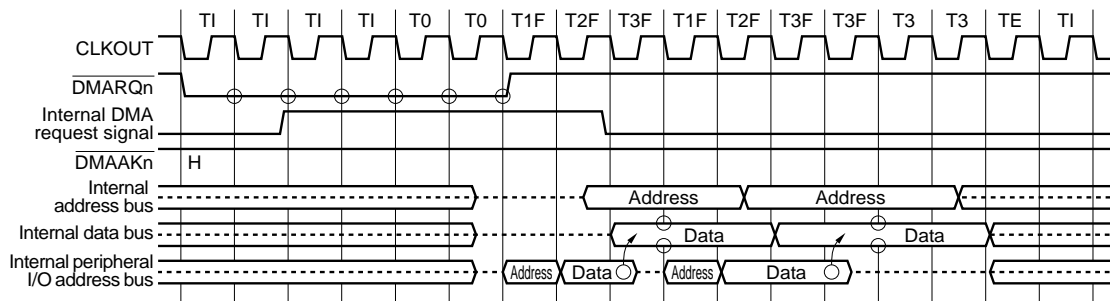


**(2) Flyby transfer between internal RAM and internal peripheral I/O**

Internal RAM and internal peripheral I/O are mapped on different address spaces. Therefore, different addresses are always output, and the read/write strobe signals for internal RAM and internal peripheral I/O are controlled at the same time.

Figure 6-9 shows an example of flyby DMA transfer (block transfer mode) between internal RAM and internal peripheral I/O.

**Figure 6-9. Timing of Flyby Transfer (Internal Peripheral I/O → Internal RAM)**



- Remarks**
1. The circles indicate the sampling timing.
  2. Broken lines indicate high impedance.
  3.  $n = 0$  to 3
  4. With this timing, the external bus operates independently of the internal bus, so there is no influence on the external bus.

## 6.7 Transfer Objects

### 6.7.1 Transfer type and transfer objects

Table 6-1 lists the relationship between transfer type and transfer object.

- Cautions**
1. Among the transfer destinations and sources shown in Table 6-1, when an “x” is indicated for a combination, that operation is not guaranteed.
  2. Make the data bus width of the transfer destination and source the same (for two-cycle transfer and flyby transfer).

Table 6-1. Relationship Between Transfer Type and Transfer Object

		(a) Two-cycle transfer						(b) Flyby transfer			
		Destination						Destination			
		Internal peripheral I/O	External I/O	Internal RAM	External memory			Internal peripheral I/O	External I/O	Internal RAM	External memory
Source	Internal peripheral I/O	×	×	○	○	Source	Internal peripheral I/O	×	×	○	×
	External I/O	×	×	○	○		External I/O	×	×	×	○
	Internal RAM	○	○	○	○		Internal RAM	○	×	×	×
	External memory	○	○	○	○		External memory	×	○	×	×

**Remark**

○: Possible

×: Impossible

### 6.7.2 External bus cycle during DMA transfer

The external bus cycle during DMA transfer is as follows.

Table 6-2. External Bus Cycle During DMA Transfer

Transfer Type	Transfer Object	External Bus Cycle	
Two-cycle transfer	Internal peripheral I/O, Internal RAM	None <sup>Note</sup>	—
	External I/O	Yes	SRAM cycle
	External memory	Yes	Memory access cycle set in the BCT register
Flyby transfer	Between internal RAM and internal peripheral I/O	None <sup>Note</sup>	—
	Between external memory and external I/O	Yes	The memory access DMA flyby transfer cycle set by the BCT register as external memory

**Note** Other external bus cycles, such as a CPU-based bus cycle, can be started.

## 6.8 DMA Channel Priorities

The DMA channel priorities are fixed, as follows:

DMA channel 0 > DMA channel 1 > DMA channel 2 > DMA channel 3

These priorities are valid in the TI state only. In block transfer mode, the channel used for transfer is never switched.

In single-step transfer mode, if a higher priority DMA transfer request is issued while the bus is released (in the TI state), the higher priority DMA transfer request is accepted.

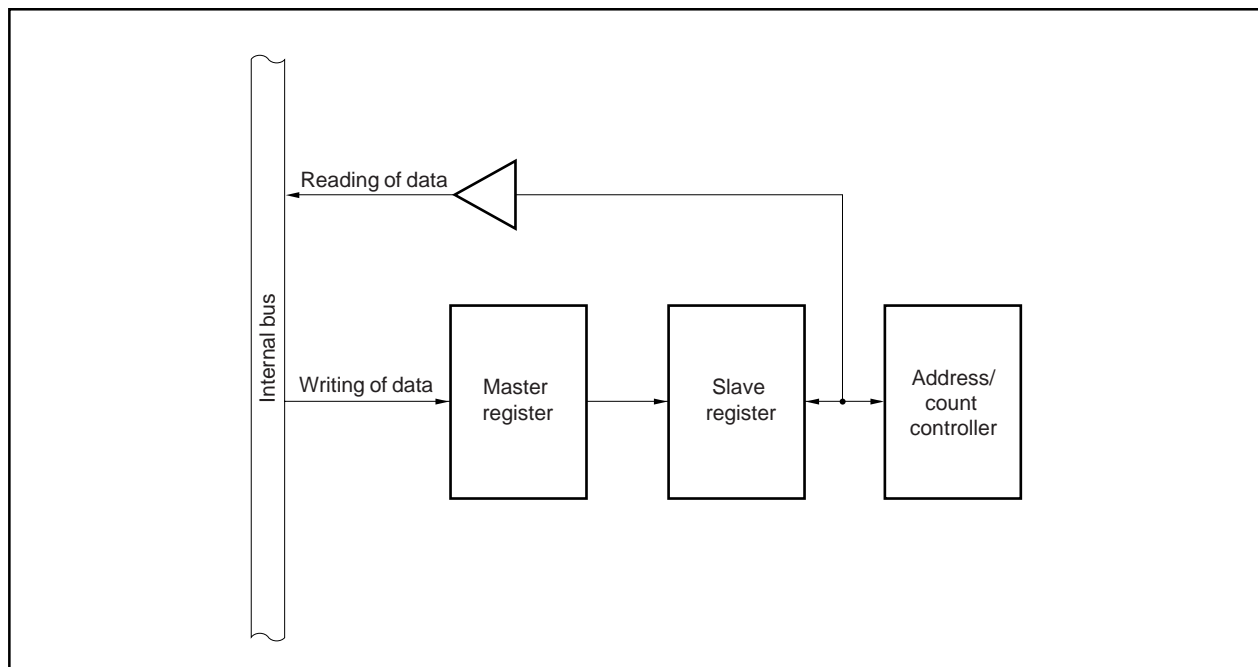
## 6.9 Next Address Setting Function

The DMA source address registers (DSAnH, DSAnL) DMA destination address registers (DDAnH, DDAnL) and DMA byte count register (DBCn) are buffer registers with a 2-stage FIFO configuration (n = 0 to 3).

When the terminal count is issued, these registers are rewritten with the value that was set just previously. Therefore, during DMA transfer, these registers' contents do not become valid even if they are rewritten. When starting DMA transfer with the rewritten contents of these registers, set the ENn bit (1) of the DCHCn register.

Figure 6-10 shows the buffer register configuration.

**Figure 6-10. Buffer Register Configuration**



## 6.10 DMA Transfer Start Factors

There are 3 types of DMA transfer start factors, as shown below.

### (1) Request from an external pin ( $\overline{\text{DMARQn}}$ )

Although requests from the  $\overline{\text{DMARQn}}$  pin are sampled each time the CLKOUT signal falls, sampling should be continued until the  $\overline{\text{DMAAKn}}$  signal becomes active ( $n = 0$  to 3).

If a state in which the ENn bit of the DCHCn register = 1 and the TCn bit = 0 is set, the  $\overline{\text{DMARQn}}$  signal in the T1 state becomes active. If the  $\overline{\text{DMARQn}}$  signal becomes active in the T1 state, it changes to the T0 state and DMA transfer starts.

### (2) Request from software

If the STGn, ENn and TCn bits of the DCHCn register are set as follows, DMA transfer starts ( $n = 0$  to 3).

- STGn bit = 1
- ENn bit = 1
- TCn bit = 0

### (3) Request from internal peripheral I/O

If, when the ENn and TCn bits of the DCHCn register are set as shown below, an interrupt request is issued from the internal peripheral I/O that is set in the DTFRn register, DMA transfer starts ( $n = 0$  to 3).

- ENn bit = 1
- TCn bit = 0

## 6.11 Interrupting DMA Transfer

### 6.11.1 Interruption factors

DMA transfer is interrupted if the following factors occur.

- Bus hold
- Refresh cycle

If the factor that is interrupting DMA transfer disappears, DMA transfer promptly restarts.

### 6.11.2 Forcible interruption

DMA transfer can be forcibly interrupted by an NMI input during DMA transfer.

At such a time, the DMAC resets the ENn bit of the DCHCn register of all channels (0) and activates the DMA transfer disabled state, after which the DMA transfer being executed when the NMI was input is terminated (n = 0 to 3).

When in the single step mode or block transfer mode, the DMA transfer request is held in the DMAC. If the ENn bit is reset (1), DMA transfer restarts from the point where it was interrupted.

When in the single transfer mode, if the ENn bit is set (1), the next DMA transfer request is received and DMA transfer starts.

## 6.12 Terminating DMA Transfer

### 6.12.1 DMA transfer end interrupt

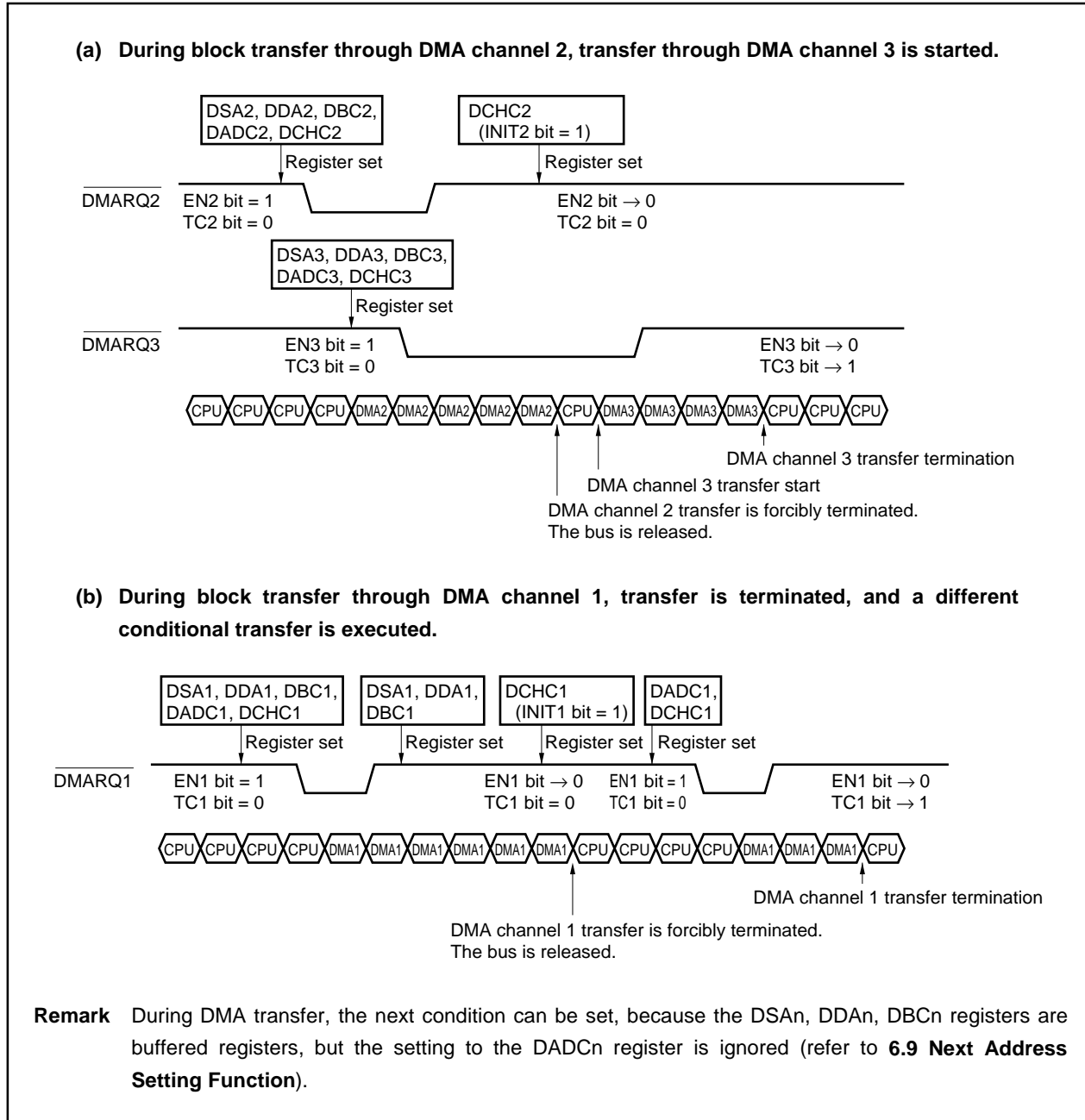
When DMA transfer ends and the TC bit of the corresponding DCHCn register is set (1), a DMA transfer end interrupt (INTDMA<sub>n</sub>) is issued (n = 0 to 3) to the interrupt controller (INTC).



### 6.12.2 Forcible termination

In addition to forcible interruption of DMA transfer by NMI input, DMA transfer can also be terminated forcibly by the INITn bit of the DCHCn register. Examples of the forcible termination operation are shown below (n = 0 to 3).

Figure 6-11. Example of Forcible Termination of DMA Transfer



### 6.13 Boundary of Memory Area

The transfer operation is not guaranteed if the source or the destination address is over the area of DMA objects (external memory, internal RAM, external I/O, or internal peripheral I/O) during DMA transfer.

### 6.14 Transfer of Misalign Data

16-bit DMA transfer of misalign data is not supported. If the source or the destination address is set to an odd address, the LSB bit of the address is forcibly accepted as "0".

### 6.15 Clocks of DMA Transfer

Table 6-3 lists the overhead before and after DMA transfer and minimum execution clock for DMA transfer.

**Table 6-3. Minimum Execution Clock in DMA Cycle**

From accepting $\overline{\text{DMARQn}}$ to falling edge of $\overline{\text{DMAAKn}}$	4 clocks
External memory access	Refer to miscellaneous memory and I/O cycle
Internal RAM access	2 clocks
Internal peripheral I/O access	3 clocks

**Remark** n = 0 to 3

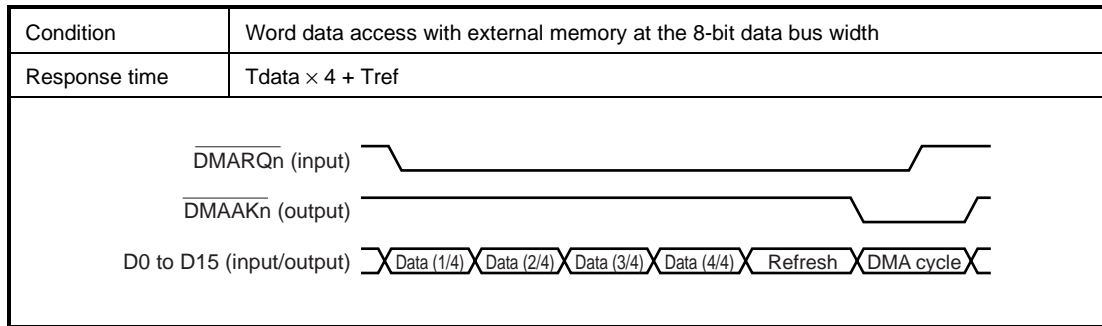
### 6.16 Maximum Response Time to DMA Request

Under the conditions shown below, the response time to a DMA request becomes the maximum time (this is the state permitted by the DRAM refresh cycle).

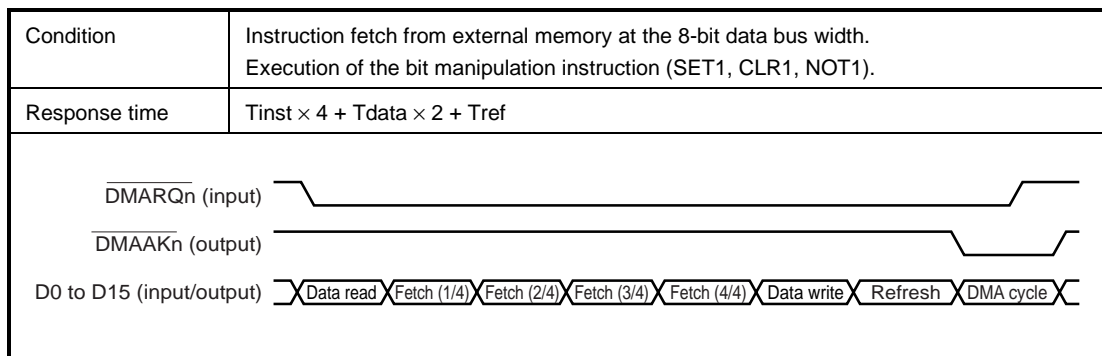
#### (1) Condition 1

Condition	Instruction fetch from external memory at the 8-bit data bus width
Response time	$T_{\text{inst}} \times 4 + T_{\text{ref}}$
<p>The diagram illustrates the timing for instruction fetch from external memory. It shows three signals: <math>\overline{\text{DMARQn}}</math> (input, active low), <math>\overline{\text{DMAAKn}}</math> (output, active low), and D0 to D15 (input/output, bidirectional). The sequence of events is: <math>\overline{\text{DMARQn}}</math> goes low, followed by four 8-bit fetch cycles (Fetch (1/4), Fetch (2/4), Fetch (3/4), Fetch (4/4)). After the fourth fetch, <math>\overline{\text{DMAAKn}}</math> goes low, and a Refresh cycle occurs. Finally, the DMA cycle begins.</p>	

(2) Condition 2



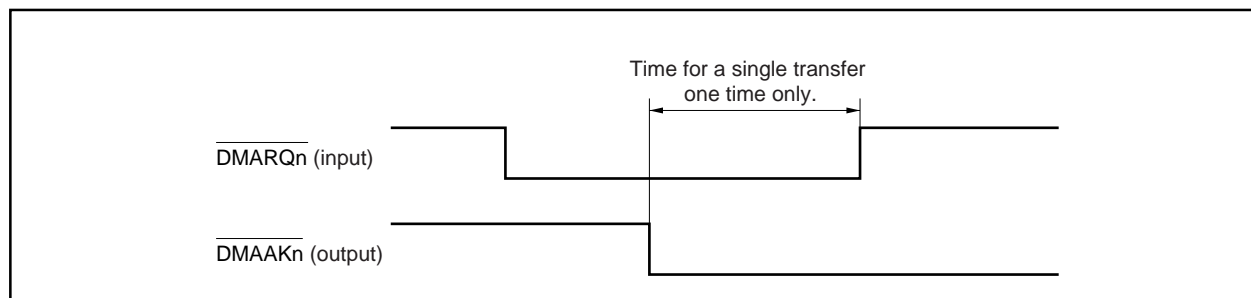
(3) Condition 3



- Remarks**
1.  $T_{inst}$ : The number of clocks per bus cycle during instruction fetch.  
 $T_{data}$ : The number of clocks per bus cycle during data access.  
 $T_{ref}$ : The number of clocks per refresh cycle.
  2.  $n = 0$  to 3

### 6.17 One Time Single Transfer with $\overline{\text{DMARQn}}$ to $\overline{\text{DMARQ3}}$

To execute one time single transfer to external memory via  $\overline{\text{DMARQn}}$  signal input,  $\overline{\text{DMARQn}}$  should be inactive within the clock time shown in Table 6-4 from when  $\overline{\text{DMAAKn}}$  becomes active (n = 0 to 3). If  $\overline{\text{DMARQn}}$  is active for more than the clock time shown in Table 6-4, single transfers are continuously executed.



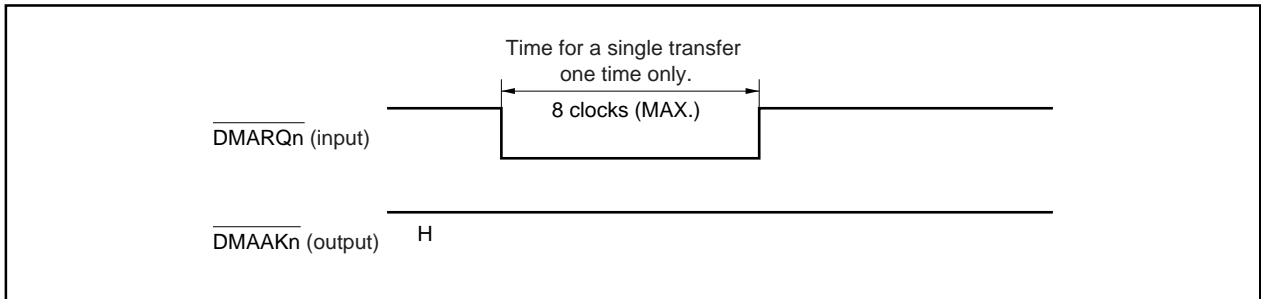
**Table 6-4.  $\overline{\text{DMAAKn}}$  Active →  $\overline{\text{DMARQn}}$  Inactive Time for Single Transfer to External Memory**

Transfer Type	Source	Destination	$\overline{\text{DMAAKn}}$ Signal Active → $\overline{\text{DMARQn}}$ Inactive Time (Max.) <sup>Note</sup>
Two-cycle transfer	DRAM (off page)	All objects	5 clocks
	DRAM (on page)	All objects	4 clocks
	SRAM or external I/O	All objects	4 clocks
	Internal RAM or internal peripheral I/O	DRAM (off page)	7 clocks
	Internal RAM or internal peripheral I/O	DRAM (on page)	6 clocks
	Internal RAM	SRAM or external I/O	6 clocks
	Internal peripheral I/O	SRAM	6 clocks
Flyby transfer	DRAM (off page) ↔ External I/O		3 clocks
	DRAM (on page) ↔ External I/O		2 clocks
	SRAM ↔ External I/O		2 clocks

**Note** When inserting waits, add the number of waits together.

**Remark** n = 0 to 3

Also, if a single transfer is executed between internal RAM and internal peripheral I/O, it is necessary that the  $\overline{\text{DMARQn}}$  signal be inactivated within 8 clock cycles after it is activated. If 8 clock cycles are exceeded, transfer may continue. Note that the  $\overline{\text{DMAAKn}}$  signal does not become active at this time.



### 6.18 Bus Arbitration for CPU

The CPU can access any external memory, external I/O, internal RAM, and internal peripheral I/O not undergoing DMA transfer.

While data is being transferred between external memory and external I/O, the CPU can access internal RAM and internal peripheral I/O.

While data transfer is being executed between internal RAM and internal peripheral I/O, the CPU can access external memory and external I/O.

### 6.19 Precaution

If a DMA transfer which satisfies all the following conditions is interrupted by NMI input, the  $\overline{\text{DMAAKn}}$  signal may become active and remain so until the next DMA transfer ( $n = 0$  to 3).

- Two-cycle transfer
- Block transfer mode
- Transfer from external memory to external memory, or from external I/O to external I/O
- The destination side is EDO DRAM, with no-wait on-page access.

Note that device operations other than the  $\overline{\text{DMAAKn}}$  signal are not influenced.

Change the  $\overline{\text{DMAAKn}}$  signal to inactive by executing the routine shown below in the NMI handler, etc.

LD.B DDIS[r0], reg ; Confirm the interrupted DMA channel by NMI input.

ST.B reg, DRST[r0]; Restart transfer in the interrupted channel.

ST.B r0, DRST[r0] ; By immediately interrupting transfer again, after DMA transfer only once, the  $\overline{\text{DMAAKn}}$  signal becomes inactive.

[MEMO]

## CHAPTER 7 INTERRUPT/EXCEPTION PROCESSING FUNCTION

The V850E/MS2 is provided with a dedicated interrupt controller (INTC) for interrupt servicing and can process a total of 36 interrupt requests.

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

The V850E/MS2 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 the generation of an exception event (fetching of an illegal op code), which is known as an exception trap.

### 7.1 Features

#### ○ Interrupts

- Non-maskable interrupts: 1 source
- Maskable interrupts: 35 sources
- 8 levels of programmable priorities
- Mask specification for interrupt requests according to priority
- Mask can be specified for each maskable interrupt request.
- Noise elimination, edge detection, and valid edge of external interrupt request signal can be specified.

#### ○ Exceptions

- Software exceptions: 32 sources
- Exception trap: 1 source (illegal op code exception)

Interrupt/exception sources are listed in Table 7-1.

Table 7-1. Interrupt List (1/2)

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Handler Address	Restored PC
		Name	Controlling Register	Source	Generating Unit				
Reset	Interrupt	RESET	—	RESET input	Pin	—	0000H	00000000H	Undefined
Non-maskable	Interrupt	NMI	—	NMI input	Pin	—	0010H	00000010H	nextPC
Software exception	Exception	TRAP0 <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	INTOV10	OVI10	Timer 10 overflow	RPU	0	0080H	00000080H	nextPC
	Interrupt	INTOV11	OVI11	Timer 11 overflow	RPU	1	0090H	00000090H	nextPC
	Interrupt	INTOV12	OVI12	Timer 12 overflow	RPU	2	00A0H	000000A0H	nextPC
	Interrupt	INTOV13	OVI13	Timer 13 overflow	RPU	3	00B0H	000000B0H	nextPC
	Interrupt	INTP100/ INTCC100	P10IC0	Match of INTP100 pin/CC100	Pin/RPU	4	0100H	00000100H	nextPC
	Interrupt	INTP101/ INTCC101	P10IC1	Match of INTP101 pin/CC101	Pin/RPU	5	0110H	00000110H	nextPC
	Interrupt	INTP102/ INTCC102	P10IC2	Match of INTP102 pin/CC102	Pin/RPU	6	0120H	00000120H	nextPC
	Interrupt	INTP103/ INTCC103	P10IC3	Match of INTP103 pin/CC103	Pin/RPU	7	0130H	00000130H	nextPC
	Interrupt	INTP110/ INTCC110	P11IC0	Match of INTP110 pin/CC110	Pin/RPU	8	0140H	00000140H	nextPC
	Interrupt	INTP111/ INTCC111	P11IC1	Match of INTP111 pin/CC111	Pin/RPU	9	0150H	00000150H	nextPC
	Interrupt	INTP112/ INTCC112	P11IC2	Match of INTP112 pin/CC112	Pin/RPU	10	0160H	00000160H	nextPC
	Interrupt	INTP113/ INTCC113	P11IC3	Match of INTP113 pin/CC113	Pin/RPU	11	0170H	00000170H	nextPC
	Interrupt	INTCC120	P12IC0	Match of CC120	RPU	12	0180H	00000180H	nextPC
	Interrupt	INTCC121	P12IC1	Match of CC121	RPU	13	0190H	00000190H	nextPC
	Interrupt	INTCC122	P12IC2	Match of CC122	RPU	14	01A0H	000001A0H	nextPC
	Interrupt	INTCC123	P12IC3	Match of CC123	RPU	15	01B0H	000001B0H	nextPC
	Interrupt	INTP130/ INTCC130	P13IC0	Match of INTP130 pin/CC130	Pin/RPU	16	01C0H	000001C0H	nextPC
	Interrupt	INTCC131	P13IC1	Match of CC131	RPU	17	01D0H	000001D0H	nextPC

**Note** n = 0 to FH



Table 7-1. Interrupt List (2/2)

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Handler Address	Restored PC
		Name	Controlling Register	Source	Generating Unit				
Maskable	Interrupt	INTCC132	P13IC2	Match of CC132	RPU	18	01E0H	000001E0H	nextPC
	Interrupt	INTCC133	P13IC3	Match of CC133	RPU	19	01F0H	000001F0H	nextPC
	Interrupt	INTCM40	CMIC40	CM40 match signal	RPU	20	0280H	00000280H	nextPC
	Interrupt	INTCM41	CMIC41	CM41 match signal	RPU	21	0290H	00000290H	nextPC
	Interrupt	INTDMA0	DMAIC0	DMA channel 0 transfer completion	DMAC	22	02A0H	000002A0H	nextPC
	Interrupt	INTDMA1	DMAIC1	DMA channel 1 transfer completion	DMAC	23	02B0H	000002B0H	nextPC
	Interrupt	INTDMA2	DMAIC2	DMA channel 2 transfer completion	DMAC	24	02C0H	000002C0H	nextPC
	Interrupt	INTDMA3	DMAIC3	DMA channel 3 transfer completion	DMAC	25	02D0H	000002D0H	nextPC
	Interrupt	INTCSIO	CSIC0	CSIO transmission/reception completion	SIO	26	0300H	00000300H	nextPC
	Interrupt	INTSER0	SEIC0	UART0 reception error	SIO	27	0310H	00000310H	nextPC
	Interrupt	INTSR0	SRIC0	UART0 reception completion	SIO	28	0320H	00000320H	nextPC
	Interrupt	INTST0	STIC0	UART0 transmission completion	SIO	29	0330H	00000330H	nextPC
	Interrupt	INTCSI1	CSIC1	CSI1 transmission/reception completion	SIO	30	0340H	00000340H	nextPC
	Interrupt	INTSER1	SEIC1	UART1 reception error	SIO	31	0350H	00000350H	nextPC
	Interrupt	INTSR1	SRIC1	UART1 reception completion	SIO	32	0360H	00000360H	nextPC
	Interrupt	INTST1	STIC1	UART1 transmission completion	SIO	33	0370H	00000370H	nextPC
	Interrupt	INTAD	ADIC	A/D conversion completion	ADC	34	0400H	00000400H	nextPC

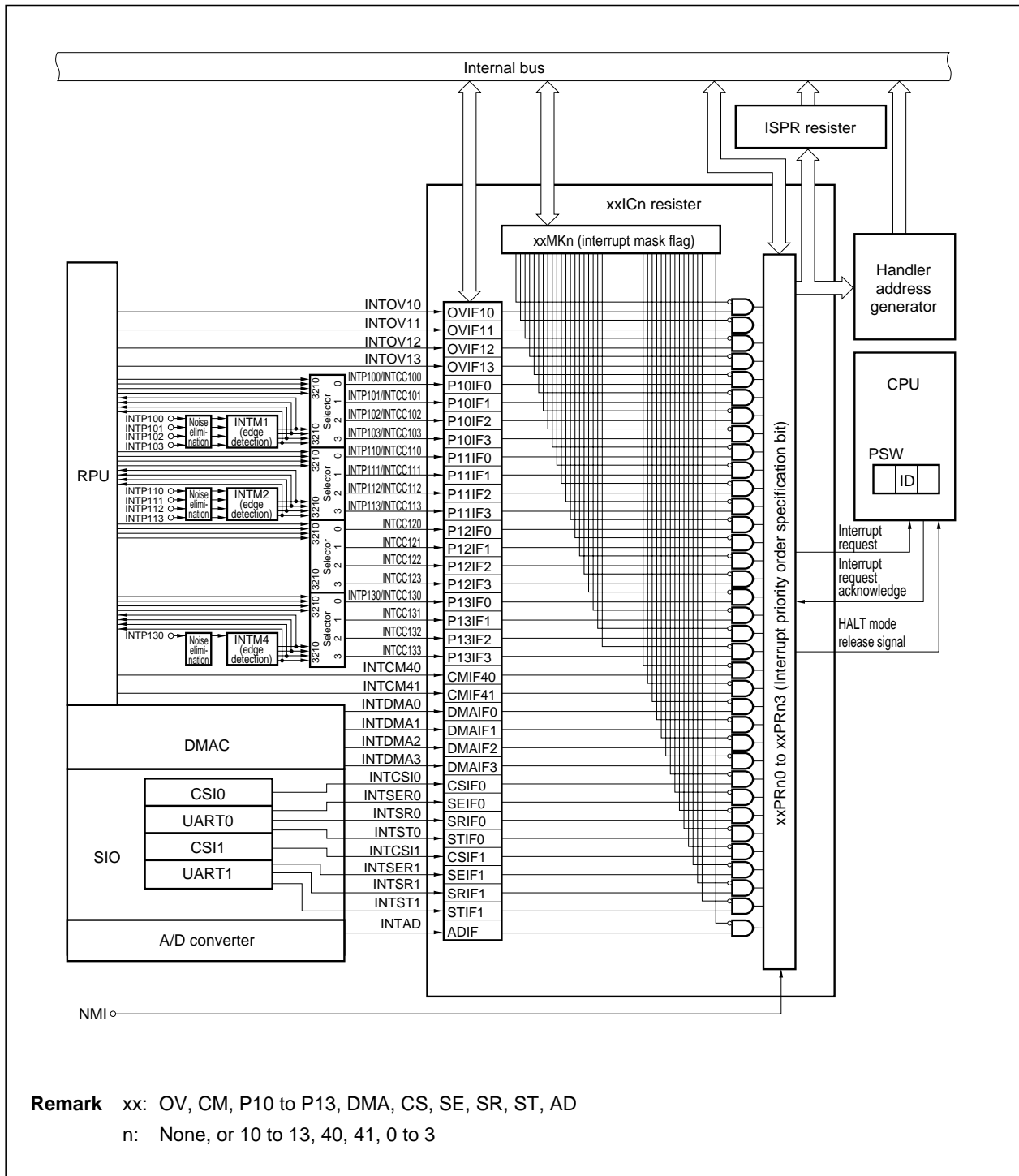
**Caution** INTP1mn (external interrupt) and INTCC1mn (compare register match interrupt) share a control register. Set the valid interrupt request using bits 3 to 0 (IMS1mn) of timer unit mode registers 10 to 13 (TUM10 to TUM13) (see 9.3 (1) Timer unit mode registers 10 to 13 (TUM10 to TUM13)).

**Remarks** 1. Default priority: The priority order 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, which is saved when an interrupt is acknowledged during division (DIV, DIVH, DIVU, and DIVHU) instruction execution, is the value of the PC of the current instruction (DIV, DIVH, DIVU, and DIVHU).

2. The execution address of the illegal instruction when an illegal op code exception occurs is d

Figure 7-1. Block Diagram of Interrupt Control Function



## 7.2 Non-Maskable Interrupt

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

A 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 program of the non-maskable interrupt is being executed (PSW.NP = 1), the acknowledgement of another non-maskable interrupt requests is held pending. The pending NMI is acknowledged after the original service program 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 program for an NMI, the number of NMIs that will be acknowledged after PSW.NP goes to "0", is only one.

**Remark** PSW.NP: The NP bit of the PSW register.

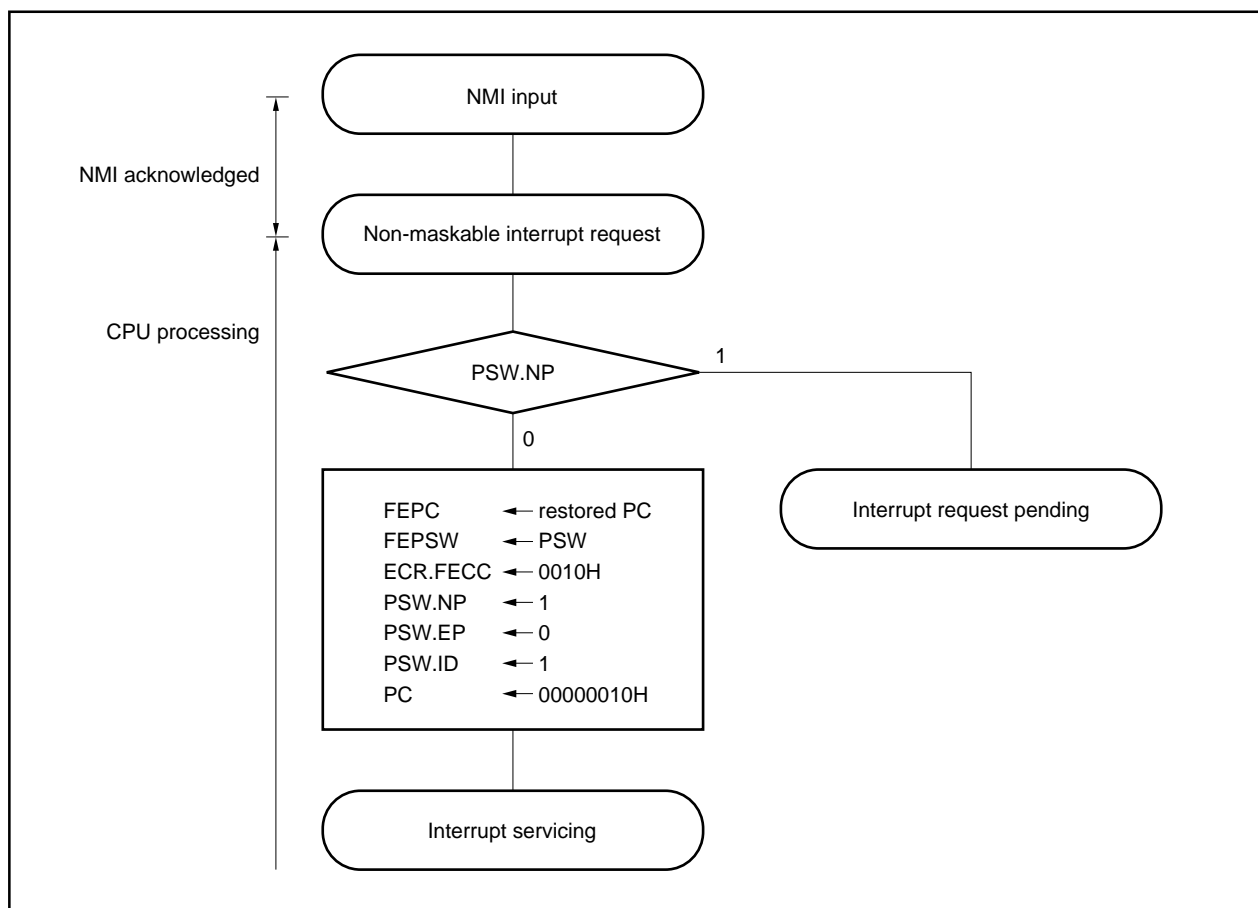
### 7.2.1 Operation

If a non-maskable interrupt is generated, 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 the exception code (0010H) to the higher halfword (FECC) of ECR.
- (4) Sets the NP and ID bits of PSW and clears the EP bit.
- (5) Sets the handler address (00000010H) corresponding to the non-maskable interrupt to the PC, and transfers control.

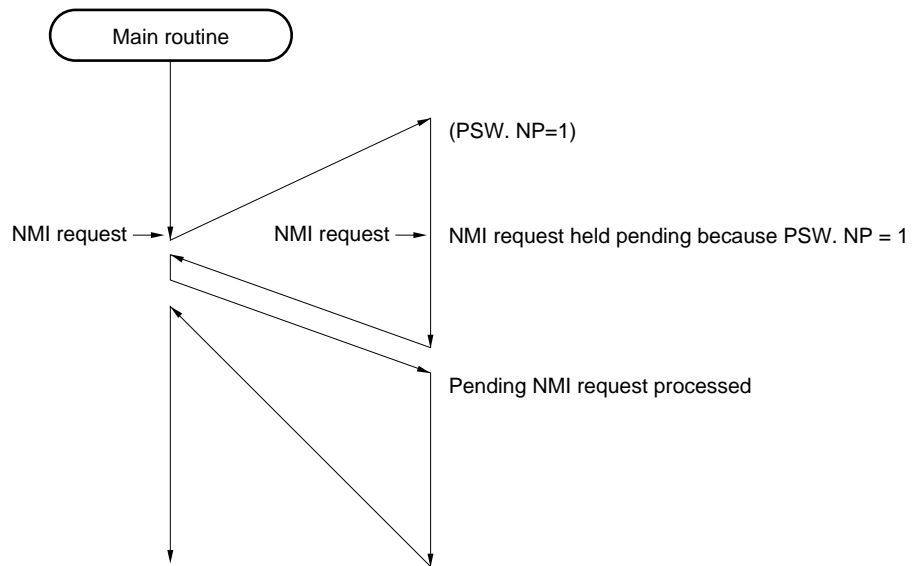
The processing configuration of a non-maskable interrupt is shown in Figure 7-2.

**Figure 7-2. Processing Configuration of Non-Maskable Interrupt**

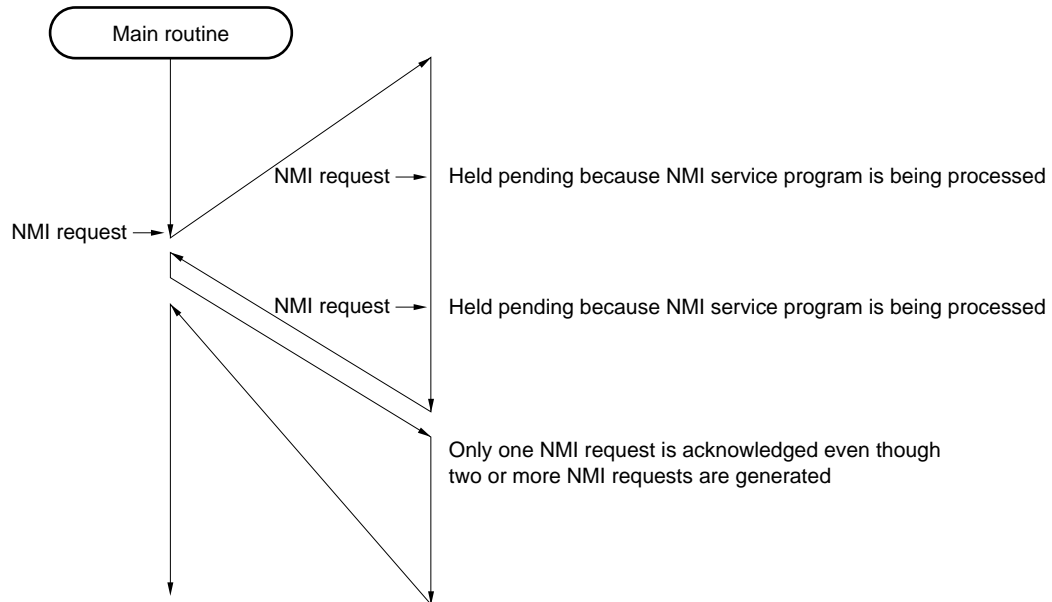


**Figure 7-3. Acknowledging Non-Maskable Interrupt Request**

**(a) If a new NMI request is generated while an NMI service program is being executed:**



**(b) If a new NMI request is generated twice while an NMI service program is being executed:**



### 7.2.2 Restore

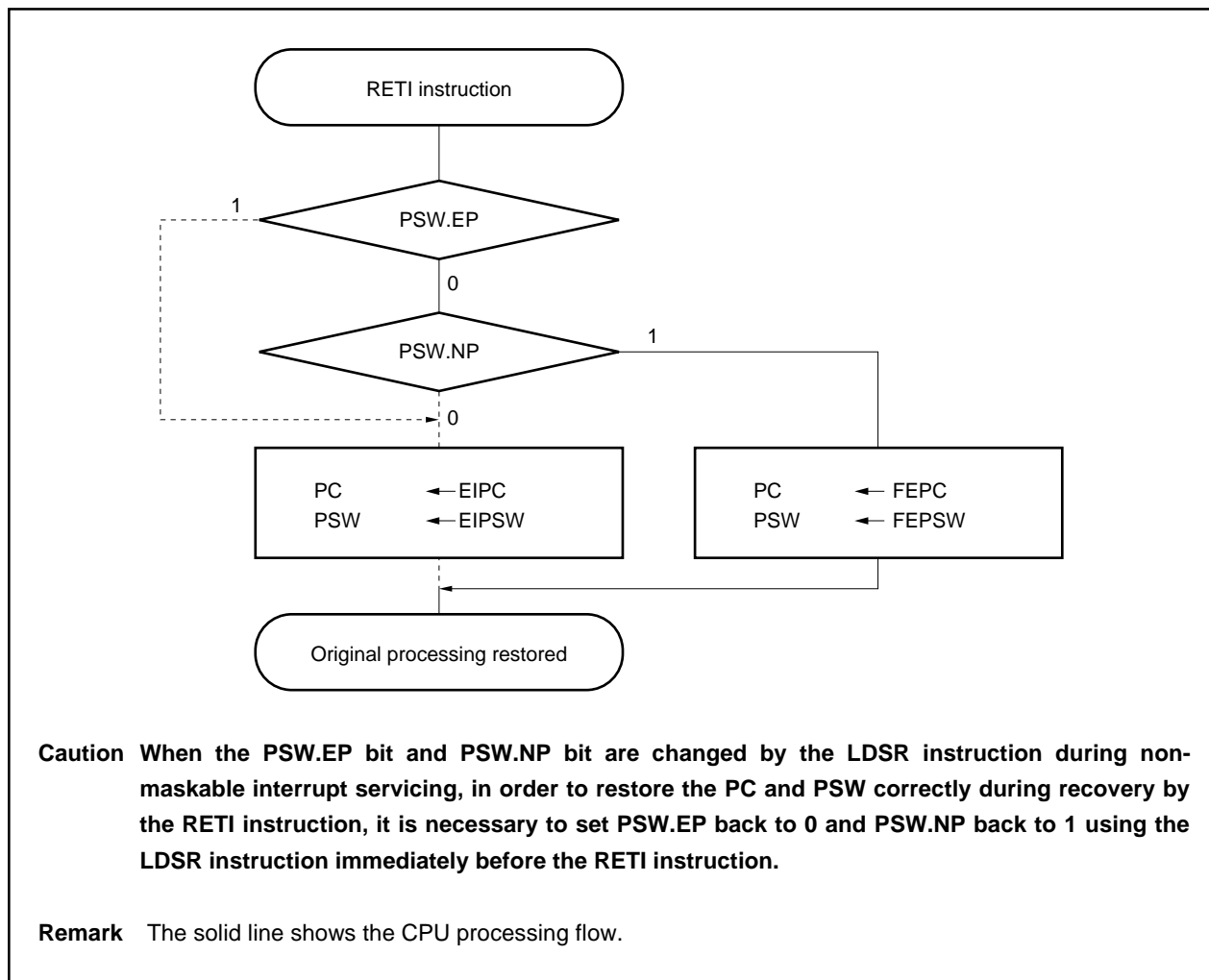
Execution is restored from the non-maskable interrupt servicing by the RETI instruction.

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 the 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 address of the restored PC and PSW.

Figure 7-4 illustrates how the RETI instruction is processed.

**Figure 7-4. RETI Instruction Processing**



### 7.2.3 Non-maskable interrupt status flag (NP)

The NP flag is bit 7 of the PSW.

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

PSW	31																										8 7 6 5 4 3 2 1 0								
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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																																
7	NP		NMI Pending Indicates that NMI interrupt servicing is in progress. 0: No NMI interrupt servicing 1: NMI interrupt currently being processed																																

### 7.2.4 Noise elimination

NMI pin noise is eliminated with analog delay. The delay time is 60 to 220 ns. The signal input that changes within the delay time is not internally acknowledged.

The NMI pin is used for releasing the software STOP mode. In the software STOP mode, the internal system clock is not used for noise elimination because the internal system clock is stopped.

### 7.2.5 Edge detection function

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

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

	7	6	5	4	3	2	1	0		
INTM0	0	0	0	0	0	0	0	ESN0	Address FFFFF180H	After reset 00H

Bit Position	Bit Name	Function
0	ESN0	Edge Select NMI Specifies the NMI pin's valid edge. 0: Falling edge 1: Rising edge

### 7.3 Maskable Interrupts

Maskable interrupt requests can be masked by interrupt control registers. The V850E/MS1 has 47 maskable interrupt sources.

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

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

When the EI instruction is executed in an interrupt servicing routine, the interrupt enabled (EI) status is set which enables interrupts having a higher priority than the interrupt requests in progress (specified by the interrupt control register). Note that only interrupts with a higher priority will have this capability; interrupts with the same priority level cannot be nested.

However, if multiplexed interrupts are executed, the following servicing is necessary.

- <1> Save EIPC and EIPSW in memory or a general-purpose register before executing the EI instruction.
- <2> Execute the DI instruction before executing the RETI instruction, then reset EIPC and EIPSW with the values saved in <1>.

#### 7.3.1 Operation

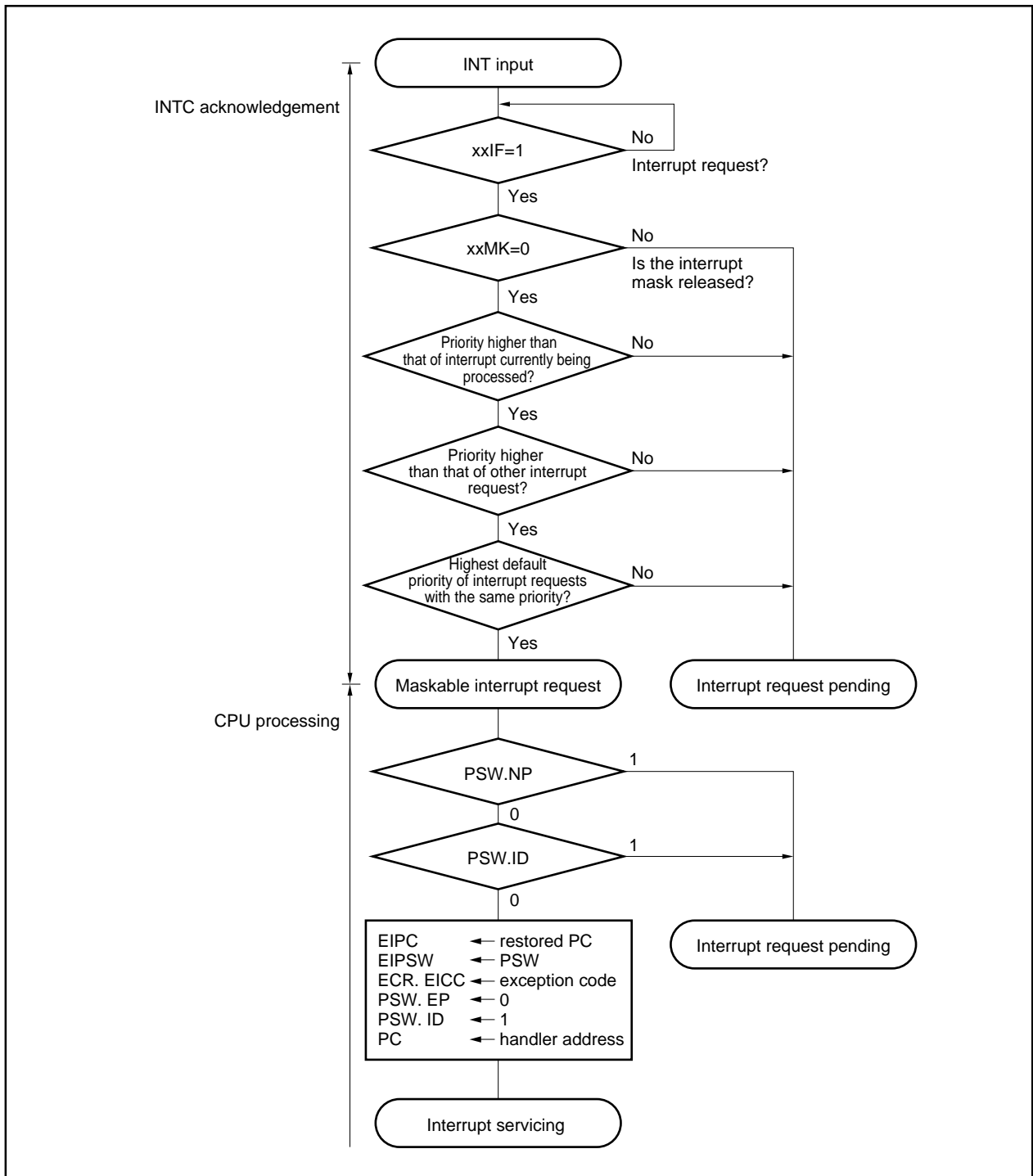
If a maskable interrupt occurs by INT input, the CPU performs the following servicing, and transfers control to a handler routine:

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

The processing configuration of a maskable interrupt is shown in Figure 7-5.



Figure 7-5. Maskable Interrupt Servicing



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

### 7.3.2 Restore

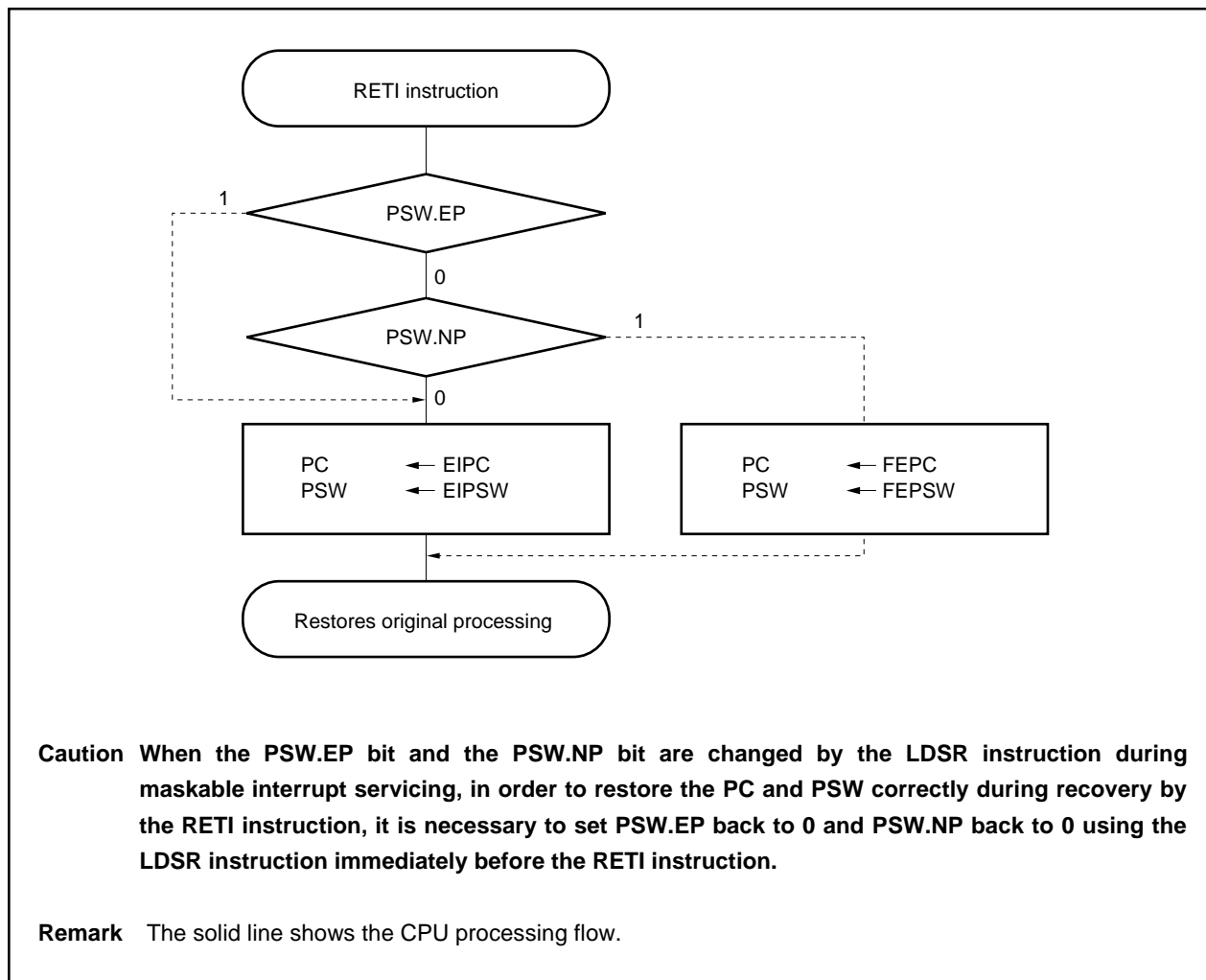
To restore from the maskable interrupt servicing, the RETI instruction is used.

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

Figure 7-6 illustrates the processing of the RETI instruction.

**Figure 7-6. RETI Instruction Processing**



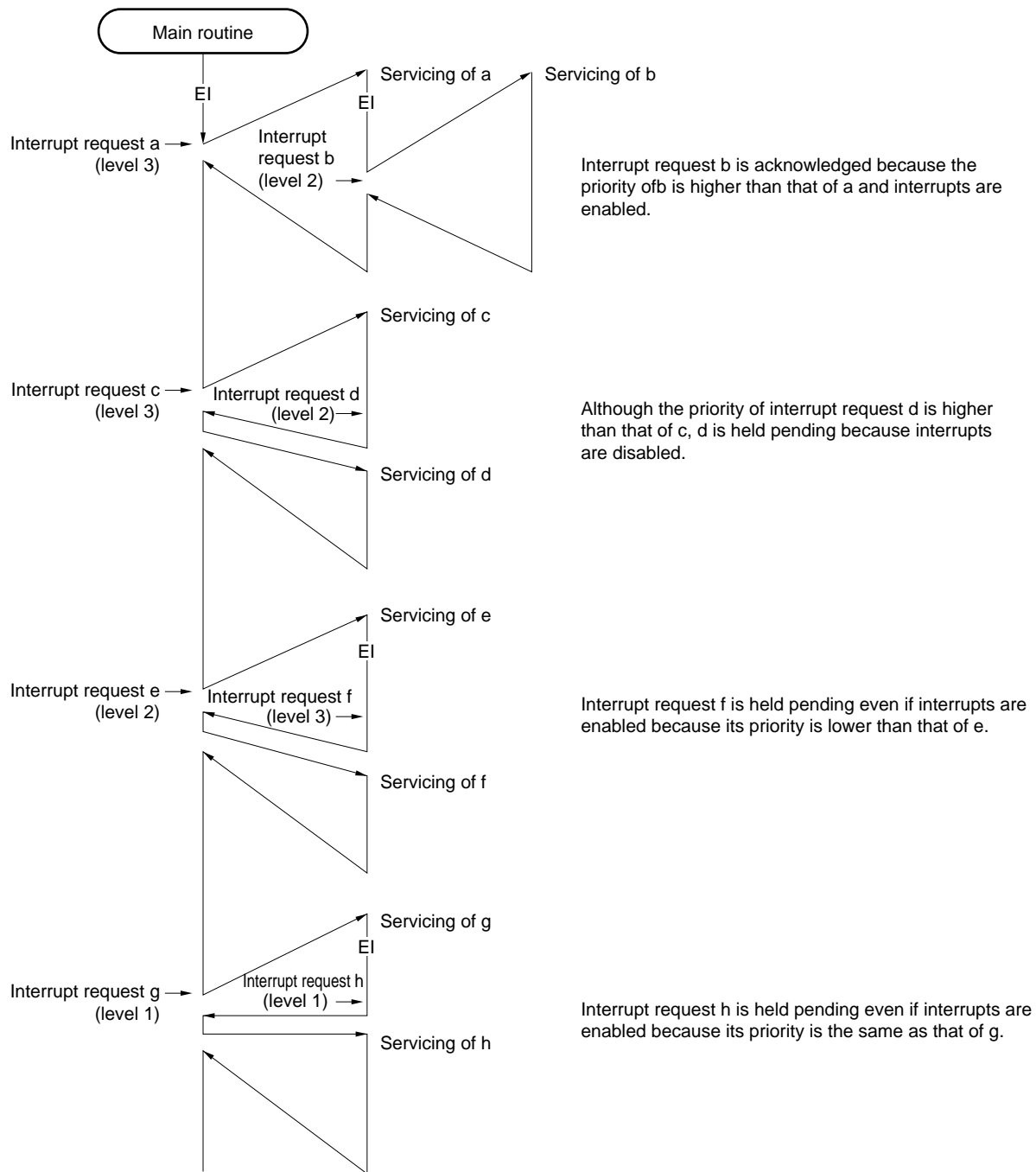
### 7.3.3 Priorities of maskable interrupts

The V850E/MS1 provides multiple interrupt servicing whereby an interrupt is acknowledged while another interrupt is being serviced. 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 which are specified by the interrupt priority level specification bit (xxPRn) of the interrupt control register (xxICn). When two or more interrupts having the same priority level specified by the xxPRn bit are generated at the same time, interrupts are serviced in order depending on the priority level allocated to each interrupt request type (default priority level) beforehand. For more information, refer to **Table 7-1 Interrupt List**. The programmable priority control customizes interrupt requests into eight levels by setting the priority level specification flag.

Note that when an interrupt request is acknowledged, the ID flag of the 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.

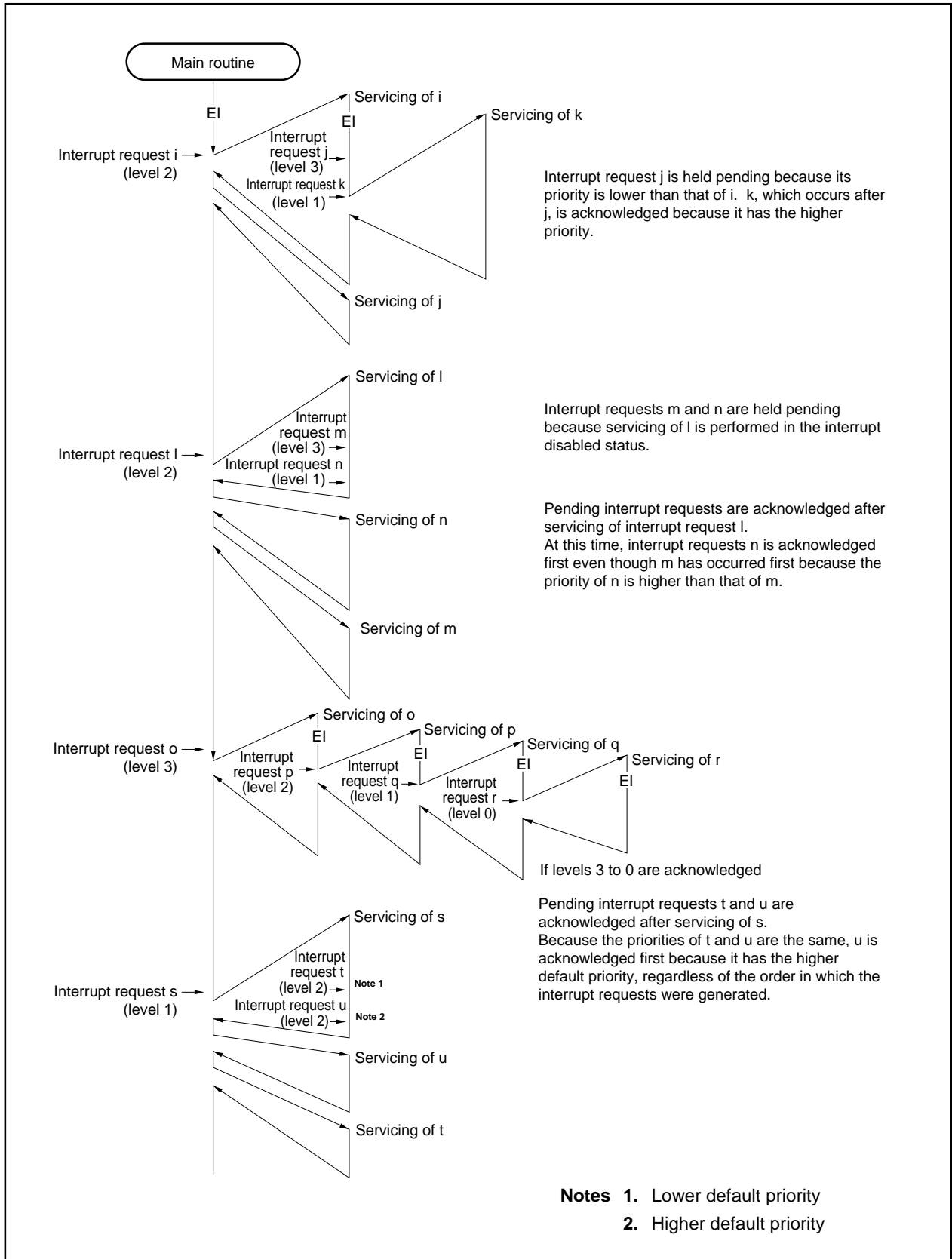
**Figure 7-7. Example of Servicing in Which Another Interrupt Request Is Issued While Interrupt Is Being Serviced (1/2)**

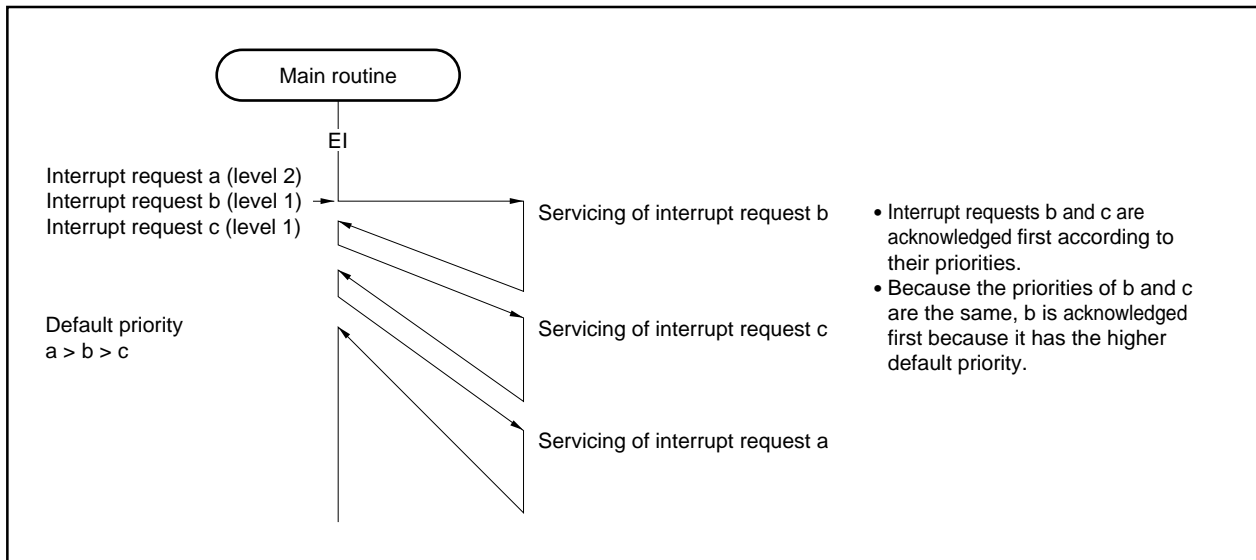


- Remarks**
1. a to u in the figure are the names of interrupt requests shown for the sake of explanation.
  2. The default priority in the figure indicates the relative priority between two interrupt requests.

**Caution** The values of the EIPC and EIPSW registers must be saved before executing multiple interrupts.

**Figure 7-7. Example of Servicing in Which Another Interrupt Request Is Issued While Interrupt Is Being Serviced (2/2)**

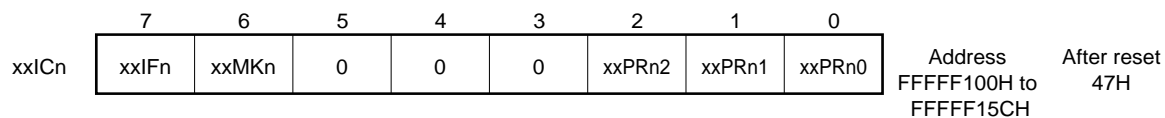


**Figure 7-8. Example of Servicing Interrupt Requests Simultaneously Generated**

### 7.3.4 Interrupt control register (xxICn)

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

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



Bit Position	Bit Name	Function																																				
7	xxIFn	<p>Interrupt Request Flag</p> <p>This is an interrupt request flag.</p> <p>0: Interrupt request not issued</p> <p>1: Interrupt request issued</p> <p>The flag xxIFn is reset automatically by the hardware if an interrupt request is received.</p>																																				
6	xxMKn	<p>Mask Flag</p> <p>This is an interrupt mask flag.</p> <p>0: Enables interrupt servicing</p> <p>1: Disables interrupt servicing (pending)</p>																																				
2 to 0	xxPRn2 to xxPRn0	<p>Priority</p> <p>8 levels of priority order are specified in each interrupt.</p> <table><tr><th>xxPRn2</th><th>xxPRn1</th><th>xxPRn0</th><th>Interrupt Priority Specification Bit</th></tr><tr><td>0</td><td>0</td><td>0</td><td>Specifies level 0 (highest).</td></tr><tr><td>0</td><td>0</td><td>1</td><td>Specifies level 1.</td></tr><tr><td>0</td><td>1</td><td>0</td><td>Specifies level 2.</td></tr><tr><td>0</td><td>1</td><td>1</td><td>Specifies level 3.</td></tr><tr><td>1</td><td>0</td><td>0</td><td>Specifies level 4.</td></tr><tr><td>1</td><td>0</td><td>1</td><td>Specifies level 5.</td></tr><tr><td>1</td><td>1</td><td>0</td><td>Specifies level 6.</td></tr><tr><td>1</td><td>1</td><td>1</td><td>Specifies level 7 (lowest).</td></tr></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, P10 to P13, CM, CS, SE, SR, ST, AD, DMA)  
 n: Peripheral unit number (None, or 0 to 3, 10 to 13, 40, 41).

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

**Table 7-2. Interrupt Control Register Addresses and Bits**

Address	Register	Bit							
		7	6	5	4	3	2	1	0
FFFFF100H	OVIC10	OVIF10	OVMK10	0	0	0	OVPR102	OVPR101	OVPR100
FFFFF102H	OVIC11	OVIC11	OVMK11	0	0	0	OVPR112	OVPR111	OVPR110
FFFFF104H	OVIC12	OVIF12	OVMK12	0	0	0	OVPR122	OVPR121	OVPR120
FFFFF106H	OVIC13	OVIF13	OVMK13	0	0	0	OVPR132	OVPR131	OVPR130
FFFFF10CH	CMIC40	CMIF40	CMMK40	0	0	0	CMPR402	CMPR401	CMPR400
FFFFF10EH	CMIC41	CMIF41	CMMK41	0	0	0	CMPR412	CMPR411	CMPR410
FFFFF110H	P10IC0	P10IF0	P10MK0	0	0	0	P10PR02	P10PR01	P10PR00
FFFFF112H	P10IC1	P10IF1	P10MK1	0	0	0	P10PR12	P10PR11	P10PR10
FFFFF114H	P10IC2	P10IF2	P10MK2	0	0	0	P10PR22	P10PR21	P10PR20
FFFFF116H	P10IC3	P10IF3	P10MK3	0	0	0	P10PR32	P10PR31	P10PR30
FFFFF118H	P11IC0	P11IF0	P11MK0	0	0	0	P11PR02	P11PR01	P11PR00
FFFFF11AH	P11IC1	P11IF1	P11MK1	0	0	0	P11PR12	P11PR11	P11PR10
FFFFF11CH	P11IC2	P11IF2	P11MK2	0	0	0	P11PR22	P11PR21	P11PR20
FFFFF11EH	P11IC3	P11IF3	P11MK3	0	0	0	P11PR32	P11PR31	P11PR30
FFFFF120H	P12IC0	P12IF0	P12MK0	0	0	0	P12PR02	P12PR01	P12PR00
FFFFF122H	P12IC1	P12IF1	P12MK1	0	0	0	P12PR12	P12PR11	P12PR10
FFFFF124H	P12IC2	P12IF2	P12MK2	0	0	0	P12PR22	P12PR21	P12PR20
FFFFF126H	P12IC3	P12IF3	P12MK3	0	0	0	P12PR32	P12PR31	P12PR30
FFFFF128H	P13IC0	P13IF0	P13MK0	0	0	0	P13PR02	P13PR01	P13PR00
FFFFF12AH	P13IC1	P13IF1	P13MK1	0	0	0	P13PR12	P13PR11	P13PR10
FFFFF12CH	P13IC2	P13IF2	P13MK2	0	0	0	P13PR22	P13PR21	P13PR20
FFFFF12EH	P13IC3	P13IF3	P13MK3	0	0	0	P13PR32	P13PR31	P13PR30
FFFFF140H	DMAIC0	DMAIF0	DMAMK0	0	0	0	DMAPR02	DMAPR01	DMAPR00
FFFFF142H	DMAIC1	DMAIF1	DMAMK1	0	0	0	DMAPR12	DMAPR11	DMAPR10
FFFFF144H	DMAIC2	DMAIF2	DMAMK2	0	0	0	DMAPR22	DMAPR21	DMAPR20
FFFFF146H	DMAIC3	DMAIF3	DMAMK3	0	0	0	DMAPR32	DMAPR31	DMAPR30
FFFFF148H	CSIC0	CSIF0	CSMK0	0	0	0	CSPR02	CSPR01	CSPR00
FFFFF14AH	CSIC1	CSIF1	CSMK1	0	0	0	CSPR12	CSPR11	CSPR10
FFFFF150H	SEIC0	SEIF0	SEMK0	0	0	0	SEPR02	SEPR01	SEPR00
FFFFF152H	SRIC0	SRIF0	SRMK0	0	0	0	SRPR02	SRPR01	SRPR00
FFFFF154H	STIC0	STIF0	STMK0	0	0	0	STPR02	STPR01	STPR00
FFFFF156H	SEIC1	SEIF1	SEMK1	0	0	0	SEPR12	SEPR11	SEPR10
FFFFF158H	SRIC1	SRIF1	SRMK1	0	0	0	SRPR12	SRPR11	SRPR10
FFFFF15AH	STIC1	STIF1	STMK1	0	0	0	STPR12	STPR11	STPR10
FFFFF15CH	ADIC	ADIF	ADMK	0	0	0	ADPR2	ADPR1	ADPR0





### 7.3.7 Noise elimination

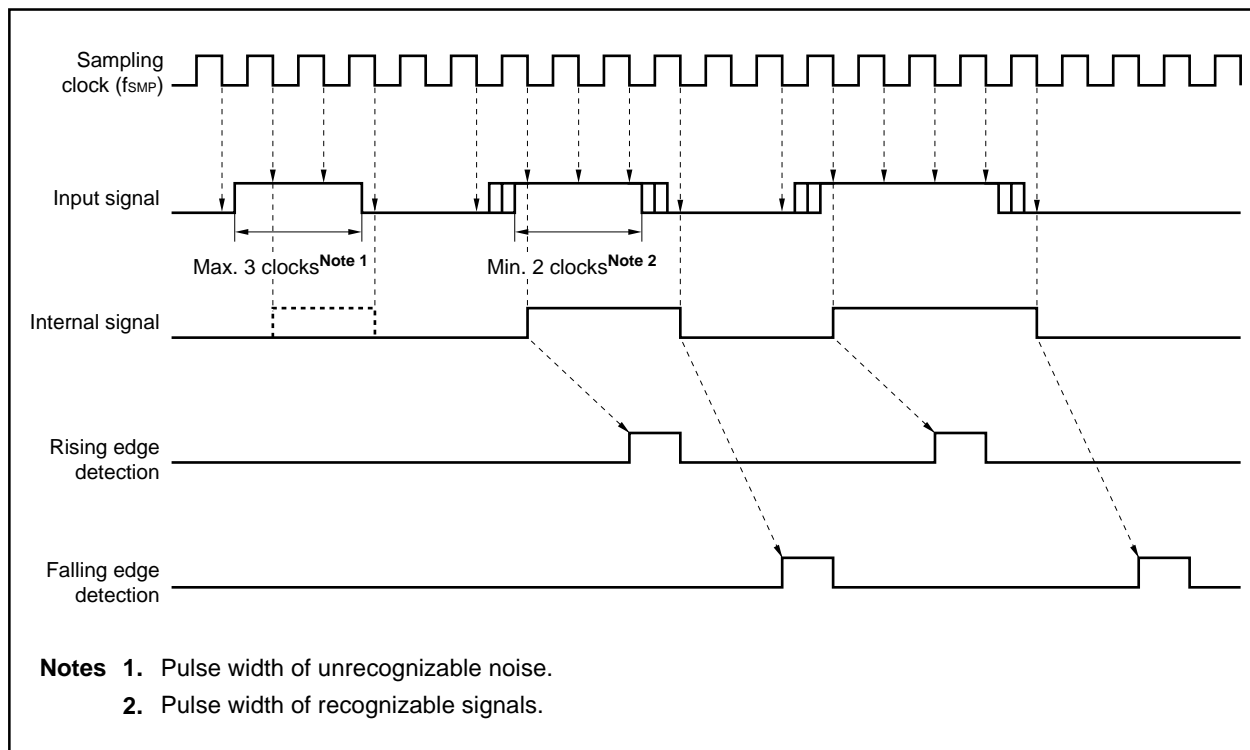
Digital noise elimination circuits are added to each of the INTPn0 to INTPn3, INTP130, TI3, and TCLR10 to TCLR12 pins ( $n = 10, 11$ ). Using these circuits, these pins' input level is sampled each sampling clock cycle ( $f_{SMP}$ ). If the same level cannot be detected 3 times consecutively in the sampling results, that input pulse is removed as noise.

The noise elimination time at each pin is shown below.

Pin	Sampling Clock ( $f_{SMP}$ )	Noise Elimination Time
TCLR10 to TCLR12	$\phi$	$2 \times \phi$ to $3 \times \phi$
TI13	$\phi$	
INTP100 to INTP103, INTP110 to INTP113, INTP130	$\phi$	

**Remark**  $\phi$ : Internal system clock

**Figure 7-9. Example of Noise Elimination Timing**



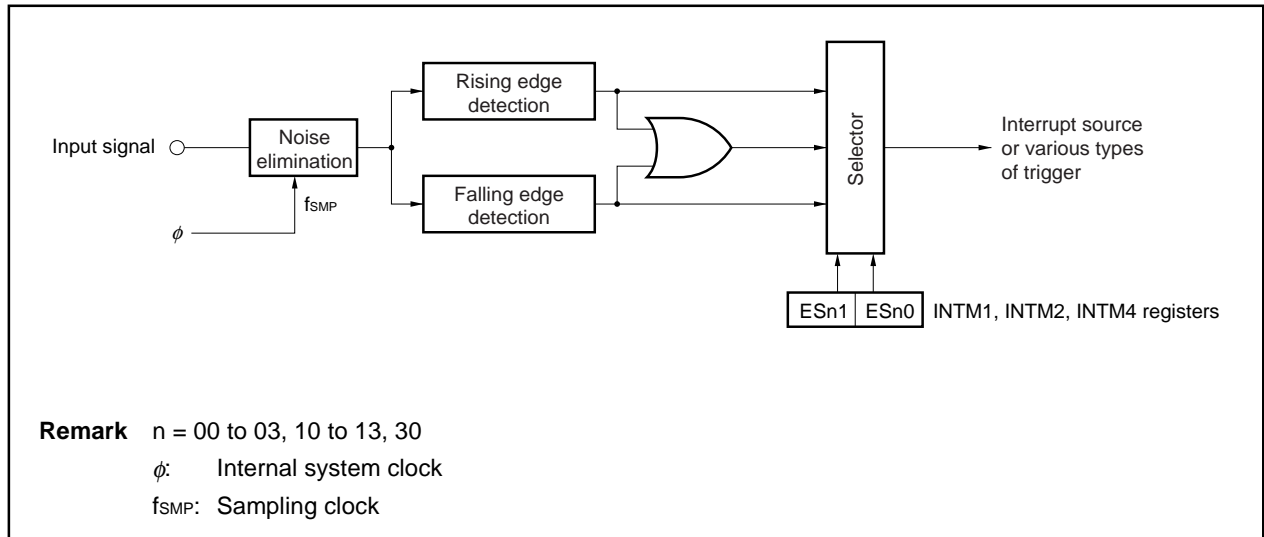
- Cautions**
1. If the input pulse width is between 2 and 3 sampling clocks, whether the input pulse is detected as a valid edge or eliminated as a noise is indefinite.
  2. To securely detect the level as a pulse, the same level input of 3 sampling clocks or more is required.
  3. When noise is generated in synchronization with a sampling clock, this may not be recognized as noise. In this case, eliminate the noise by attaching a filter to the input pin.

### 7.3.8 Edge detection function

The valid edge of pins INTPn0 to INTPn3 and INTP130 can be selected by program. The valid edge that can be selected is one of the following (n = 10, 11).

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

Edge detected INTPn0 to INTPn3 and INTP130 signals become interrupt factors or capture triggers. The block diagram of the edge detectors for these pins is shown below.



Valid edges are specified in external interrupt mode registers 1, 2, 4 (INTM1, INTM2, INTM4).

#### (1) External interrupt mode registers 1, 2, 4 (INTM1, INTM2, INTM4)

These are registers that specify the valid edge for external interrupt requests (INTP100 to INTP103, INTP110 to INTP113, INTP130), by external pins. The correspondence between each register and the external interrupt requests which that register controls is shown below.

- INTM1: INTP100 to INTP103
- INTM2: INTP110 to INTP113
- INTM4: INTP130

The valid edge can be specified independently for each pin, as the rising edge, the falling edge or both the rising and falling edges.

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

	7	6	5	4	3	2	1	0		
INTM1	ES031	ES030	ES021	ES020	ES011	ES010	ES001	ES000	Address	After reset
Control pins	INTP103		INTP102		INTP101		INTP100		FFFFF182H	00H
INTM2	ES131	ES130	ES121	ES120	ES111	ES110	ES101	ES100	FFFFF184H	00H
Control pins	INTP113		INTP112		INTP111		INTP110			
INTM4 <sup>Note</sup>	0	0	0	0	0	0	ES301	ES300	FFFFF188H	00H
Control pins								INTP130		

Bit Position	Bit Name	Function															
7 to 0	ESmn1, ESmn0 (m = 3, 1, 0, n = 3 to 0)	Edge Select Specifies the valid edge of the INTP1mn pins. <table> <tr> <th>ESmn1</th><th>ESmn0</th><th>Operation</th></tr> <tr> <td>0</td><td>0</td><td>Falling edge</td></tr> <tr> <td>0</td><td>1</td><td>Rising edge</td></tr> <tr> <td>1</td><td>0</td><td>RFU (reserved)</td></tr> <tr> <td>1</td><td>1</td><td>Both the rising and falling edges</td></tr> </table>	ESmn1	ESmn0	Operation	0	0	Falling edge	0	1	Rising edge	1	0	RFU (reserved)	1	1	Both the rising and falling edges
ESmn1	ESmn0	Operation															
0	0	Falling edge															
0	1	Rising edge															
1	0	RFU (reserved)															
1	1	Both the rising and falling edges															

**Note** Be sure to set bits 2 to 7 of INTM4 to 0.

## 7.4 Software Exception

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

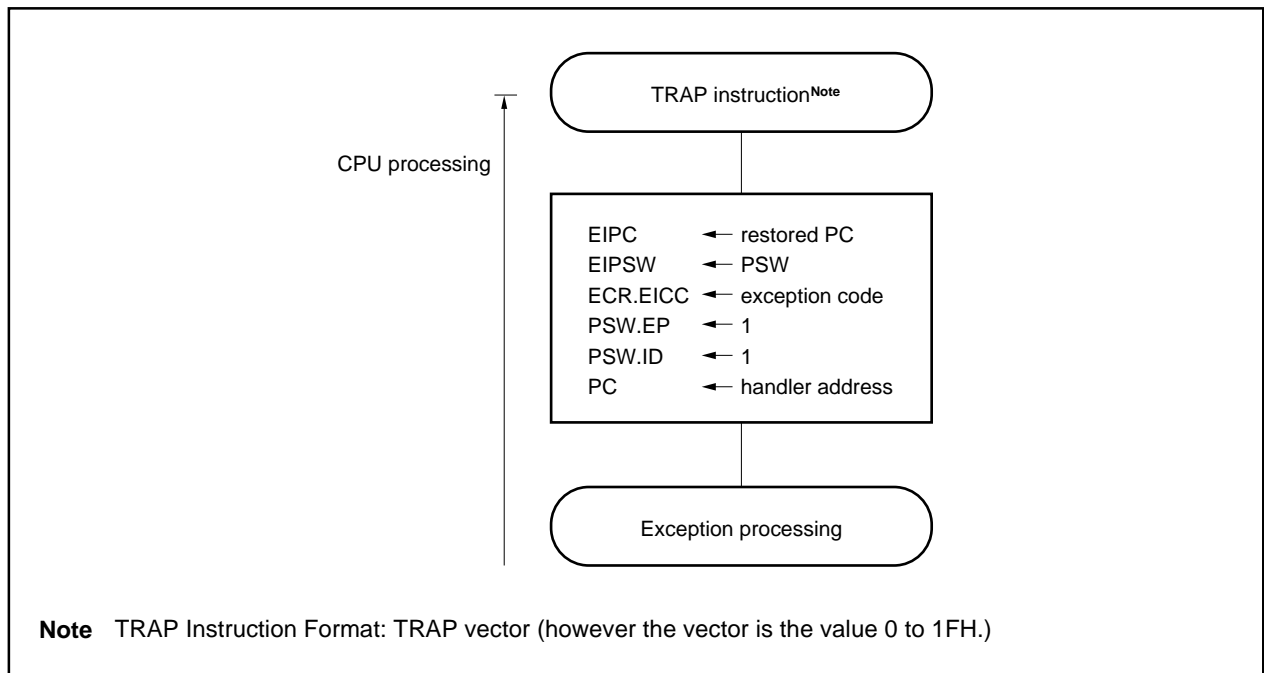
### 7.4.1 Operation

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 the PSW.
- (5) Sets the handler address (00000040H or 00000050H) corresponding to the software exception to the PC, and transfers control.

Figure 7-10 illustrates how a software exception is processed.

**Figure 7-10. Software Exception Processing**



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

### 7.4.2 Restore

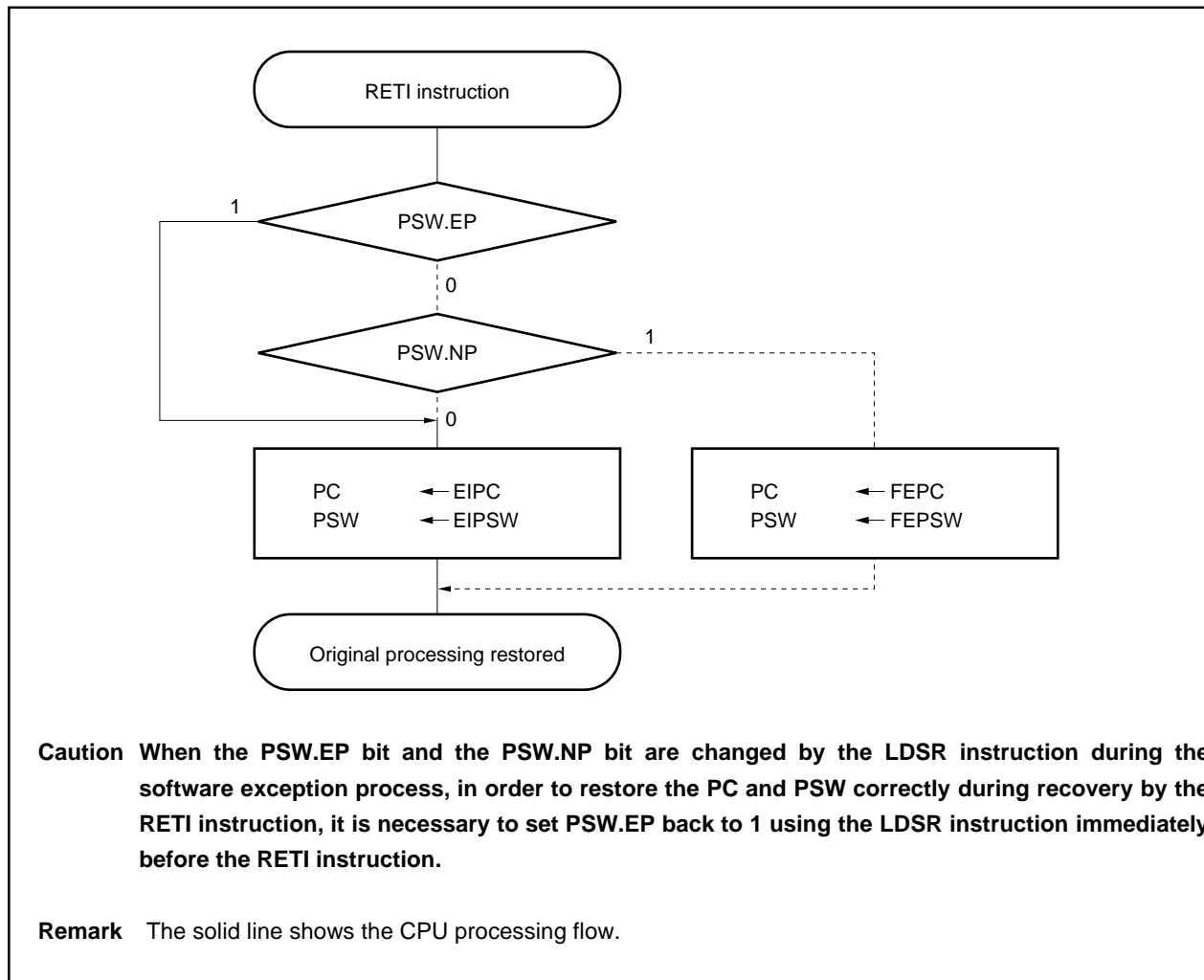
To restore from the software exception processing, the RETI instruction is used.

By executing the RETI instruction, the CPU carries out the following processing and shifts control to the restored PC's address.

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

Figure 7-11 illustrates the processing of the RETI instruction.

**Figure 7-11. RETI Instruction Processing**





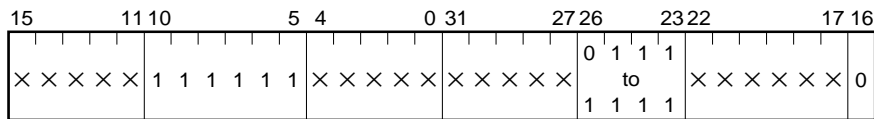
## 7.5 Exception Trap

The exception trap is an interrupt that is requested when illegal execution of an instruction takes place. In the V850E/MS2, an illegal op code exception (ILGOP: ILleGal Opcode trap) is considered an exception trap.

An illegal op code exception is generated in the case where the sub op code of the following instruction is an illegal op code when execution of that instruction is attempted.

### 7.5.1 Illegal op code definition

The illegal op code has a 32-bit long instruction format: bits 10 to 5 are 11111B and bits 26 to 23 are 0111B to 1111B, with bit 16 defined as an optional instruction code, 0B.



×: don't care

**Caution** Since it is possible to assign this instruction to an illegal op code in the future, it is recommended that it not be used.



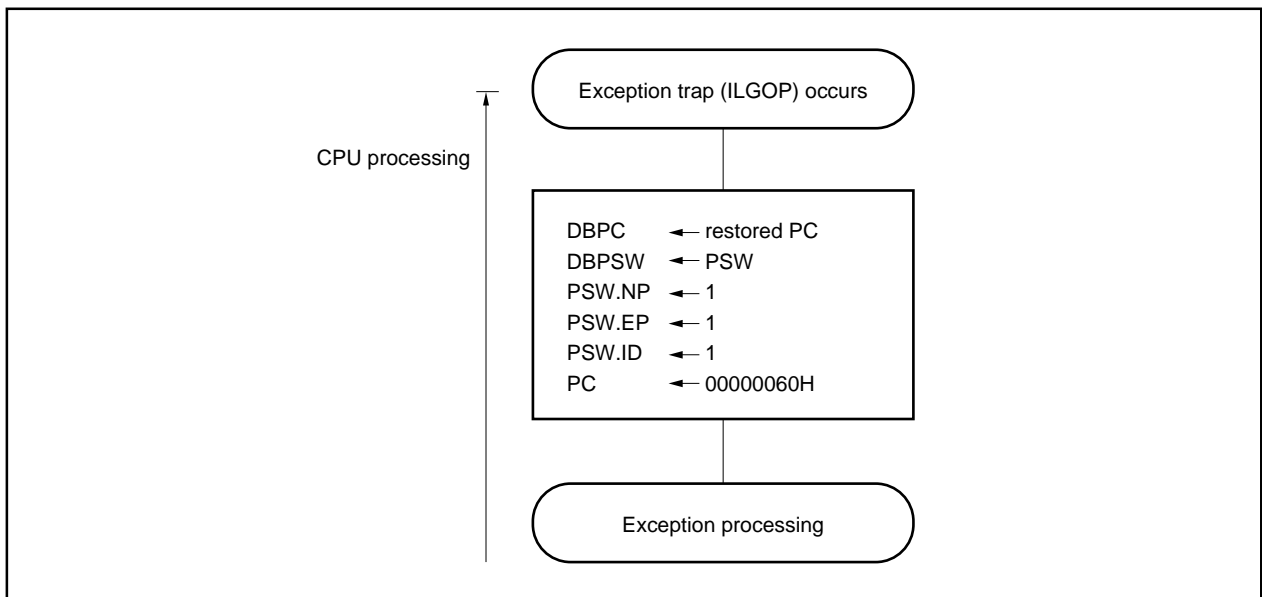
### 7.5.2 Operation

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

- (1) Saves the restored PC to DBPC.
- (2) Saves the current PSW to DBPSW.
- (3) Sets the NP, EP and ID bits of PSW.
- (4) Sets the handler address (00000060H) corresponding to the exception trap to the PC, and transfers control.

Figure 7-12 illustrates how the exception trap is processed.

**Figure 7-12. Exception Trap Processing**



### 7.5.3 Restore

Recovery from an exception trap is not possible. Perform system reset by  $\overline{\text{RESET}}$  input.

## 7.6 Multiple Interrupt Servicing Control

Multiple interrupt servicing control is a process by which the interrupt request currently being serviced can be interrupted during servicing if there is an interrupt request with a higher priority level, and the higher priority interrupt request is acknowledged and serviced first.

If there is an interrupt request with a lower priority level than the interrupt request currently being processed, that interrupt request is held pending.

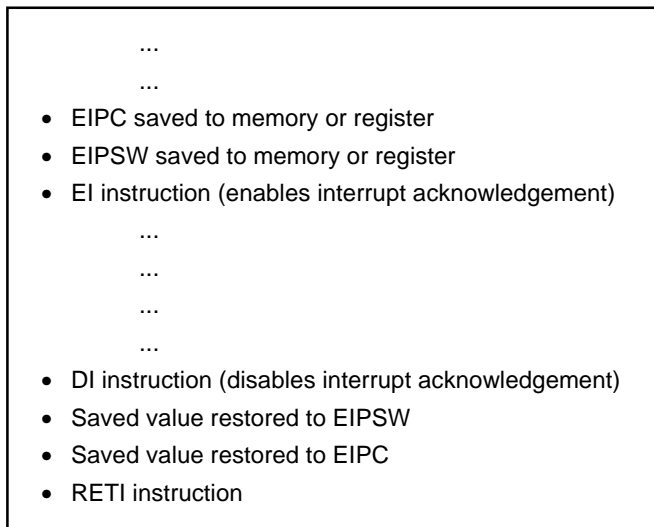
Maskable interrupt multiple processing control is executed when an interrupt has an enable status (ID = 0). Thus, if multiple interrupts are executed, it is necessary to have an interrupt enable status (ID = 0) even for an interrupt servicing routine.

If a maskable interrupt or a software exception is generated in a maskable interrupt or software exception service program, it is necessary to save EIPC and EIPSW.

This is accomplished by the following procedure.

### (1) To acknowledge maskable interrupts in a service program

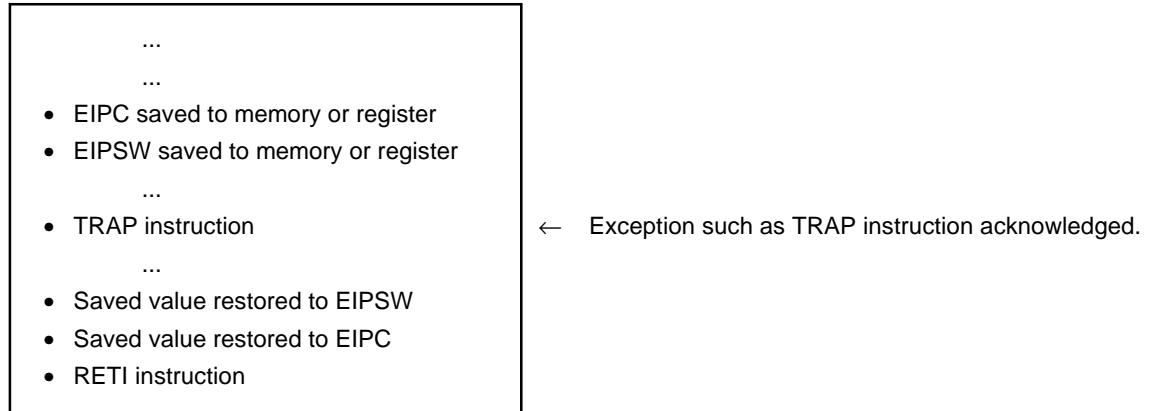
Service program of maskable interrupt or exception



← Maskable interrupt acknowledgement

**(2) To generate an exception in a service program**

Service program of maskable interrupt or exception



The priority order for multiple interrupt servicing control has 8 levels, from 0 to 7 for each maskable interrupt request (0 is the highest priority), which can be set as desired via software. The priority order level is set with the xxPRn0 to xxPRn2 bits of the interrupt control request register (xxICn), which is provided for each maskable interrupt request. At system reset time, an interrupt request is masked by the xxMKn bit and the priority order is set to level 7 by the xxPRn0 to xxPRn2 bits.

The priority order of maskable interrupts is as follows.

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

Interrupt servicing that has been suspended as a result of multiple servicing control is resumed after the interrupt servicing of the higher priority has been completed and the RETI instruction has been executed.

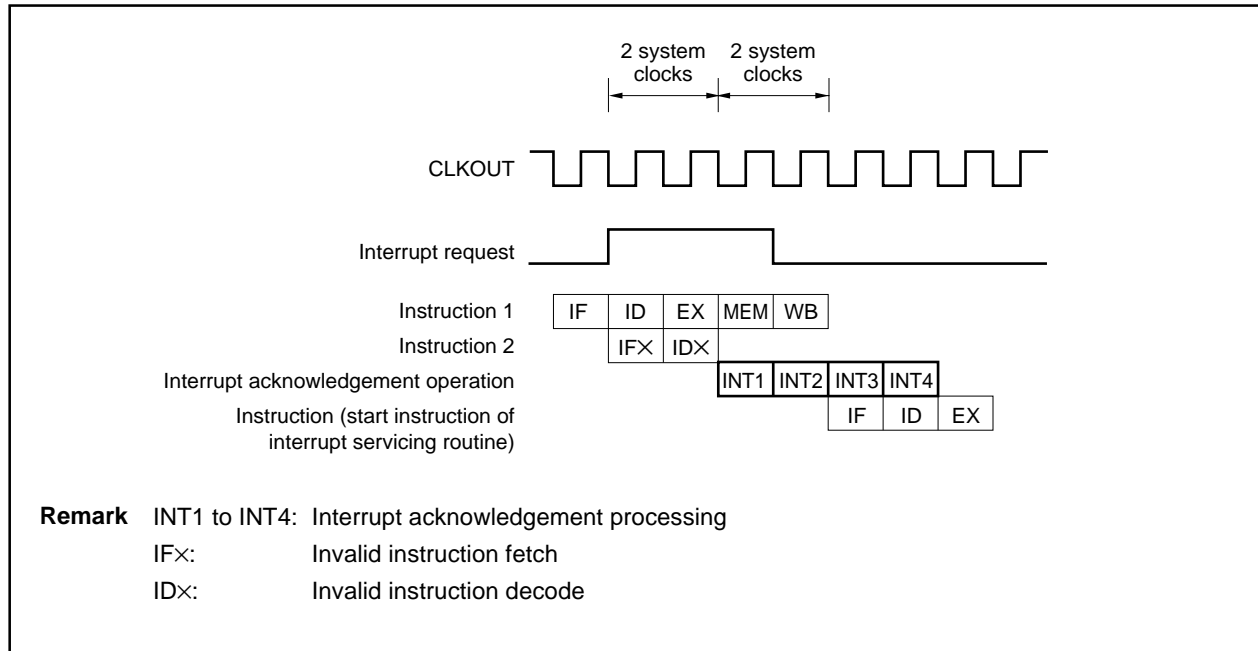
A pending interrupt request is acknowledged after the current interrupt servicing has been completed and the RETI instruction has been executed.

**Caution** In the non-maskable interrupt servicing routine (time until the RETI instruction is executed), maskable interrupts are not acknowledged but are held pending.

## 7.7 Interrupt Latency Time

The following table describes the V850E/MS1 interrupt latency time (from interrupt generation to start of interrupt servicing).

**Figure 7-13. Pipeline Operation at Interrupt Request Acknowledgement (Outline)**



Interrupt Latency Time (Internal System Clock)			Condition
	Internal Interrupt	External Interrupt	
Minimum	4	6	The following cases are exceptions. <ul style="list-style-type: none"> <li>• In IDLE/software STOP mode</li> <li>• External bus is accessed</li> <li>• Two or more interrupt request non-sample instructions are executed in succession</li> <li>• Access to interrupt control register</li> </ul>
Maximum	10	12	

## 7.8 Periods in Which Interrupt Is Not Acknowledged

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

The interrupt request non-sampling instructions are as follows.

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (vs. PSW)
- The store instruction for the interrupt control register (xxICn) and command register (PRCMD)

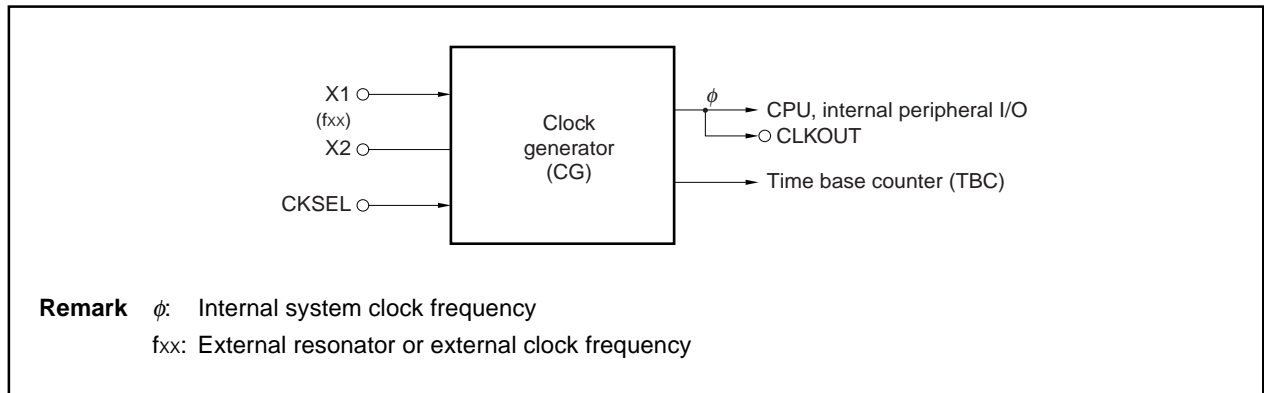
## CHAPTER 8 CLOCK GENERATOR FUNCTIONS

The clock generator (CG) generates and controls the internal system clock ( $\phi$ ) which is supplied to each internal unit, of which the CPU is the primary unit.

### 8.1 Features

- Multiplier function using a PLL (phase locked loop) synthesizer
- Clock Source
  - Oscillation by connecting an oscillator:  $f_{xx} = \phi/5$
  - External clock:  $f_{xx} = 2 \times \phi, \phi/5$
- Power save control
  - HALT mode
  - IDLE mode
  - Software STOP mode
  - Clock output inhibit function
- Internal system clock output function

### 8.2 Configuration



### 8.3 Input Clock Selection

The clock generator is configured from an oscillator and a PLL synthesizer. If, for example a 6 MHz crystal resonator or ceramic resonator is connected to pins X1 and X2, an internal system clock ( $\phi$ ) of 30 MHz can be generated.

Also, an external clock can be input directly to the oscillator. In this case, input a clock signal to the X1 pin only and leave the X2 pin open.

Two types of mode, a PLL mode and a direct mode, are provided as the basic operation modes for the clock generator. Selection of the operation mode is done by the CKSEL pin. The input of this pin latches at reset time.

CKSEL	Operation Mode
0	PLL mode
1	Direct mode

**Caution** Fix the input level of the CKSEL pin before use. If it is switched during operation, there is a possibility of malfunction occurring.

#### 8.3.1 Direct mode

In the direct mode, an external clock with double the internal system clock's frequency is input. Mainly, the V850E/MS2 is used in application systems where it operates at relatively low frequencies. In consideration of EMI countermeasures, if the external clock frequency ( $f_{xx}$ ) is 32 MHz (internal system clock ( $\phi$ ) = 16 MHz) or greater, the PLL mode is recommended.

**Caution** In the direct mode, be sure to input an external clock (do not connect an external resonator).

#### 8.3.2 PLL mode

In the PLL mode, by connecting an external resonator or inputting an external clock and multiplying this clock by the PLL synthesizer, an internal system clock ( $\phi$ ) is generated.

At reset time, an internal system clock ( $\phi$ ) which is 5 times the frequency of the input clock's frequency ( $f_{xx}$ ) ( $5 \times f_{xx}$ ), is generated.

In the PLL mode, if the clock supply from an external resonator or external clock source stops, the internal system clock ( $\phi$ ) continues to operate based on the self-propelled frequency of the clock generator's internal voltage controlled oscillator (VCO). In this case,  $\phi$  = approx. 1 MHz (target). However, do not devise an application method in which you expect to use this self-propelled frequency.

**Example** Clock used when in the PLL mode

System Clock Frequency ( $\phi$ ) [MHz]	External Resonator/External Clock Frequency ( $f_{xx}$ ) [MHz]
30.000	6.0000
25.000	5.0000
20.000	4.0000
16.384	3.2768

### 8.3.3 Clock control register (CKC)

When in the PLL mode, this is an 8-bit register which controls the internal system clock frequency ( $\phi$ ), and it can be written to only by a specific combination of instruction sequences so that it cannot be rewritten easily by mistake due to program runaway.

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

**Caution** When in the direct mode, do not change the setting of this register.

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

Bit Position	Bit Name	Function															
1, 0	CKDIV1, CKDIV0	Clock Divide Sets the internal system clock frequency ( $\phi$ ) when in the PLL mode. <table border="1"> <tr> <th>CKDIV1</th><th>CKDIV0</th><th>Internal System Clock (<math>\phi</math>)</th></tr> <tr> <td>0</td><td>0</td><td><math>5 \times f_{xx}</math></td></tr> <tr> <td>0</td><td>1</td><td>Setting prohibited</td></tr> <tr> <td>1</td><td>0</td><td><math>f_{xx}</math></td></tr> <tr> <td>1</td><td>1</td><td><math>f_{xx}/2</math></td></tr> </table>	CKDIV1	CKDIV0	Internal System Clock ( $\phi$ )	0	0	$5 \times f_{xx}$	0	1	Setting prohibited	1	0	$f_{xx}$	1	1	$f_{xx}/2$
CKDIV1	CKDIV0	Internal System Clock ( $\phi$ )															
0	0	$5 \times f_{xx}$															
0	1	Setting prohibited															
1	0	$f_{xx}$															
1	1	$f_{xx}/2$															

The sequence of setting data to this register is the same as for the power save control register (PSC). However, the restrictions shown in **Remark 2** of **3.4.9 Specific registers** do not apply. For details, refer to **8.5.2 Control registers**.

**Example** Clock generator setting

Operation Mode	CKSEL Pin	CKC Register		Input Clock (f <sub>xx</sub> )	Internal System Clock (ϕ)
		CKDIV1 Bit	CKDIV0 Bit		
Direct mode	High-level input	0	0	16 MHz	8 MHz
PLL mode	Low-level input	0	0	6 MHz	30 MHz
		1	0	6 MHz	6 MHz
		1	1	6 MHz	3 MHz
Other than above				Setting prohibited	

## 8.4 PLL Lockup

Lockup time (frequency stabilization time) is the amount of time from immediately after the software STOP mode is released after the power is turned on, until the phase locks at the proper frequency and becomes stable. The state until this stabilization occurs is called the unlocked state and the stabilized state is called the locked state.

There is a LOCK flag which reflects the PLL's frequency stabilization state, and a PRERR flag which shows when a protection error occurs, in the system status register (SYS).

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	LOCK	Address FFFFF078H	After reset 0000000xB

Bit Position	Bit Name	Function
0	LOCK	Lock Status Flag This is an exclusive read flag and shows the PLL's locked state. As long as the lockup state is maintained, it is kept at 0, and is not initialized when system reset occurs. 0: Indicates that the PLL is in a locked state. 1: Indicates that the PLL is not locked (in an unlocked state).

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

If the clock stops, the power fails, or some other factor occurs to cause the unlocked state, in control processing which depends on software execution speed such as real-time processing, be sure to begin processing after judging the LOCK flag by software immediately after operation starts, and after waiting for the clock to stabilize again.

On the other hand, for static processing such as setting of internal hardware, or initialization of register data and memory data, it is possible to execute these without waiting for the LOCK flag to be reset.

The relationship between the oscillation stabilization time (the time from when the resonator starts to oscillate until the input waveform stabilizes) when a resonator is used, and the PLL lockup time (the time until the frequency is stabilized) is shown below.

Oscillation stabilization time < PLL lockup time



## 8.5 Power Saving Control

### 8.5.1 Outline

The V850E/MS1 standby function comprises the following three modes:

#### (1) HALT mode

In this mode, the clock generator (oscillator and PLL synthesizer) continues to operate, but the CPU's operation clock stops. Supply of the clock to the other internal peripheral functions is continued. Through intermittent operation by combining with the normal operating mode, the system's total power consumption can be reduced.

The system is switched to the HALT mode via an exclusive instruction (the HALT instruction).

#### (2) IDLE mode

In this mode, the clock generator (oscillator and PLL synthesizer) continues to operate, but supply of the internal system clock is stopped, which causes the system overall to stop.

When releasing the system from the IDLE mode, it is not necessary to secure the oscillation stabilization time of the oscillator, so it is possible to switch to normal operation at high speed.

The system enters the IDLE mode in accordance with the settings in the PSC register (specific register).

The IDLE mode is positioned midway between the software STOP mode and the HALT mode in relation to clock stabilization time and current consumption and is used for cases where the low current consumption mode is used and where it is desired to eliminate the clock stabilization time after it is released.

#### (3) Software STOP mode

In this mode, the clock generator (oscillator and PLL synthesizer) is stopped and the system overall is stopped, thus entering an ultra-low power consumption state where only leak current is lost.

It is possible to enter the software STOP mode by setting the PSC register (specific register).

##### (a) When in the PLL Mode

By setting the register by software, you can enter the software STOP mode. At the same time the oscillator stops, the PLL synthesizer's clock output stops. After releasing the software STOP mode, it is necessary to secure oscillation stabilization time for the oscillator for a period of time until the system clock stabilizes. Also, depending on the program, PLL lockup time may be required.

#### (4) Clock output inhibit mode

Internal system clock output from the CLKOUT pin is prohibited.

The operation of the clock generator in normal operation, and in the HALT, IDLE, and software STOP modes is shown in Table 8-1.

By combining each of the modes and by switching modes according to the required usage, it is possible to realize an effective low power consumption system.

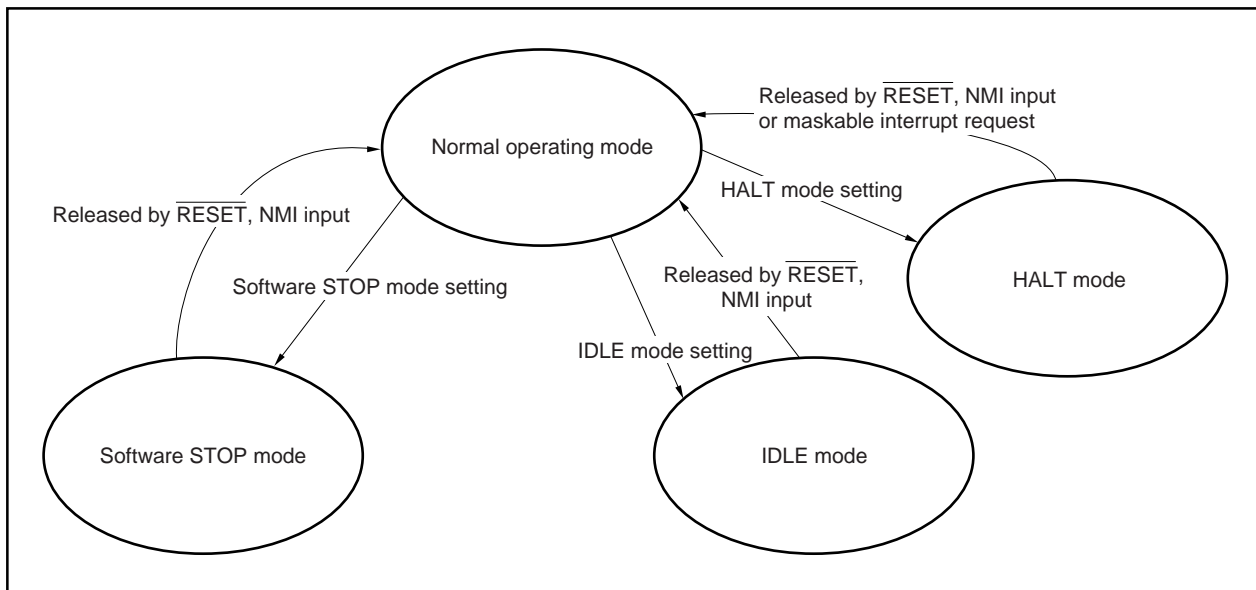
Table 8-1. Clock Generator Operation by Power Save Control

Clock Source		Power Save Mode	Oscillator (OSC)	PLL Synthesizer	Supply of Clock to Internal Peripheral I/O	Supply of Clock to the CPU
PLL mode	Oscillation by resonator	(During normal operation)	○	○	○	○
		HALT mode	○	○	○	×
		IDLE mode	○	○	×	×
		Software STOP mode	×	×	×	×
	External clock	(During normal operation)	×	○	○	○
		HALT mode	×	○	○	×
		IDLE mode	×	○	×	×
		Software STOP mode	×	×	×	×
Direct mode		(During normal operation)	×	×	○	○
		HALT mode	×	×	○	×
		IDLE mode	×	×	×	×
		Software STOP mode	×	×	×	×

○: Operating

×: Stopped

Figure 8-1. Power Save Mode State Transition Diagram



## 8.5.2 Control registers

## (1) Power save control register (PSC)

This is an 8-bit register that controls the power save mode.

This is one of the specific registers and is active only when accessed by a specific sequence during a write operation. For details, refer to **3.4.9 Specific registers**.

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

	7	6	5	4	3	2	1	0		
PSC	DCLK1	DCLK0	TBCS	CESEL	0	IDLE	STP	0	Address FFFFF070H	After reset 00H

Bit Position	Bit Name	Function															
7, 6	DCLK1, DCLK0	Disable CLKOUT This specifies the CLKOUT pin's operating mode. <table border="1"> <tr> <th>DCLK1</th><th>DCLK0</th><th>Mode</th></tr> <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> </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 the time base counter clock. 0: $f_{xx}/2^8$ 1: $f_{xx}/2^9$ Details are shown in <b>8.6.2 Time base counter (TBC)</b> .															
4	CESEL	Crystal/External Select Specifies the function of pins X1 and X2. 0: An oscillator is connected to pins X1 and X2. 1: An external clock is connected to pin X1. If CESEL = 1, the oscillator's feedback loop is cut and current leakage is prevented when in the software STOP mode. Also, the oscillation stabilization time count by the time base counter (TBC) after the software STOP mode is released is not carried out.															
2	IDLE <sup>Note</sup>	IDLE Mode Specifies the IDLE mode. It enters the IDLE state if 1 is written. It is automatically reset (0) if the IDLE mode is released.															
1	STP <sup>Note</sup>	STOP Mode Specifies the software STOP mode. It enters the STOP state if 1 is written. It is automatically reset (0) if the software STOP mode is released.															

**Note** If the IDLE bit is set at 1 and the STP bit is also set at 1, the system enters the software STOP mode.

### 8.5.3 HALT mode

#### (1) Setting and operating state

In this mode, the clock generator (oscillator and PLL synthesizer) continues to operate, but the CPU's operation clock stops. Supply of the clock to other internal peripheral I/O functions is continued and their operation continues. By setting the HALT mode during the time when CPU is idle, the system's total power consumption can be reduced.

Switching to the HALT mode is accomplished by executing the HALT instruction.

In the HALT mode, program execution stops, but all the contents of all the registers, internal RAM, and ports are held in the state they were in just before the HALT mode was entered. Also, internal peripheral I/O (other than the ports) that is not dependent on CPU instruction processing continues operation. The state of each hardware unit when in the HALT mode is shown in Table 8-2.

**Remark** Even after HALT instruction execution, instruction fetch operations continue until the internal instruction prefetch queue becomes full. When the prefetch queue becomes full, it stops in the state shown in Table 8-2.

**Table 8-2. Operating States When in HALT Mode**

Function		Operating State
Clock generator		Operating
Internal system clock		Operating
CPU		Stop
Port		Hold
Internal peripheral I/O (except ports)		Operating
Internal data		All the CPU's registers, status, data, internal RAM contents and other internal data, etc. are retained in the state they were in before entering the HALT mode.
When in external expansion mode	D0 to D15	Operating
	A0 to A23	
	$\overline{RD}$ , $\overline{WE}$ , $\overline{OE}$ , $\overline{BCYST}$	
	$\overline{LWR}$ , $\overline{UWR}$ , $\overline{IORD}$ , $\overline{IOWR}$	
	$\overline{CS0}$ , $\overline{CS3}$ to $\overline{CS5}$	
	RAS3 to RAS5	
	$\overline{LCAS}$ , $\overline{UCAS}$	
	$\overline{HLD RQ}$	
	$\overline{HLD AK}$	
	$\overline{WAIT}$	
CLKOUT		Clock output (when not in clock output inhibit)

**(2) Releasing HALT mode**

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

**(a) Release by NMI pin input, maskable interrupt request**

The HALT mode is unconditionally released by NMI pin input or an unmasked maskable interrupt request regardless of the priority. However, if the HALT mode is set in an interrupt servicing 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 acknowledged. 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 acknowledged.

**Table 8-3. Operations After HALT Mode Is Released by Interrupt Request**

Releasing Source	Interrupt Enable (EI) State	Interrupt Disable (DI) State
NMI request	Branch to handler address	
Maskable interrupt request	Branch to the handler address or execute the next instruction.	Execute the next instruction.

**(b) Release by  $\overline{\text{RESET}}$  pin input**

This operation is the same as a normal reset operation.

### 8.5.4 IDLE mode

#### (1) Settings and operating state

In this mode, the clock generator (oscillator and PLL synthesizer) continues to operate, but supply of the internal system clock is stopped, which causes the system overall to stop.

When releasing the system from the IDLE mode, it is not necessary to secure the oscillation stabilization time of the oscillator, so it is possible to switch to normal operation at high speed.

The IDLE mode is entered by the setting of the PSC register (specific register), set through a store instruction (ST/SST instruction) or a bit operation instruction (SET1/CLR1/NOT1 instruction) (refer to **3.4.9 Specific registers**).

In the IDLE mode, program execution is stopped, but all the contents of all the registers, internal RAM, and ports are held. Operation of the internal peripheral I/O (except the ports) is also stopped.

The state of each hardware unit when in IDLE mode is as shown in Table 8-4.

**Table 8-4. Operating States When in IDLE Mode**

Function		Operating State
Clock generator		Operating
Internal system clock		Stop
CPU		Stop
Port		Hold
Internal peripheral I/O (except ports)		Stop
Internal data		All the CPU's registers, status, data, internal RAM contents and other internal data, etc. are retained in the state they were in before entering the HALT mode.
When in external expansion mode	D0 to D15	High-impedance
	A0 to A23	
	$\overline{RD}$ , $\overline{WE}$ , $\overline{OE}$ , $\overline{BCYST}$	
	$\overline{LWR}$ , $\overline{UWR}$ , $\overline{IORD}$ , $\overline{IOWR}$	High-level output
	$\overline{CS0}$ , $\overline{CS3}$ to $\overline{CS5}$	
	$\overline{RAS3}$ to $\overline{RAS5}$	Operating
	$\overline{LCAS}$ , $\overline{UCAS}$	
	$\overline{HLDRQ}$	Input (no sampling)
	$\overline{HLDAR}$	High-impedance
	$\overline{WAIT}$	Input (no sampling)
CLKOUT		Low-level output

**(2) Releasing IDLE mode**

The IDLE Mode is released by NMI pin input or  $\overline{\text{RESET}}$  pin input.

**(a) Release by NMI pin input**

This is acknowledged as a NMI request together with a release of the IDLE mode.

However, in cases where setting the system in the IDLE mode is included in the NMI servicing routine, the IDLE mode is released only, and this interrupt is not acknowledged. The interrupt request itself is held pending.

The interrupt servicing that is started when the IDLE mode is released by NMI pin input is treated in the same way as ordinary NMI interrupt servicing in an emergency, etc. (since the NMI interrupt handler's address is unique). Consequently, in cases where it is necessary to distinguish between the two in a program, it is necessary to prepare the software status in advance and set the status before setting the PSC register using the store instruction or a bit operation instruction. By checking this status in NMI interrupt servicing, it is possible to distinguish it from an ordinary NMI.

**(b) Release by  $\overline{\text{RESET}}$  pin input**

This is the same as an ordinary reset operation.

### 8.5.5 Software STOP mode

#### (1) Settings and operating state

In this mode, the clock generator (oscillator and PLL synthesizer) is stopped. The system overall is stopped, and it enters an ultra-low power consumption state where only device leakage current is lost.

It is possible to enter the software STOP mode by setting the PSC register (specific register) using a store instruction (ST/SST instruction) or a bit manipulation instruction (SET1/CLR1/NOT1 instruction) in software (refer to **3.4.9 Specific registers**).

In the case of the PLL mode and oscillator connection mode (CESEL bit of the PSC register = 0), it is necessary to secure the oscillation stabilization of the oscillator after releasing the software STOP mode.

In the software STOP mode, program execution stops, but all the contents of all the registers, internal RAM, and ports are held in the state they were in just before entering the software STOP mode. Operation of the internal peripheral I/O (except the ports) is also stopped.

The status of each hardware unit during the software STOP mode is as shown in Table 8-5.

**Caution** In the case of the direct mode (CKSEL pin = 1) or external clock connection mode (CESEL bit of the PSC register = 1), the software STOP mode cannot be used.

**Table 8-5. Operating States When in Software STOP Mode**

Function		Operating State
Clock generator		Stop
Internal system clock		Stop
CPU		Stop
Port <sup>Note</sup>		Hold
Internal peripheral I/O (except ports)		Stop
Internal data <sup>Note</sup>		All the CPU's registers, status, data, internal RAM contents, other internal data, etc. are retained in the state they were in before entering the HALT mode.
When in external expansion mode	D0 to D15	High-impedance
	A0 to A23	
	$\overline{\text{RD}}$ , $\overline{\text{WE}}$ , $\overline{\text{OE}}$ , $\overline{\text{BCYST}}$	
	$\overline{\text{LWR}}$ , $\overline{\text{UWR}}$ , $\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$	High-level output
	$\overline{\text{CS0}}$ , $\overline{\text{CS3}}$ to $\overline{\text{CS5}}$	
	$\overline{\text{RAS3}}$ to $\overline{\text{RAS5}}$	Operating
	$\overline{\text{LCAS}}$ , $\overline{\text{UCAS}}$	
	$\overline{\text{HLDRQ}}$	Input (no sampling)
	$\overline{\text{HLDK}}$	High-impedance
	$\overline{\text{WAIT}}$	Input (no sampling)
CLKOUT		Low-level output

**Note** If the  $V_{DD}$  value is within the operable range.

However, even when it drops below the minimum operable voltage, if the data hold voltage  $V_{DDR}$  is maintained, the contents of internal RAM only are held.



**(2) Releasing software STOP mode**

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

Also, when releasing the software STOP mode in the PLL mode and the oscillator connection mode (CESEL bit of the PSC register = 0), it is necessary to secure oscillation stabilization time for the oscillator.

Note that depending on the program, PLL lockup time may also be necessary. For details, refer to **8.4 PLL Lockup**.

**(a) Release by NMI pin input**

An NMI pin input is acknowledged as an NMI request as well as a release of the software STOP mode.

However, if setting in the software STOP mode is included in an NMI servicing routine, the software STOP mode only is released and the interrupt is not acknowledged. The interrupt request itself is held pending.

The interrupt servicing started when the STOP mode is released by an NMI pin input is treated in the same way as ordinary NMI interrupt servicing in an emergency, etc. (since the NMI interrupt handler address is unique). Consequently, in cases where it is necessary to distinguish between the two, it is necessary to prepare the software status in advance and set the status before setting the PSC register using the store instruction or a bit operation instruction. By checking this status in NMI interrupt servicing, it is possible to distinguish it from an ordinary NMI.

**(b) Release by  $\overline{\text{RESET}}$  pin input**

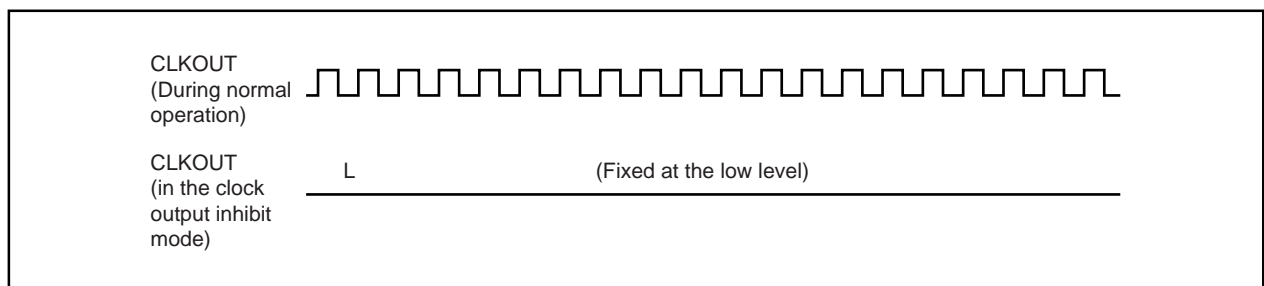
This is the same as an ordinary reset operation.

**8.5.6 Clock output inhibit mode**

If the DCLK0 bit and DCLK1 bit of the PSC register are set to 1, the system enters the clock output inhibit mode, in which clock output from the CLKOUT pin is disabled.

This is most appropriate in systems which access instruction fetches or data from external expansion devices asynchronously.

In this mode, since the CLKOUT signal output's operation is completely stopped, much lower power consumption and suppression of radiation noise from the CLKOUT pin is possible. Also, by combining this mode with the HALT, IDLE, and software STOP mode, more effective power saving becomes possible (refer to **8.5.2 Control registers**).



## 8.6 Securing Oscillation Stabilization Time

### 8.6.1 Specifying securing of oscillation stabilization time

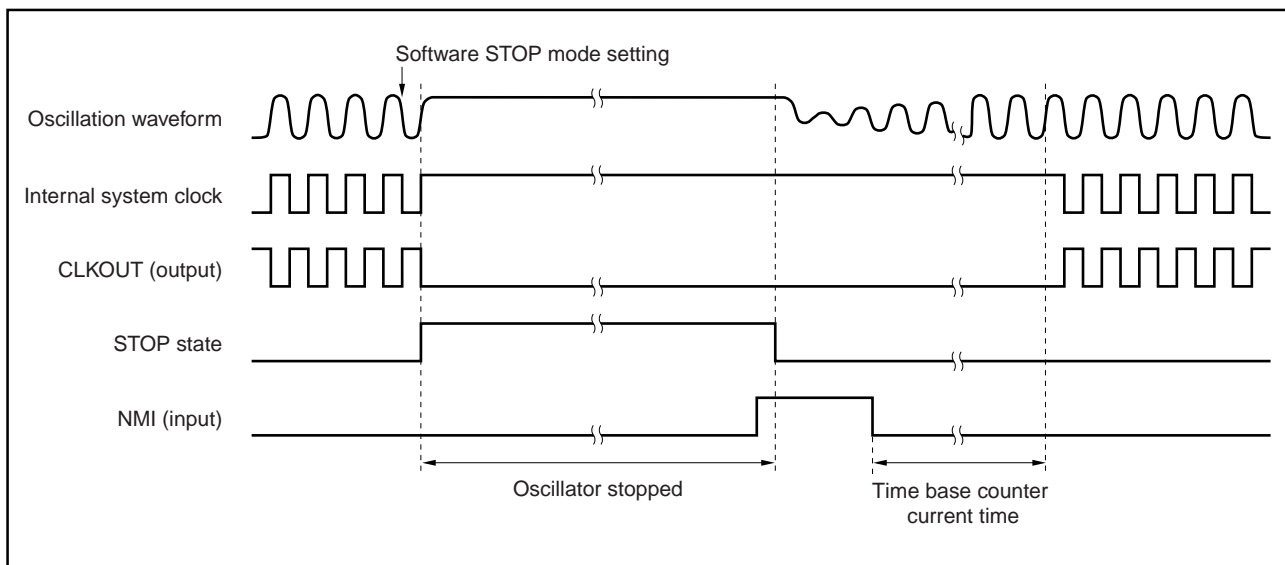
There are 2 methods for specifying securing of time for stabilizing the oscillator in the stop mode after releasing the software STOP mode.

#### (1) If securing time by the internal time base counter (NMI pin input)

If the active edge of the NMI pin is input, the software STOP mode is released. When the inactive edge is input to the pin, the time base counter (TBC) starts counting, and at that count time, the time until the clock output from the oscillator stabilizes is secured.

Oscillation stabilization time  $\equiv$  (Active level width after NMI input active edge detection) + (TBC count time)

After the proper time, start internal system clock output and branch to the NMI interrupt handler address.



The NMI pin should normally be set at the inactive level (for example, so that it changes to high level when the active edge is specified to be falling).

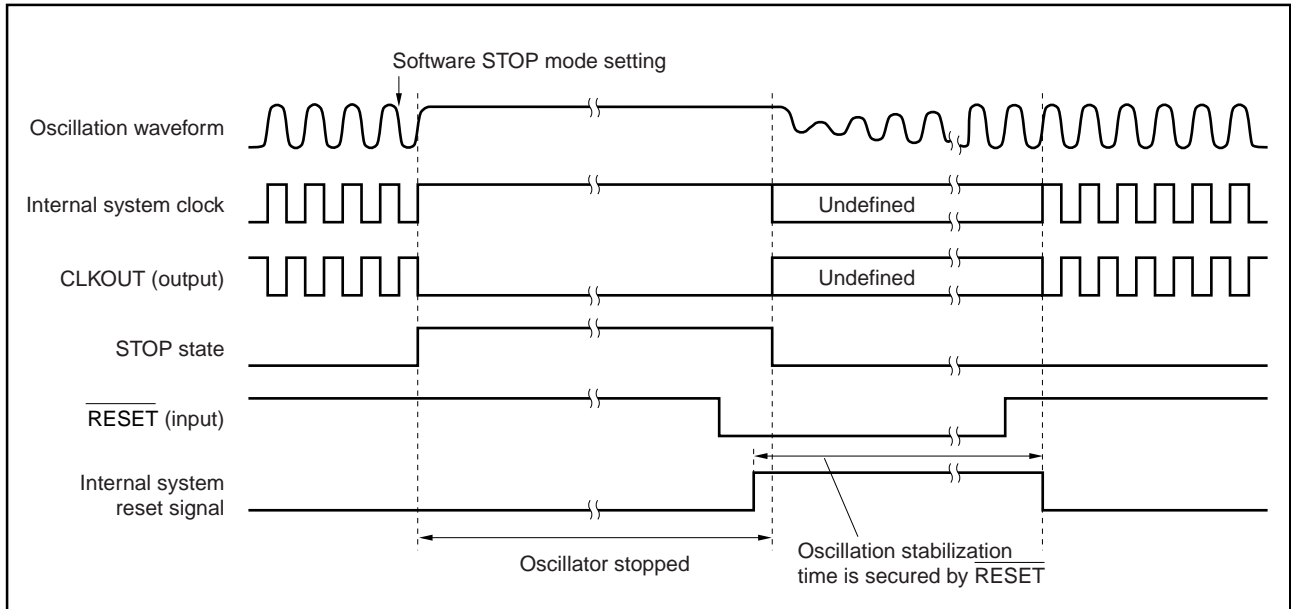
Furthermore, if an operation is executed which sets the system in the STOP mode for a time until an interrupt is received from the CPU from the NMI active edge input timing, the software STOP mode is quickly released. In the case of the PLL mode and the resonator connection mode (CESEL bit of PSC register = 0), program execution starts after the oscillation stabilization time is secured by the time base counter after input of the NMI pin's inactive edge.

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

By inputting the falling edge to the  $\overline{\text{RESET}}$  pin, the software STOP mode is released.

At the signal low level width input to the pin, enough time is secured until the clock output from the oscillator stabilizes.

After inputting the rising edge to the  $\overline{\text{RESET}}$  pin, supply of the internal system clock begins and the system branches to the handler address that was set at system reset time.



### 8.6.2 Time base counter (TBC)

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

- **Resonator connection time (PLL Mode, and CESEL bit of the PSC Register = 0)**

After releasing the software STOP mode, the oscillation stabilization time is counted by the TBC and after counting is ended, program execution begins.

The TBC count clock is selected by the TBCS bit in the PSC register, and it is possible to set the following count times (refer to **8.5.2 (1) Power save control register (PSC)**).

**Table 8-6. Example of Count Time ( $\phi = 5 \times f_{xx}$ )**

TBCS Bit	Count Clock	Count Time		
		$f_{xx} = 3.2768 \text{ MHz}$	$f_{xx} = 5.0000 \text{ MHz}$	$f_{xx} = 6.0000 \text{ MHz}$
		$\phi = 16.384 \text{ MHz}$	$\phi = 25.000 \text{ MHz}$	$\phi = 30.000 \text{ MHz}$
0	$f_{xx}/2^8$	20.0 ms	13.1 ms	10.9 ms
1	$f_{xx}/2^9$	40.0 ms	26.2 ms	21.8 ms

$f_{xx}$ : External resonator frequency

$\phi$ : Internal system clock frequency

## CHAPTER 9 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)

### 9.1 Features

- Measures the pulse interval and frequency and outputs a programmable pulse.
  - 16-bit measurements are possible.
  - Pulse multiple states can be generated (interval pulse, one shot pulse)
- Timer 1
  - 16-bit timer/event counter
  - Count clock sources: 2 types (internal system clock division selection, external pulse input)
  - Capture/compare common registers: 9
  - Capture registers: 7
  - Count clear pins: TCLR10 to TCLR12
  - Interrupt sources: 20 types
  - External pulse outputs: 3
- Timer 4
  - 16-bit interval timer
  - The count clock is selected from the internal system clock divisions.
  - Compare registers: 2
  - Interrupt sources: 2 types

## 9.2 Basic Configuration

The basic configuration is shown below.

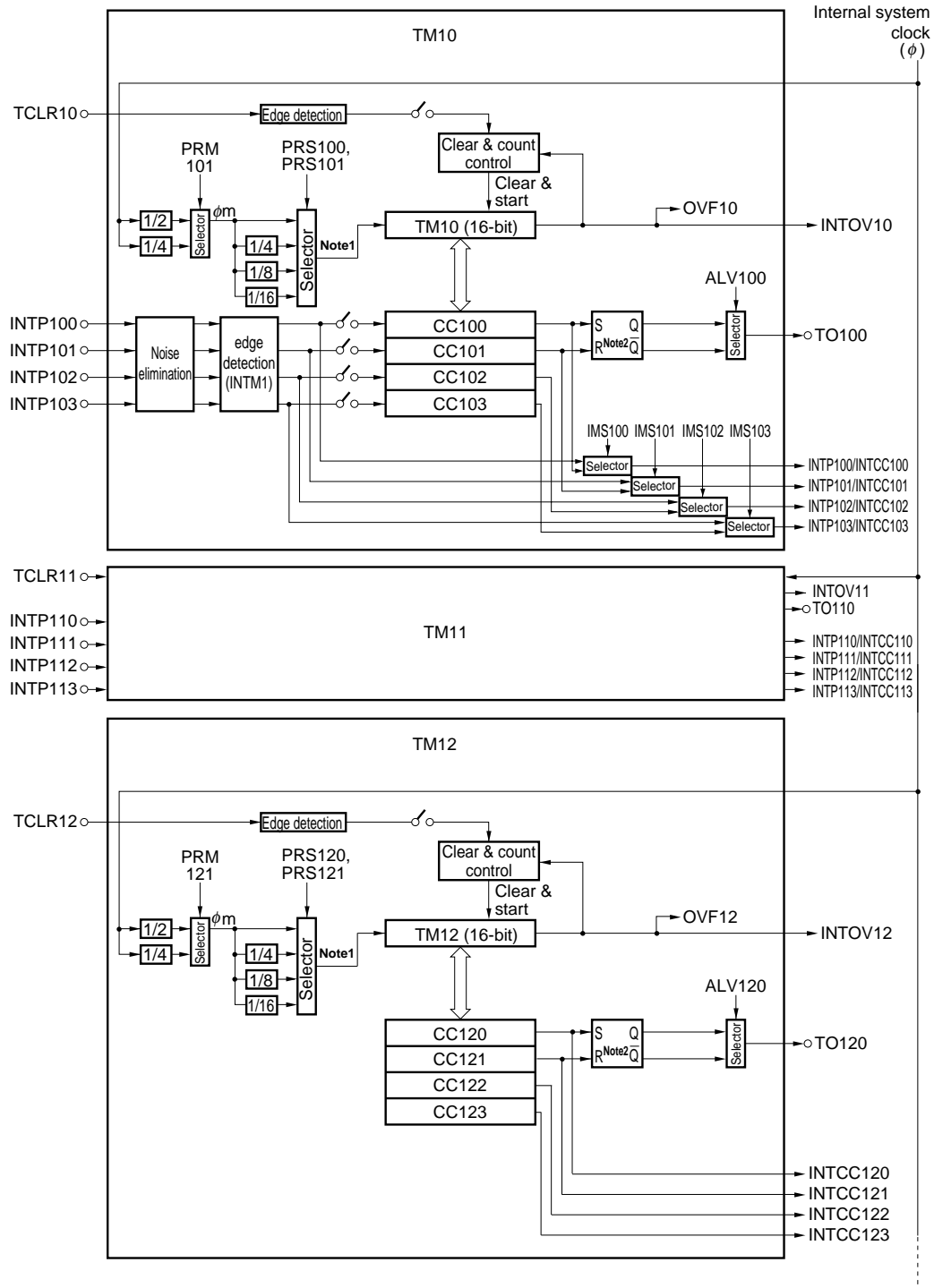
**Table 9-1. RPU Configuration List**

Timer	Count Clock	Register	Read/Write	Interrupt Signals Generated	Capture Trigger	Timer Output S/R	Other Functions
Timer 1	$\phi/2$ $\phi/4$ $\phi/8$ $\phi/16$ $\phi/32$ $\phi/64$ TI13 Pin Input	TM10	Read	INTOV10	—	—	External clear
		CC100	Read/write	INTCC100	INTP100	TO100 (S)	—
		CC101	Read/write	INTCC101	INTP101	TO100 (R)	—
		CC102	Read/write	INTCC102	INTP102	—	—
		CC103	Read/write	INTCC103	INTP103	—	—
		TM11	Read	INTOV11	—	—	External clear
		CC110	Read/write	INTCC110	INTP110	TO110 (S)	A/D conversion start trigger
		CC111	Read/write	INTCC111	INTP111	TO110 (R)	A/D conversion start trigger
		CC112	Read/write	INTCC112	INTP112	—	A/D conversion start trigger
		CC113	Read/write	INTCC113	INTP113	—	A/D conversion start trigger
		TM12	Read	INTOV12	—	—	External clear
		CC120	Read/write	INTCC120	—	TO120 (S)	—
		CC121	Read/write	INTCC121	—	TO120 (R)	—
		CC122	Read/write	INTCC122	—	—	—
		CC123	Read/write	INTCC123	—	—	—
		TM13	Read	INTOV13	—	—	—
		CC130	Read/write	INTCC130	INTP130	—	—
		CC131	Read/write	INTCC131	—	—	—
		CC132	Read/write	INTCC132	—	—	—
		CC133	Read/write	INTCC133	—	—	—
Timer 4	$\phi/32$ $\phi/64$ $\phi/128$ $\phi/256$	TM40	Read	—	—	—	—
		CM40	Read/write	INTCM40	—	—	—
		TM41	Read	—	—	—	—
		CM41	Read/write	INTCM41	—	—	—

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

(1) Timer 1 (16-bit timer/event counter)

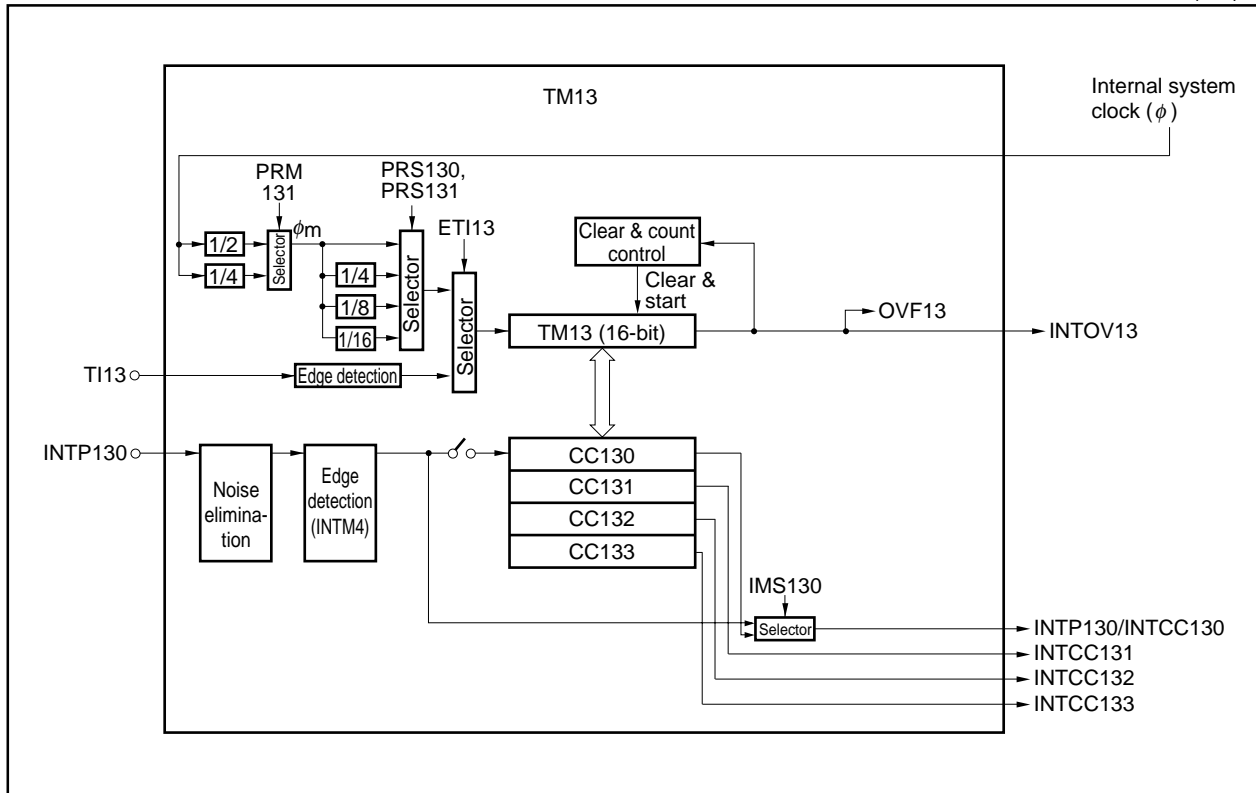
(1/2)



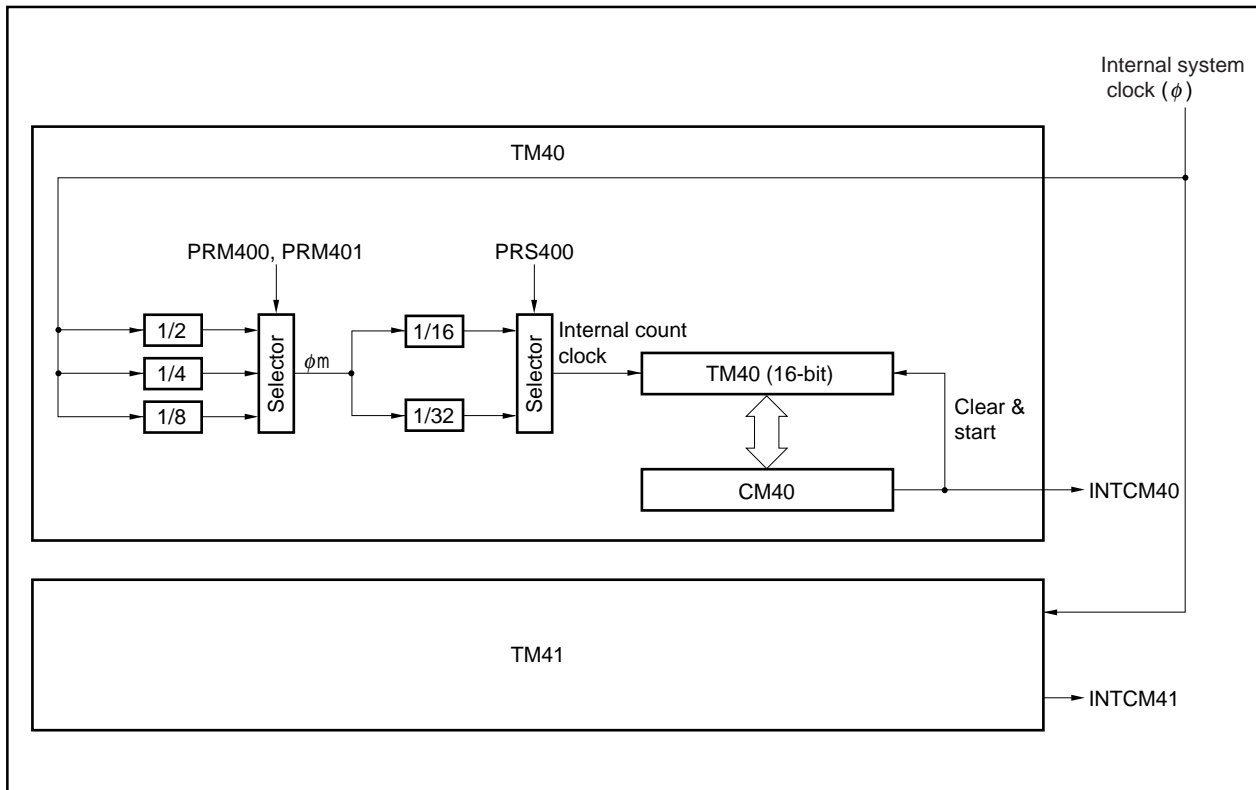
- Notes**
1. Internal count clock
  2. Reset priority

(1) Timer 1 (16-bit timer/event counter)

(2/2)



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









### 9.2.1 Timer 1

#### (1) Timers 10 to 13 (TM10 to TM13)

TM1n functions as a 16-bit free running timer or as an event counter for an external signal. Mainly, besides period measurement and frequency measurement, it can be used as a pulse output ( $n = 0$  to 3). TM1n is read-only, in 16-bit units.

	15															0			
TM10																	Address FFFFFF250H	After reset 0000H	
TM11																	FFFFFF270H	0000H	
TM12																	FFFFFF290H	0000H	
TM13																	FFFFFF2B0H	0000H	

TM1n carries out count-up operations of the internal count clock or of an external count clock. Starting and stopping of the timer is controlled by the CE1n bit of timer control register 1n (TMC1n). Selection of internal or external count clocks is performed by the TMC1n register.

#### (a) Selection of an external count clock

TM13 operates as an event counter. The active edge is specified by the timer unit mode register 13 (TUM13) and through input of pin TI13, TM13 is counted up.

#### (b) Selection of an internal count clock

TM1n operates as a free running timer. The counter clock can be selected from among the divisions performed by the prescaler,  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ , or  $\phi/64$ , through the TMC1n register.

If the timer overflows, an overflow interrupt can be generated. Also, the timer can be stopped after an overflow through the TUM1n register specification.

The timer can also be cleared and started using the external input TCLR1n. When this is done, the prescaler is cleared at the same time, so the time from TCLR1n input to timer count-up is constant corresponding to the prescaler's dividing ratio. The operation setting is carried out by the TUM1n register.

**Caution** The count clock cannot be changed during timer operation.

**(2) Capture/compare registers 1n0 to 1n3 (CC1n0 to CC1n3) (n = 0 to 3)**

The capture/compare registers are 16-bit registers to which TM1n is connected. They can be used as either a capture register or a compare register in accordance with the specification in timer unit mode register 1n (TUM1n). These registers can be read/written in 16-bit units.

CC100 to CC103	<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;">15</div> <div style="border: 1px solid black; width: 150px; height: 20px; position: relative;"> <div style="position: absolute; right: 0; top: -5px;">0</div> </div> </div>	Address FFFFF252H to FFFFF258H	After reset Undefined
CC110 to CC113	<div style="border: 1px solid black; width: 150px; height: 20px;"></div>	FFFFF272H to FFFFF278H	Undefined
CC120 to CC123	<div style="border: 1px solid black; width: 150px; height: 20px;"></div>	FFFFF292H to FFFFF298H	Undefined
CC130 to CC133	<div style="border: 1px solid black; width: 150px; height: 20px;"></div>	FFFFF2B2H to FFFFF2B8H	Undefined

**(a) Set as a capture register**

If set as a capture register, the active edge of the corresponding signals in external interrupts INTP100 to INTP103, INTP110 to INTP113, and INTP130 is detected as a capture trigger. Timer 1n is synchronized with the capture trigger and latches a count value (capture operation). The capture operation is performed out of synch with the count clock. The latched value is held in the capture register until the next capture operation is performed.

If the capture (latch) timing to the capture register and writing to the register in response to an instruction are in contention, the latter has the priority and the capture operation is disregarded.

Also, specification of the active edge of external interrupts (rising, falling, or both edges) can be selected by the external interrupt mode register (INTM1, INTM2, INTM4).

When there is a specification in the capture register, an interrupt is issued when the active edge of INTP100 to INTP103, INTP110 to INTP113, and INTP130 signals is detected. When this is done, an interrupt cannot be issued by INTCC100 to INTCC103, INTCC110 to INTCC113, and INTCC130, which are the compare register's matching signals.

**(b) Set as a compare register**

If set as a compare register, these registers perform a comparison of the timer and register values at each count clock of the timer, and issue an interrupt if the values match.

The compare registers are provided with a set/reset output function. In synch with matching signal generation, the corresponding timer output (TO1n0: n = 0 to 2) is set or reset.

The interrupt source differs with the function of the register.

If specified a compare register, these registers can be made interrupt signals by selecting, through the specification of the TUM1n (n = 0, 1, 3) register, active edge detection of either the INTCC100 to INTCC103, INTCC110 to INTCC113, and INTCC130 signals, which are the matching signals, or the INTP100 to INTP103, INTP110 to INTP113, and INTP130 signals.

Furthermore, if the INTP100 to INTP103, INTP110 to INTP113, and INTP130 signals are selected, acknowledgement of an external interrupt request and timer output by the compare register's set/reset output function can be carried out in parallel.

## 9.2.2 Timer 4

## (1) Timers 40, 41 (TM40, TM41)

TM4n is a 16-bit timer. It can mainly be used as an interval timer for software (n = 0, 1).

TM4n is read-only in 16-bit units.

TM40	<div> <div>15</div> <div>0</div> <div> <div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div> </div> </div>	Address FFFFF350H	After reset 0000H
TM41	<div> <div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div> </div>	FFFFF354H	0000H

Starting and stopping of TM4n is controlled by the CE4n bit of timer control register 4n (TMC4n).

The count clock can be selected from  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ , or  $\phi/256$  divisions of the prescaler via register TMC4n.

**Caution** Since the timer is cleared at the next count clock after a compare match is issued, when the division ratio is large, even if the timer's value is read immediately after the match interrupt is issued, the timer's value may not be 0.

Also, the count clock cannot be changed during timer operation.

## (2) Compare registers 40, 41 (CM40, CM41)

CM4n is a 16-bit register and is connected to TM4n. This register can be read/written in 16-bit units.

CM40	<div> <div>15</div> <div>0</div> <div> <div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div> </div> </div>	Address FFFFF352H	After reset Undefined
CM41	<div> <div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div> </div>	FFFFF356H	Undefined

This register compares TM4n and CM4n each TM4n count clock and if they match, issues an interrupt (INTCM4n). TM4n is cleared in synchronization with this match.

### 9.3 Control Registers

#### (1) Timer unit mode registers 10 to 13 (TUM10 to TUM13)

The TUM1n register is a register which controls the operation of timer 1 and specifies the capture/compare register operation mode (n = 0 to 3).

These registers can be read/written in 16-bit units.

(1/2)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
TUM10	0	0	OST0	ECLR <sub>10</sub>	0 <sup>Note</sup>	0 <sup>Note</sup>	CES <sub>101</sub>	CES <sub>100</sub>	CMS <sub>103</sub>	CMS <sub>102</sub>	CMS <sub>101</sub>	CMS <sub>100</sub>	IMS <sub>103</sub>	IMS <sub>102</sub>	IMS <sub>101</sub>	IMS <sub>100</sub>	Address FFFFFF240H	After reset 0000H
TUM11	0	0	OST1	ECLR <sub>11</sub>	0 <sup>Note</sup>	0 <sup>Note</sup>	CES <sub>111</sub>	CES <sub>110</sub>	CMS <sub>113</sub>	CMS <sub>112</sub>	CMS <sub>111</sub>	CMS <sub>110</sub>	IMS <sub>113</sub>	IMS <sub>112</sub>	IMS <sub>111</sub>	IMS <sub>110</sub>	FFFFFF260H	0000H
TUM12	0	0	OST2	ECLR <sub>12</sub>	0 <sup>Note</sup>	0 <sup>Note</sup>	CES <sub>121</sub>	CES <sub>120</sub>	CMS <sub>123</sub>	CMS <sub>122</sub>	CMS <sub>121</sub>	CMS <sub>120</sub>	IMS <sub>123</sub>	IMS <sub>122</sub>	IMS <sub>121</sub>	IMS <sub>120</sub>	FFFFFF280H	0000H
TUM13	0	0	OST3	0 <sup>Note</sup>	TES <sub>131</sub>	TES <sub>130</sub>	0 <sup>Note</sup>	0 <sup>Note</sup>	CMS <sub>133</sub>	CMS <sub>132</sub>	CMS <sub>131</sub>	CMS <sub>130</sub>	IMS <sub>133</sub>	IMS <sub>132</sub>	IMS <sub>131</sub>	IMS <sub>130</sub>	FFFFF2A0H	0000H

Bit Position	Bit Name	Function
13	OSTn	<p>Overflow Stop</p> <p>Specifies the timer's operation after overflow. This flag is valid only in TM1n.</p> <p>0: Timer continues to count up after timer overflow.</p> <p>1: Timer holds 0000H and is in the stopped state after timer overflow.</p> <p>When this happens, the CE1 bit in the TMC1n register remains at 1.</p> <p>Counting up resumes with the next operation.</p> <p>When ECLR1m = 0: 1 write operation to the CE1n bit.</p> <p>When ECLR1m = 1: Trigger input to the timer clear pin (TCLR1m).</p>
12	ECLR1m	<p>External Input Timer Clear</p> <p>Clearing of the timer is enabled by the TM1m external clear input (TCLR1m).</p> <p>0: Timer is not cleared by an external input.</p> <p>1: TM1m is cleared by an external input.</p> <p>Counting up starts after clearing.</p>

**Note** Be sure to set these bits to 0.

**Remark** n = 0 to 3  
m = 0 to 2

Bit Position	Bit Name	Function															
11, 10	TES131, TES130	<p>TI13 Edge Select Specifies the active edge of the external clock input (TI13).</p> <table border="1"> <thead> <tr> <th>TES131</th><th>TES130</th><th>Active Edge</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>Falling edge</td></tr> <tr> <td>0</td><td>1</td><td>Rising edge</td></tr> <tr> <td>1</td><td>0</td><td>RFU (reserved)</td></tr> <tr> <td>1</td><td>1</td><td>Both the rising and falling edges</td></tr> </tbody> </table>	TES131	TES130	Active Edge	0	0	Falling edge	0	1	Rising edge	1	0	RFU (reserved)	1	1	Both the rising and falling edges
TES131	TES130	Active Edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	RFU (reserved)															
1	1	Both the rising and falling edges															
9, 8	CES1m1, CES1m0	<p>TCLR1m Edge Select Specifies the active edge of the external clear input (TCLR1m).</p> <table border="1"> <thead> <tr> <th>CES1m1</th><th>CES1m0</th><th>Active Edge</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>Falling edge</td></tr> <tr> <td>0</td><td>1</td><td>Rising edge</td></tr> <tr> <td>1</td><td>0</td><td>RFU (reserved)</td></tr> <tr> <td>1</td><td>1</td><td>Both the rising and falling edges</td></tr> </tbody> </table>	CES1m1	CES1m0	Active Edge	0	0	Falling edge	0	1	Rising edge	1	0	RFU (reserved)	1	1	Both the rising and falling edges
CES1m1	CES1m0	Active Edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	RFU (reserved)															
1	1	Both the rising and falling edges															
7 to 4	CMS1nm	<p>Capture/Compare Mode Select Selects the capture/compare register's (CC1n) operation mode.</p> <p>0: Operates as a capture register. However, the capture operation when it is specified as a capture register is performed only when the CE1x bit of the TMC1x register = 1 (x = 0 to 3).</p> <p>1: Operates as a compare register.</p> <p><b>Caution</b> CC120 to CC123 and CC131 to CC133 cannot be used as capture registers. To use these registers, be sure to set bit 1 of CMS1n to 1.</p>															
3 to 0	IMS1n <sup>Note</sup>	<p>Interrupt Mode Select Selects either INTP1n or INTCC1n as the interrupt source.</p> <p>0: Makes the compare register's matching signal INTCC1n the interrupt request signal.</p> <p>1: It makes the external input signal INTP1n the interrupt request signal.</p>															

**Note** No external signal is input to CC120 to CC123 and CC131 to CC133. To generate an interrupt request signal, therefore, set IMS1n to 0.

**Remark** n = 00 to 03, 10 to 13, 20 to 23, 30 to 33  
m = 0 to 2

**Remark** If the A/D converter is set in the timer trigger mode, the compare register's match interrupt becomes the A/D conversion start trigger, starting the conversion operation. When this happens, the compare register's match interrupt functions as a compare register match interrupt to the CPU. In order for a compare register match interrupt not to be issued to the CPU, disable interrupts with the interrupt mask bits (P11MK0 to P11MK3) of the interrupt control register (P11IC0 to P11IC3).

**(2) Timer control registers 10 to 13 (TMC10 to TMC13)**

TMC10 to 13 control the respective operations of TM10 to TM13.

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

(1/2)

	7	6	5	4	3	2	1	0		
TMC10	CE10	0	0	0 <sup>Note</sup>	PRS101	PRS100	PRM101	0	Address FFFFFF242H	After reset 00H
TMC11	CE11	0	0	0 <sup>Note</sup>	PRS111	PRS110	PRM111	0	FFFFFF262H	00H
TMC12	CE12	0	0	0 <sup>Note</sup>	PRS121	PRS120	PRM121	0	FFFFFF282H	00H
TMC13	CE13	0	0	ETI13	PRS131	PRS130	PRM131	0	FFFFFF2A2H	00H

Bit Position	Bit Name	Function
7	CE1n	Count Enable Controls timer operation. 0: The timer is stopped in the 0000H state and does not operate. 1: The timer performs a count operation. However, when the ECLR1n bit of the TUM1n register is 1, the timer does not start counting up until there is a TCLR1n input. When the ECLR1n bit is 0, the operation of setting (1) in the CE1n bit becomes the count start trigger. Thus, after the CE1n bit is set (1) when the ECLR1n bit = 1, the timer will not start even if the ECLR1n bit is made 0.
4	ETI13	External TI13 Input Specifies whether switching of the count clock is external or internal. 0: Specifies the $\phi$ system (internal). 1: Specifies TI13 (external).

**Note** Be sure to set these bits to 0.

**Caution** Do not change the count clock during timer operation.

**Remark** n = 0 to 3

Bit Position	Bit Name	Function
3, 2	PRS1n1, PRS1n0	Prescaler Clock Select Selects the internal count clock ( $\phi m$ is the intermediate clock).
1	PRM1n1	Prescaler Clock Mode Selects the intermediate count clock ( $\phi m$ ). ( $\phi$ is the internal system clock). 0: $\phi/2$ 1: $\phi/4$

**Caution** Do not change the count clock during timer operation.

**Remark** n = 0 to 3

**(3) Timer control registers 40, 41 (TMC40, TMC41)**

TMC40 and TMC41 control the operation of TM40 and TM41, respectively.

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

	7	6	5	4	3	2	1	0		
TMC40	CE40	0	0	0	0	PRS400	PRM401	PRM400	Address FFFFFF342H	After reset 00H
TMC41	CE41	0	0	0	0	PRS410	PRM411	PRM410	FFFFFF346H	00H

Bit Position	Bit Name	Function															
7	CE4n	Count Enable Controls timer operations. 0: The timer is stopped in the 0000H state and does not operate. 1: The timer performs a count operation.															
2	PRS4n0	Prescaler Clock Select Selects the internal count clock ( $\phi_m$ is the intermediate clock). 0: $\phi_m/16$ 1: $\phi_m/32$															
1, 0	PRM4n1, PRM4n0	Prescaler Clock Mode Selects the intermediate count clock ( $\phi_m$ ). ( $\phi$ is the internal system clock). <table border="1"> <tr> <th>PRM4n1</th><th>PRM4n0</th><th><math>\phi_m</math></th></tr> <tr> <td>0</td><td>0</td><td><math>\phi/2</math></td></tr> <tr> <td>0</td><td>1</td><td><math>\phi/4</math></td></tr> <tr> <td>1</td><td>0</td><td><math>\phi/8</math></td></tr> <tr> <td>1</td><td>1</td><td>RFU (reserved)</td></tr> </table>	PRM4n1	PRM4n0	$\phi_m$	0	0	$\phi/2$	0	1	$\phi/4$	1	0	$\phi/8$	1	1	RFU (reserved)
PRM4n1	PRM4n0	$\phi_m$															
0	0	$\phi/2$															
0	1	$\phi/4$															
1	0	$\phi/8$															
1	1	RFU (reserved)															

**Caution** Do not change the count clock during timer operation.

**Remark** n = 0, 1



**(4) Timer output control registers 10 to 12 (TOC10 to TOC12)**

The TOC1n register controls the timer output from the TO1n0 pin (n = 0 to 2).

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

	7	6	5	4	3	2	1	0		
TOC10	0 <sup>Note</sup>	0 <sup>Note</sup>	ENTO100	ALV100	0	0	0	0	Address FFFFFF244H	After reset 00H
TOC11	0 <sup>Note</sup>	0 <sup>Note</sup>	ENTO110	ALV110	0	0	0	0	FFFFFF264H	00H
TOC12	0 <sup>Note</sup>	0 <sup>Note</sup>	ENTO120	ALV120	0	0	0	0	FFFFFF284H	00H

Bit Position	Bit Name	Function
5	ENTO1n0	<p>Enable TO pin Enables output of each corresponding timer (TO1n0).</p> <p>0: Timer output is disabled. The reverse phase level (inactive level) of the ALV1n0 bit is output from the TO1n0 pin. Even if a match signal is generated by the corresponding compare register, the level of the TO1n0 pin does not change.</p> <p>1: Timer output is enabled. If a match signal is generated from the corresponding compare register, the timer's output changes. From the timer that timer output is enabled until match signals are first generated, the reverse phase level (inactive level) of the ALV1n0 bit is output.</p>
4	ALV1n0	<p>Active Level TO pin Specifies the timer output's active level.</p> <p>0: The active level is the low level.</p> <p>1: The active level is the high level.</p>

**Note** Be sure to set these bits to 0.

**Remarks** 1. The TO1n0 output flip-flop is reset priority.  
2. n = 0 to 2

**Caution** The TO1m0 output is not changed by an external interrupt signal (INTP1m0 to INTP1m3, INTP130). When the TO1m0 signal is used, specify the capture/compare register as the compare register (CMS1m0 to CMS1m3 bit of the TUM1m register = 1) (m = 0, 1).

**(5) External interrupt mode registers 1, 2, 4 (INTM1, INTM2, INTM4)**

If CC1n0 to CC1n3, CC130 (n = 0, 1) of TM1n are used as a capture register, the active edge of the external interrupt INTP1n0 to INTP1n3 signals is detected as a capture trigger (for details, refer to **CHAPTER 7 INTERRUPT/EXCEPTION PROCESSING FUNCTION**).

**(6) Timer overflow status register (TOVS)**

This interrupts overflow flags from TM10 to TM13, TM40, and TM41.

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

By setting and resetting the TOVS register through software, polling of overflow occurrences can be accomplished.

	7	6	5	4	3	2	1	0		
TOVS	OVF41	OVF40	0	0	OVF13	OVF12	OVF11	OVF10	Address FFFFFF230H	After reset 00H

Bit Position	Bit Name	Function
7, 6, 3 to 0	OVF41, OVF40, OVF13 to OVF10	<p>Overflow Flag This is the overflow flag for TM41, TM40 and TM1n. 0: No overflow is generated. 1: Overflow is generated.</p> <p><b>Caution</b> Interrupt requests (INTOV1n) for the interrupt controller are generated in synch with an overflow from TM1n, but because interrupt operations and the TOVS register are independent, the overflow flag (OVF1n) from TM1n can be operated by software just like other overflow flags. At this time, the interrupt request flag (OVF1n) corresponding to INTOV1n is not affected.</p> <p>During CPU access interval, transfers to the TOVS register cannot be made. Therefore, even if an overflow is generated during a readout from the TOVS register, the flag's value does not change and it is reflected in the next read operation.</p>

**Remark** n = 0 to 3

## 9.4 Timer 1 Operation

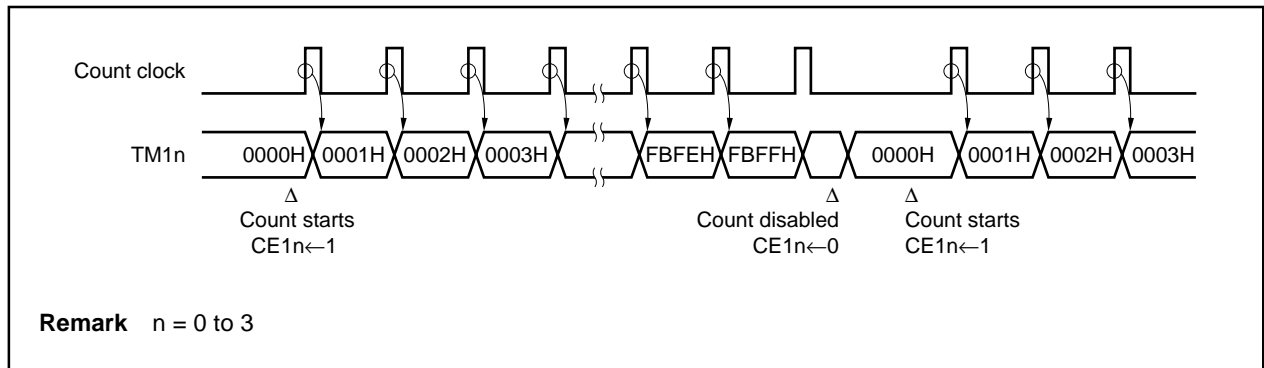
### 9.4.1 Count operation

Timer 1 functions as a 16-bit free-running timer or an event counter for an external signal.

Whether the timer operates as a free-running timer or event counter is specified by timer control register 1n (TMC1n) (n = 0 to 3).

When it is used as a free-running timer, and when the count values of TM1n match with the value of any of the CC1n0 to CC1n3 registers, an interrupt signal is generated, and timer output signal TO1m0 (m = 0 to 2) can be set/reset. In addition, a capture operation that holds the current count value of TM1n and loads it into one of the four registers CC1n0 to CC1n3, is performed in synchronization with the valid edge detected from the corresponding external interrupt request pin as an external trigger. The captured value is retained until the next capture trigger is generated.

Figure 9-1. Basic Operation of Timer 1



### 9.4.2 Count clock selection

The count clock input to Timer 13 is either internal or external, and can be selected by the ETI13 bit in the TMC13 register. The count clock input to timers 10, 11, and 12 is internal only.

**Caution** Do not change the count clock during timer operation.

#### (1) Internal count clock (ETI1n bit = 0)

An internal count clock can be selected from among 6 possible clock rates,  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ , or  $\phi/64$ , by the setting of the PRS1n1, PRS1n0, and PRM1n1 bits of the TMC1n register.

PRS1n1	PRS1n0	PRM1n1	Internal Count Clock
0	0	0	$\phi/2$
0	0	1	$\phi/4$
0	1	0	$\phi/8$
0	1	1	$\phi/16$
1	0	0	$\phi/16$
1	0	1	$\phi/32$
1	1	0	$\phi/32$
1	1	1	$\phi/64$

**Remark** n = 0 to 3

#### (2) External count clock (ETI13 bit = 1)

This counts the signals input to the TI13 pin. At this time, Timer 1 can be operated as an event counter. The TI13 active edge can be set by the TES131 and TES130 bits of the TUM13 register.

TES131	TES130	Active Edge
0	0	Rising edge
0	1	Falling edge
1	0	RFU (reserved)
1	1	Both the rising and falling edges

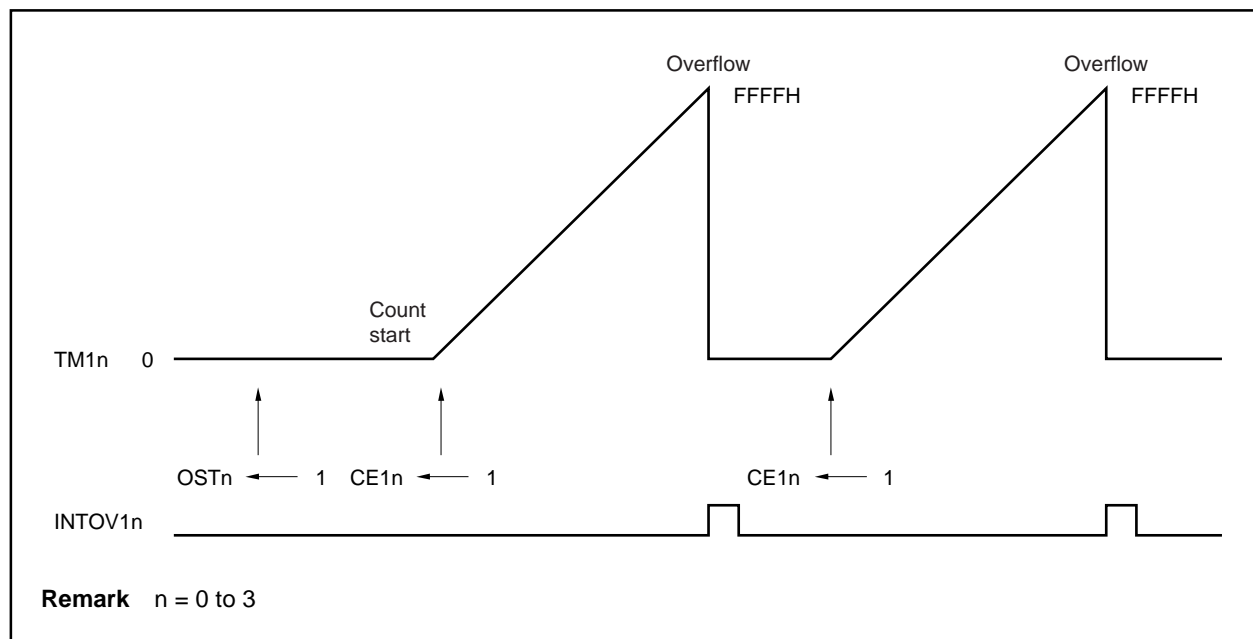
### 9.4.3 Overflow

When the TM1n register counts the count clock to FFFFH and overflow occurs as a result, a flag is set in the OVF1n bit of the TOVS register and an overflow interrupt (INTOV1n) is generated (n = 0 to 3).

Also, by setting the OSTn bit (1) in the TUM1n register, the timer can be stopped after overflow. If the timer is stopped due to an overflow, the count operation does not resume until the CE1n bit in the TMC1n register is set (1).

Note that even if the CE1n bit is set (1) during a count operation, it has no influence on operation.

**Figure 9-2. Operation After Overflow (If ECLR1n = 0 and OSTn = 1)**



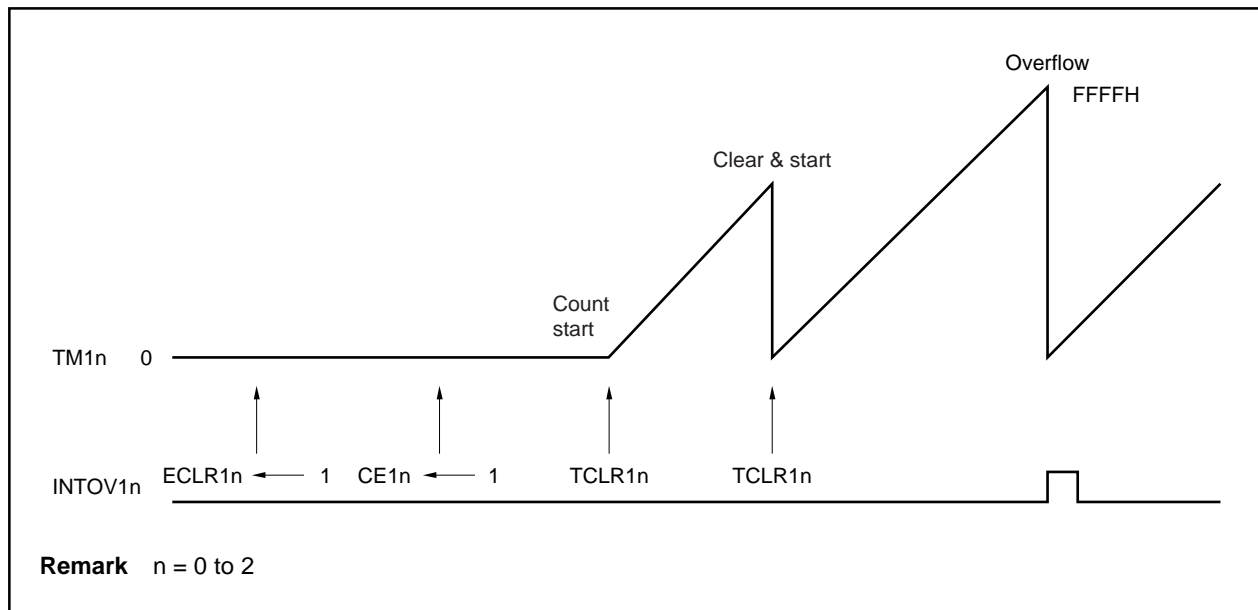
#### 9.4.4 Clearing/starting timer by TCLR1n signal input

Timer 1 ordinarily starts a counting operation when the CE1n bit in the TMC1n register is set (1), but TM1n can be cleared and a count operation started by input of the TCLR1n signal ( $n = 0$  to 2).

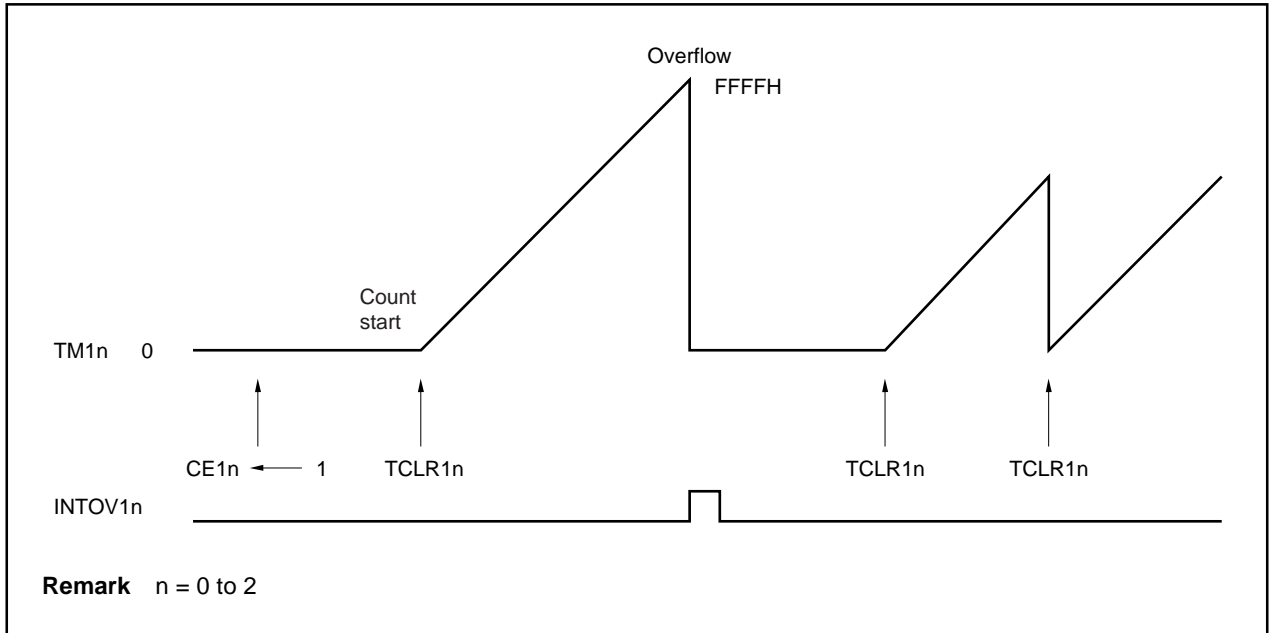
If the ECLR1n bit of the TUM1n register is set to 1, and the OSTn bit is set to 0, if the active edge is input to the TCLR1n signal after the CE1n bit is set (1), the counting operation starts. Also, if the active edge is input to the TCLR1n signal during operation, the TM1n's value is cleared and the count operation resumes (refer to **Figure 9-3**).

If the ECLR1n bit of the TUM1n register is set to 1, and the OSTn bit is set to 1, the counting operation starts if the active edge is input to the TCLR1n signal after the CE1n bit is set (1). If TM1n overflows, the count operation stops once and it does not resume the count operation until the active edge is input again to the TCLR1n signal. If the active edge of the TCLR1n signal is detected during a counting operation, TM1n is cleared and the count operation continues (refer to **Figure 9-4**). Note that if the CE1n bit is set (1) after an overflow, the count operation does not resume.

**Figure 9-3. Timer Clear/Start Operation by TCLR1n Signal Input (If ECLR1n = 1 and OSTn = 0)**



**Figure 9-4. Relationship Between Clear/Start by TCLR1n Signal Input and Overflow Operation**  
(If ECLR1n = 1 and OSTn = 1)



#### 9.4.5 Capture operation

In synch with an external trigger, a capture operation is performed in which the TM1n count value is captured and held in the capture register asynchronous to the count clock (n = 0, 1, 3). The active edge detected from external interrupt request input pins INTP1m0 to INTP1m3 (m = 0, 1) and INTP130 is used as the external trigger (capture trigger). In synch with that capture trigger signal, the count value of TM1n, as it is counting, is captured and held in the capture register. The value in the capture register is held until the next capture trigger is generated.

Also, interrupt requests (INTCC1m0 to INTCC1m3, INTCC130) are generated from the INTP1m0 to INTP1m3 and INTP130 signal inputs.

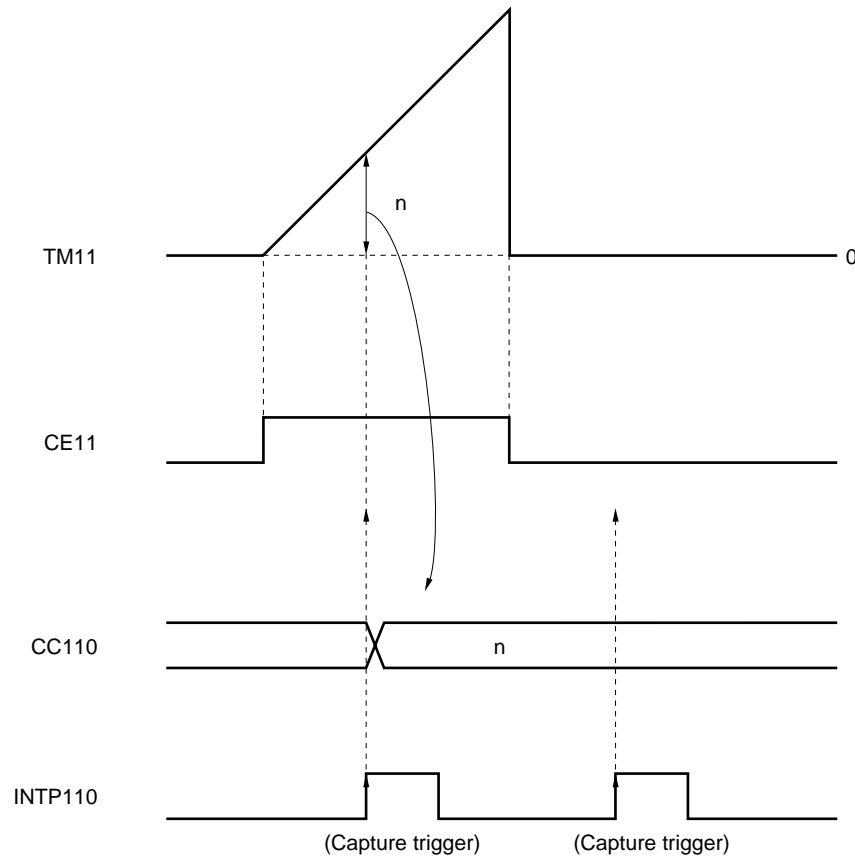
**Table 9-2. Capture Trigger Signals (TM1n) to 16-Bit Capture Registers**

Capture Register	Capture Trigger Signal
CC1m0	INTP1m0
CC1m1	INTP1m1
CC1m2	INTP1m2
CC1m3	INTP1m3
CC130	INTP130

- Remarks**
1. CC1m0 to CC1m3 and CC130 are the capture/compare registers. Which register is used is specified in timer unit mode register 1n (TUM1n).
  2. n = 0, 1, 3  
m = 0, 1

The capture trigger's active edge is set by the external interrupt mode register (INTM1, INTM2, INTM4). If both the rising and falling edges are made capture triggers, the input pulse width from an external source can be measured. Also, if the edge from one side is used as the capture trigger, the input pulse's period can be measured.

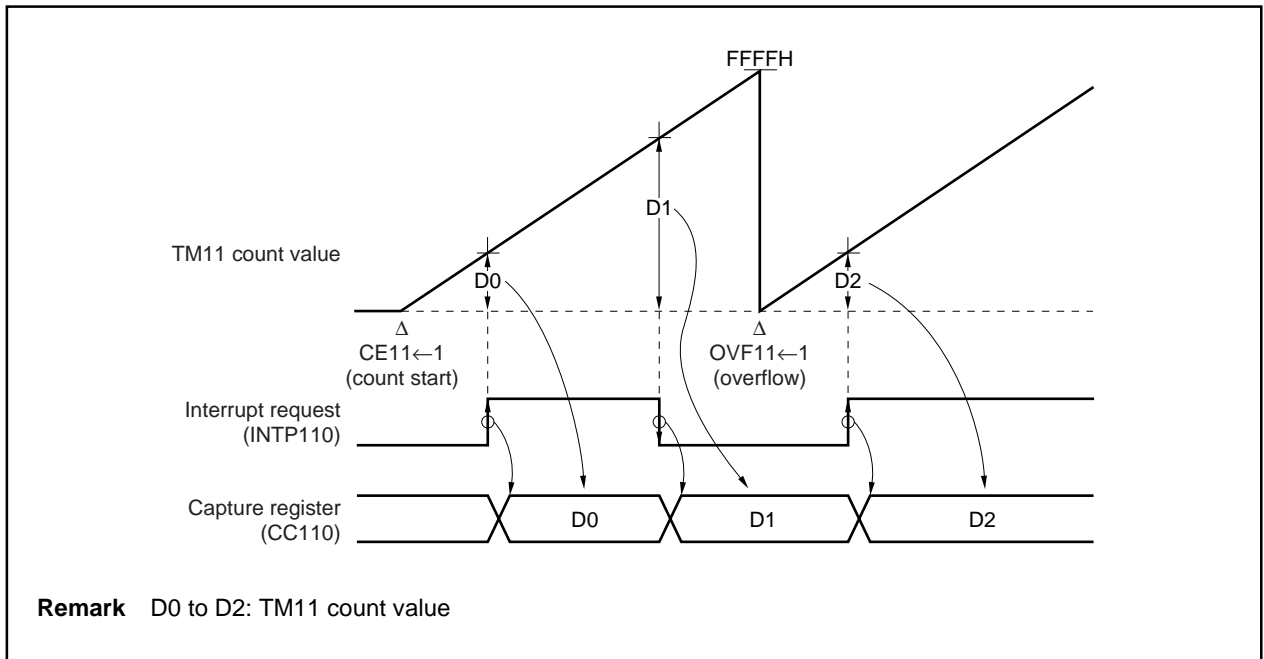
**Figure 9-5. Example of Capture Operation**



**Remark** When the CE11 bit = 0, no capture operation is performed even if INTP110 is input.



Figure 9-6. Example of TM11 Capture Operation (When Both Edges Are Specified)



### 9.4.6 Compare operation

Compare operations in which the value set in the compare register is compared with the TM1n count value are performed ( $n = 0$  to 3).

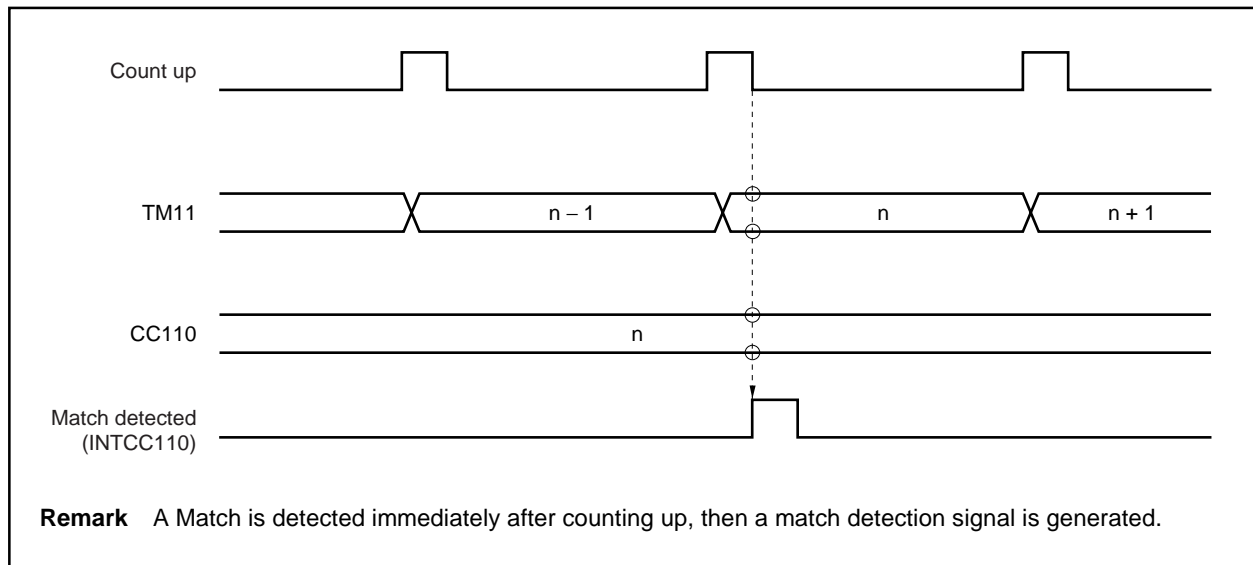
If the TM1n count value matches the value that has been previously set in the compare register, a match signal is sent to the output control circuit (refer to **Figure 9-7**). The timer output pins (TO100, TO110, TO120) are changed by the match signal and simultaneously issue interrupt request signals.

**Table 9-3. Interrupt Request Signals (TM1n) from 16-Bit Compare Registers**

Compare Register	Interrupt Request Signal
CC1n0	INTCC1n0
CC1n1	INTCC1n1
CC1n2	INTCC1n2
CC1n3	INTCC1n3

- Remarks**
1. CC1x0 to CC1x3 and CC130 ( $x = 0, 1$ ) are capture/compare registers. Which register will be used is specified by the timer unit mode register 1m (TUM1m).
  2.  $n = 0$  to 3  
 $m = 0, 1, 3$

**Figure 9-7. Example of Compare Operation**

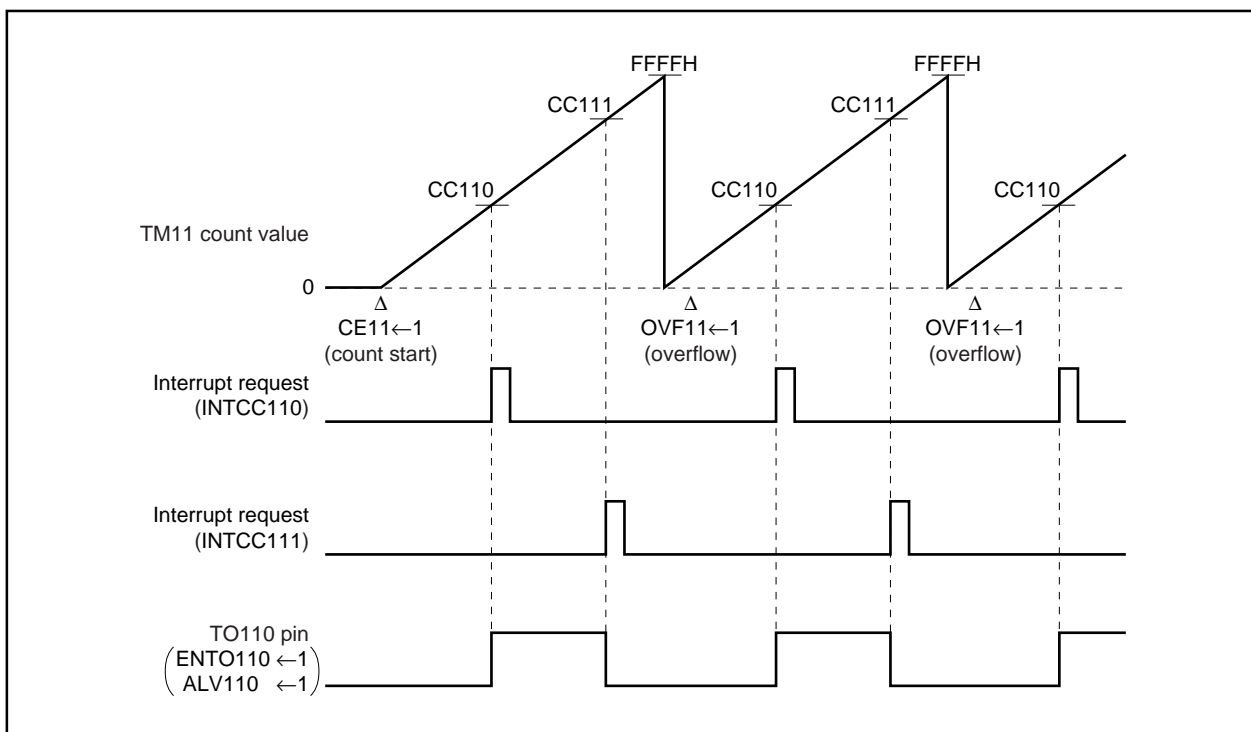


Timer 1 has 3 timer output pins (TO1n0).

The TM1n count value and the CC1n0 value are compared and if they match, the output level of the TO1n0 pin is set. Also, the TM1n count value and the CC1n1 value are compared, and if they match, the TO1n0 pin's output level is reset.

The output level of pin TO1n0 can also be specified by the TOC1n register (n = 0 to 2).

**Figure 9-8. Example of TM11 Compare Operation (Set/Reset Output Mode)**



## 9.5 Timer 4 Operation

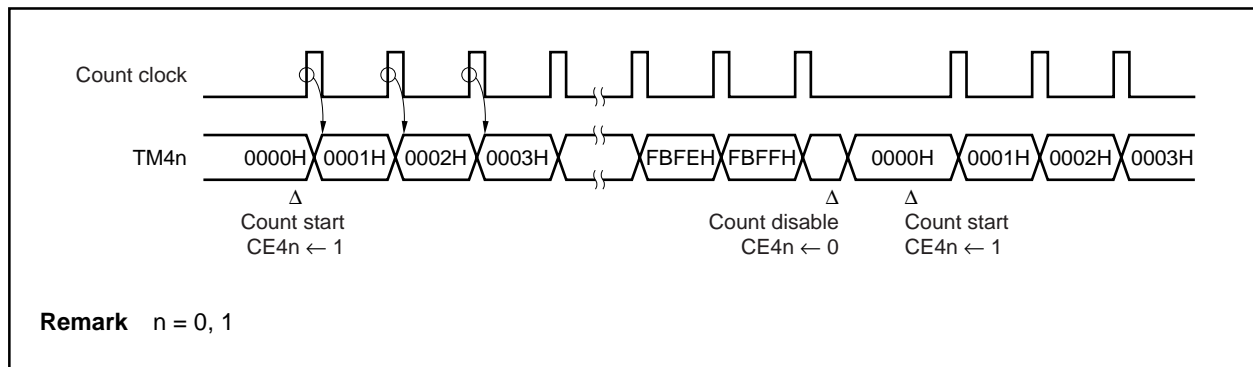
### 9.5.1 Count operation

Timer 4 functions as a 16-bit interval timer. Setting of its operation is specified in timer control register 4n (TMC4n) (n = 0, 1).

In a timer 4 count operation, the internal count clock ( $\phi/32$  to  $\phi/256$ ) specified by the PRS4n0, PRM4n1, and PRM4n0 bits of the TMC4n register is counted up.

If the count results in TM4n match the value in CM4n, TM4n is cleared. At the same time, a matching interrupt (INTCM4n) is generated.

Figure 9-9. Basic Operation of Timer 4



### 9.5.2 Count clock selection

Using the setting of the TMC4n register's PRS4n0, PRM4n1, and PRM4n0 bits, one of four possible internal count clocks,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$  or  $\phi/256$ , can be selected (n = 0, 1).

**Caution** Do not change the count clock during timer operation.

PRS4n0	PRM4n1	PRM4n0	Internal Count Clock
0	0	0	$\phi/32$
0	0	1	$\phi/64$
0	1	0	$\phi/128$
0	1	1	RFU (reserved)
1	0	0	$\phi/64$
1	0	1	$\phi/128$
1	1	0	$\phi/256$
1	1	1	RFU (reserved)

**Remark** n = 0, 1

### 9.5.3 Overflow

If the TM4n overflows as a result of counting the internal count clock, the OVF4n bit of the TOVS register is set (1) (n = 0, 1).

### 9.5.4 Compare operation

In Timer 4, a compare operation which compares the value set in the compare register (CM4n) with the TM4n count value is performed ( $n = 0, 1$ ).

If values are found to match in the compare operation, an interrupt (INTCM4n) is issued. By issuing an interrupt, TM4n is cleared (0) with the following timing (refer to **Figure 9-10 (a)**). Through this function, Timer 4 is used as an interval timer.

CM4n can also be set to 0. In this case, if TM4n overflows and becomes 0, a value match is detected and INTCM4n is issued. Using the following count timing, the TM4n value is cleared (0), but with this match, INTCM4n is not issued (refer to **Figure 9-10 (b)**).

**Figure 9-10. Example of TM40 Compare Operation (1/2)**

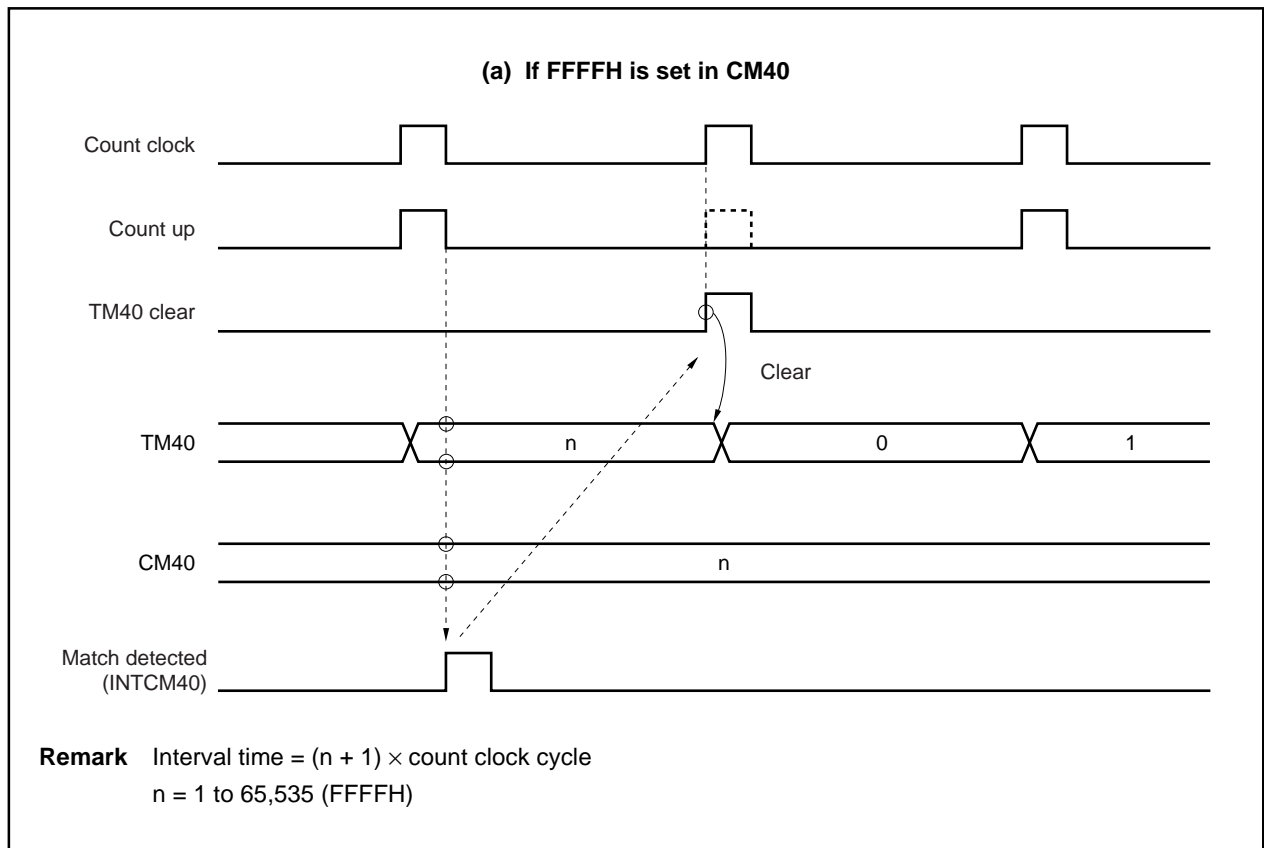
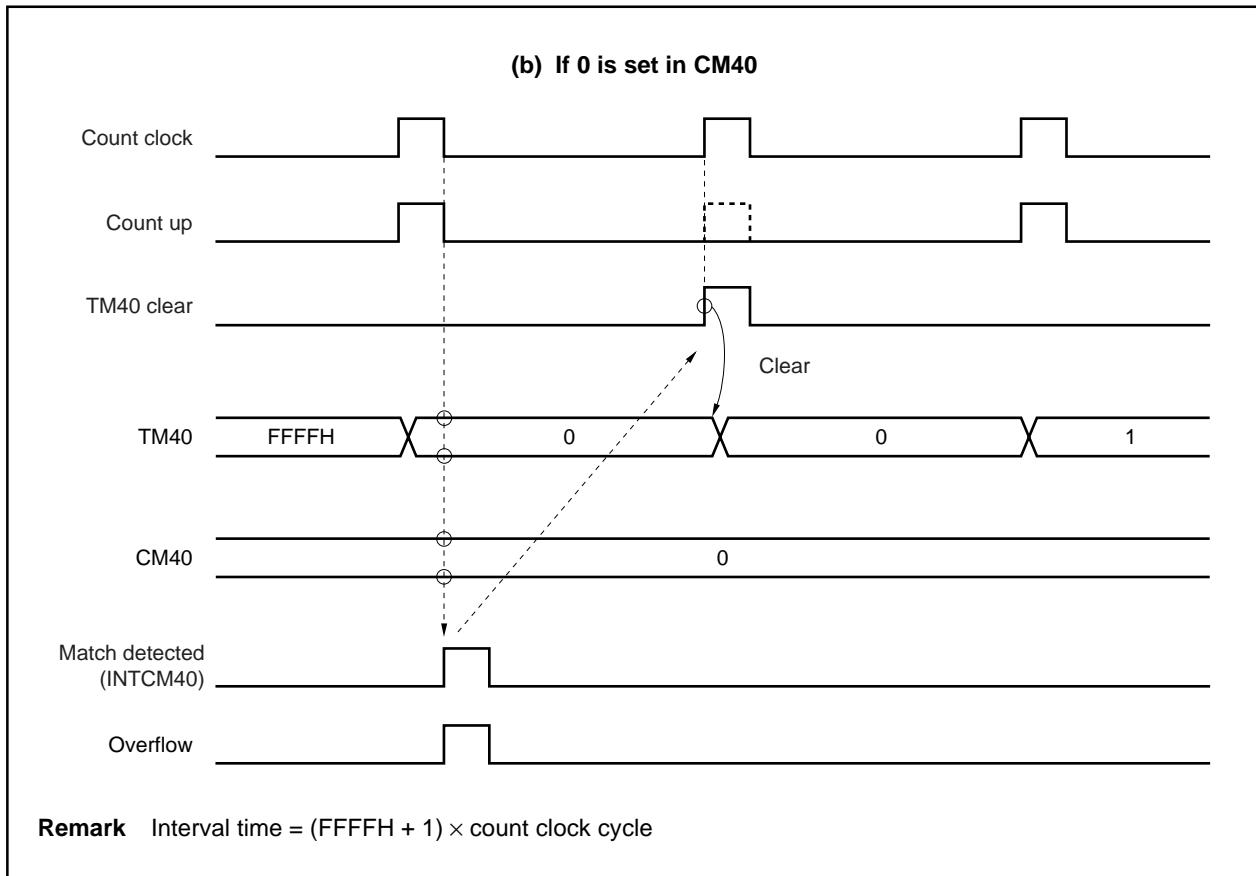


Figure 9-10. Example of TM40 Compare Operation (2/2)



## 9.6 Application Example

### (1) Operation as an interval timer (Timer 4)

In this example, timer 4 is used as an interval timer that repeatedly issues an interrupt at intervals specified by the count time preset in the compare register (CM4n) ( $n = 0, 1$ ).

Figure 9-11. Example of Timing in Interval Timer Operation

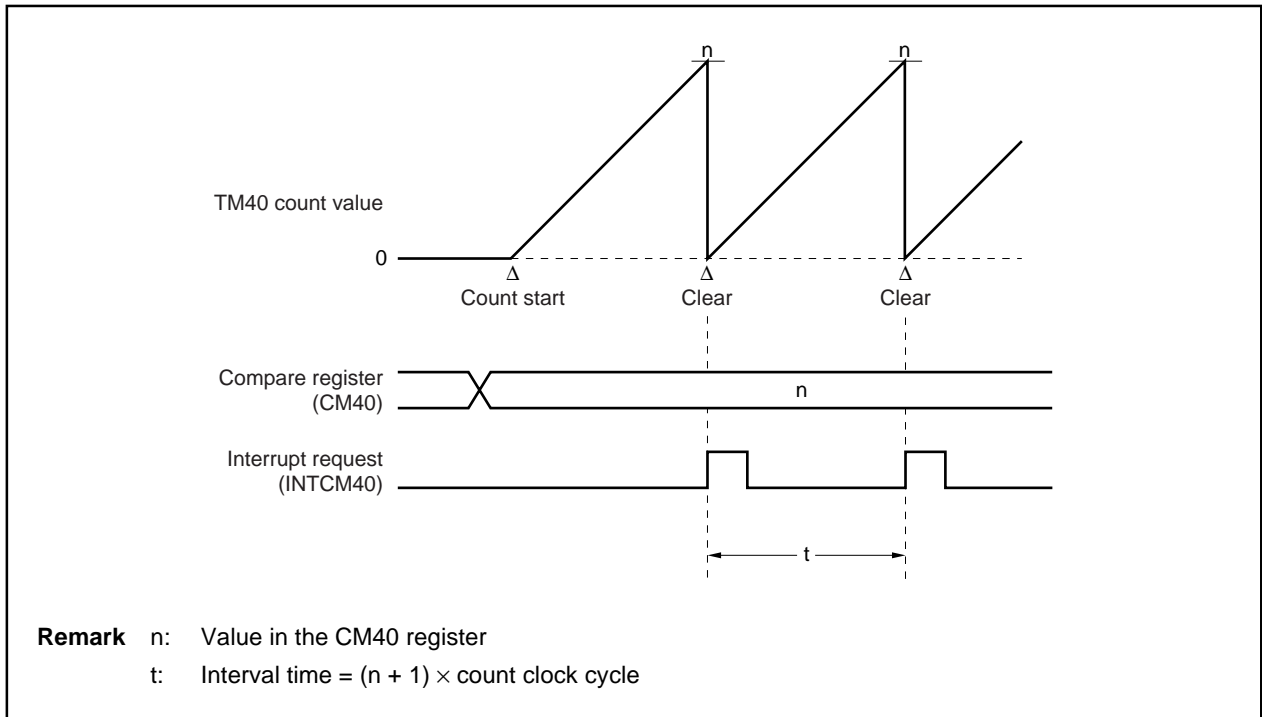
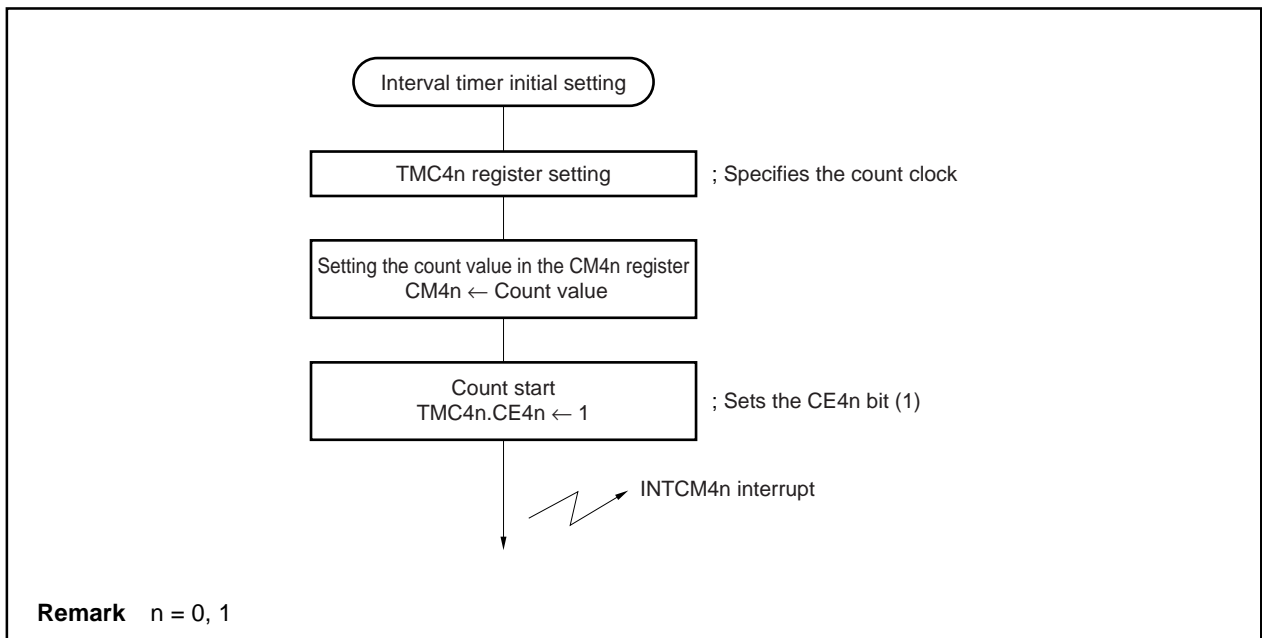


Figure 9-12. Example of Interval Timer Operation Setting Procedure



**(2) Operation for pulse width measurement (Timer 1)**

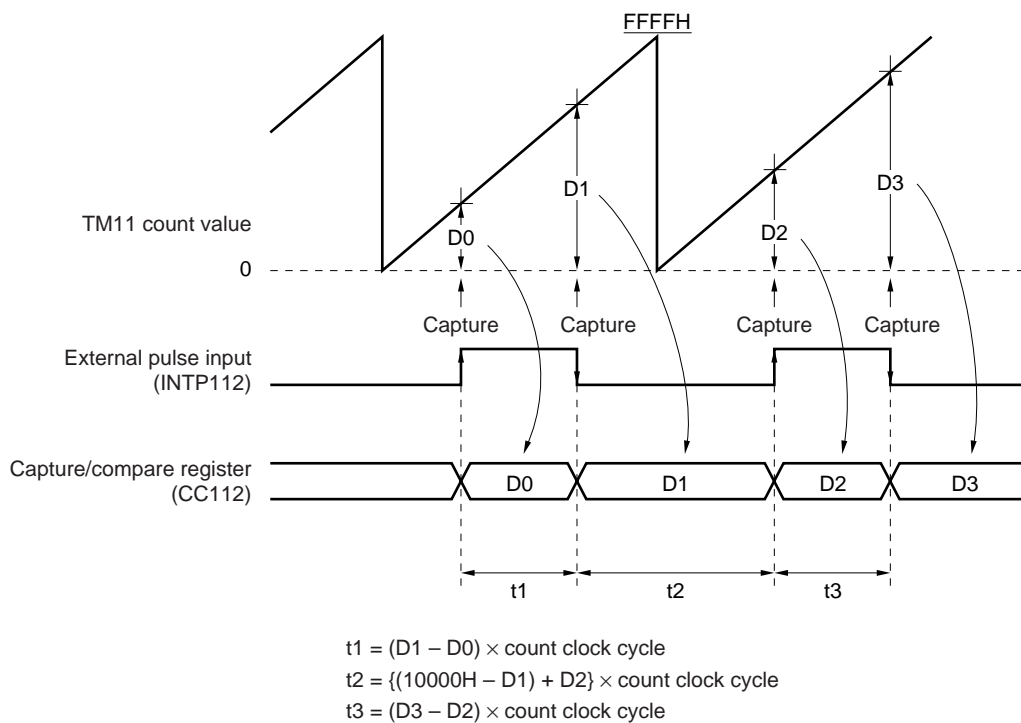
In measuring the pulse width, timer 1 is used.

Here, an example is given of measurement of high level or low level width of an external pulse input to the INTP112 pin.

As shown in Figure 9-13, in synch with the active edge (specified as both the rising edge and falling edge) of the INTP112 pin's input, the value of the counting timer 1 (TM11) is fetched to and held in the capture/compare register (CC112).

The pulse width is calculated by determining the difference between the count value of TM11 captured in the CC112 register through active edge detection the  $n$ th time and the count value ( $D_n - 1$ ) captured through active edge detection the  $(n - 1)$ th time, then multiplying this value by the count clock.

**Figure 9-13. Example of Pulse Measurement Timing**



**Remark** D0 to D3: TM11 count values



Figure 9-14. Example of Pulse Width Measurement Setting Procedure

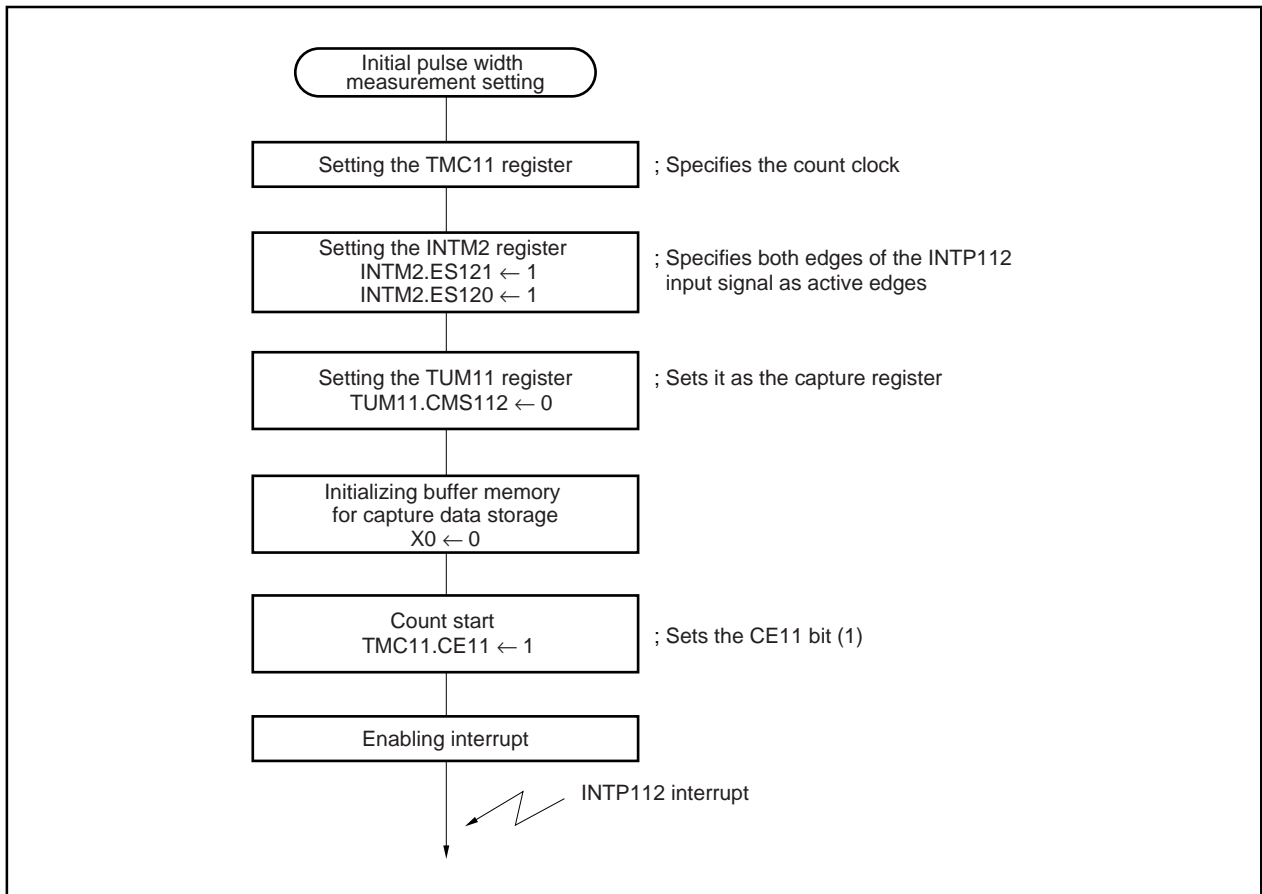
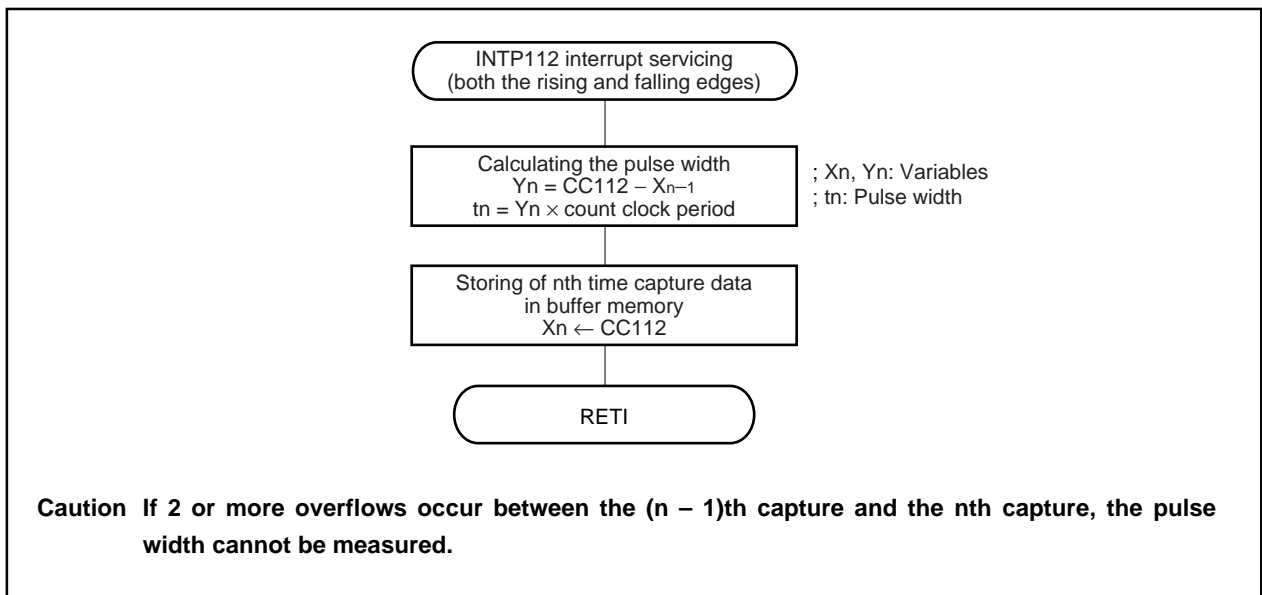


Figure 9-15. Example of Interrupt Request Servicing Routine Which Calculates the Pulse Width



**(3) Operation as a PWM output (Timer 1)**

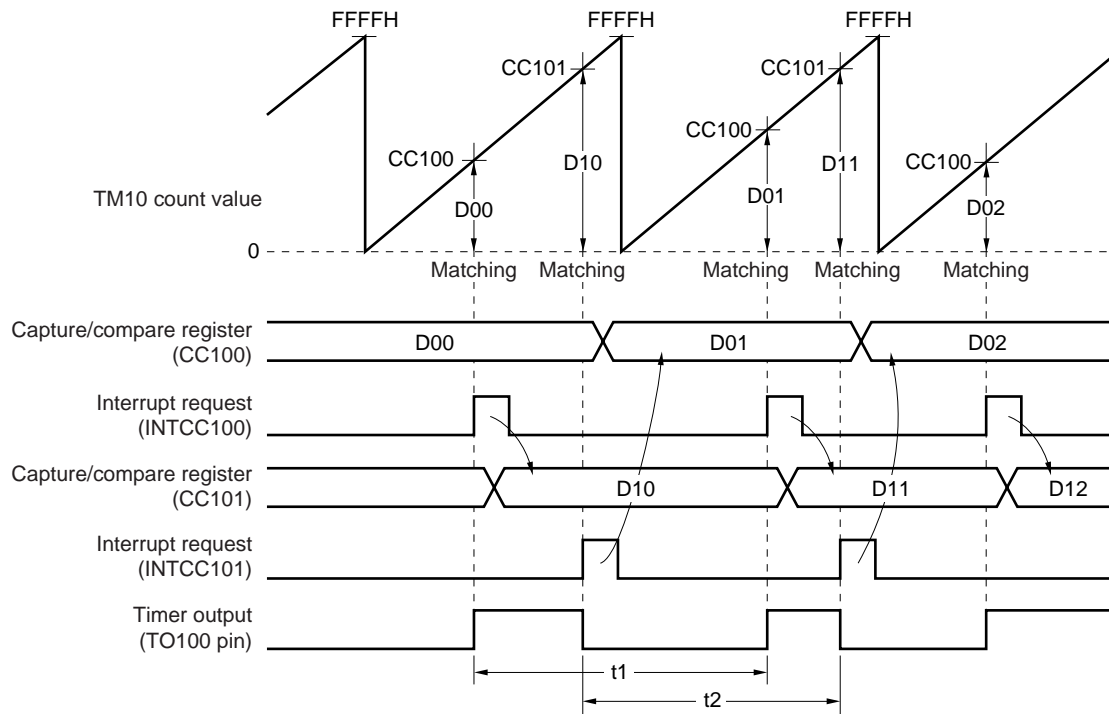
Through a combination of timer 1 and the timer output function, the desired rectangular wave can be output to the timer output pin (TO1n0) and used as a PWM output (n = 0 to 2).

Here an example is shown using the capture/compare registers CC100 and CC101.

In this case, a PWM signal with 16-bit precision can be output from the TO100 pin. The timing is shown in Figure 9-16.

If used as a 16-bit timer, the PWM output's rise timing set in the capture/compare register (CC100) is determined as shown in Figure 9-16, and the fall timing is determined by the value set in the capture/compare register (CC101).

**Figure 9-16. Example of PWM Output Timing**



**Remark** D00 to D02, D10 to D12: Compare register setting values

$$t1 = \{(10000H - D00) + D01\} \times \text{count clock period}$$

$$t2 = \{(10000H - D10) + D11\} \times \text{count clock period}$$

Figure 9-17. Example of PWM Output Setting Procedure

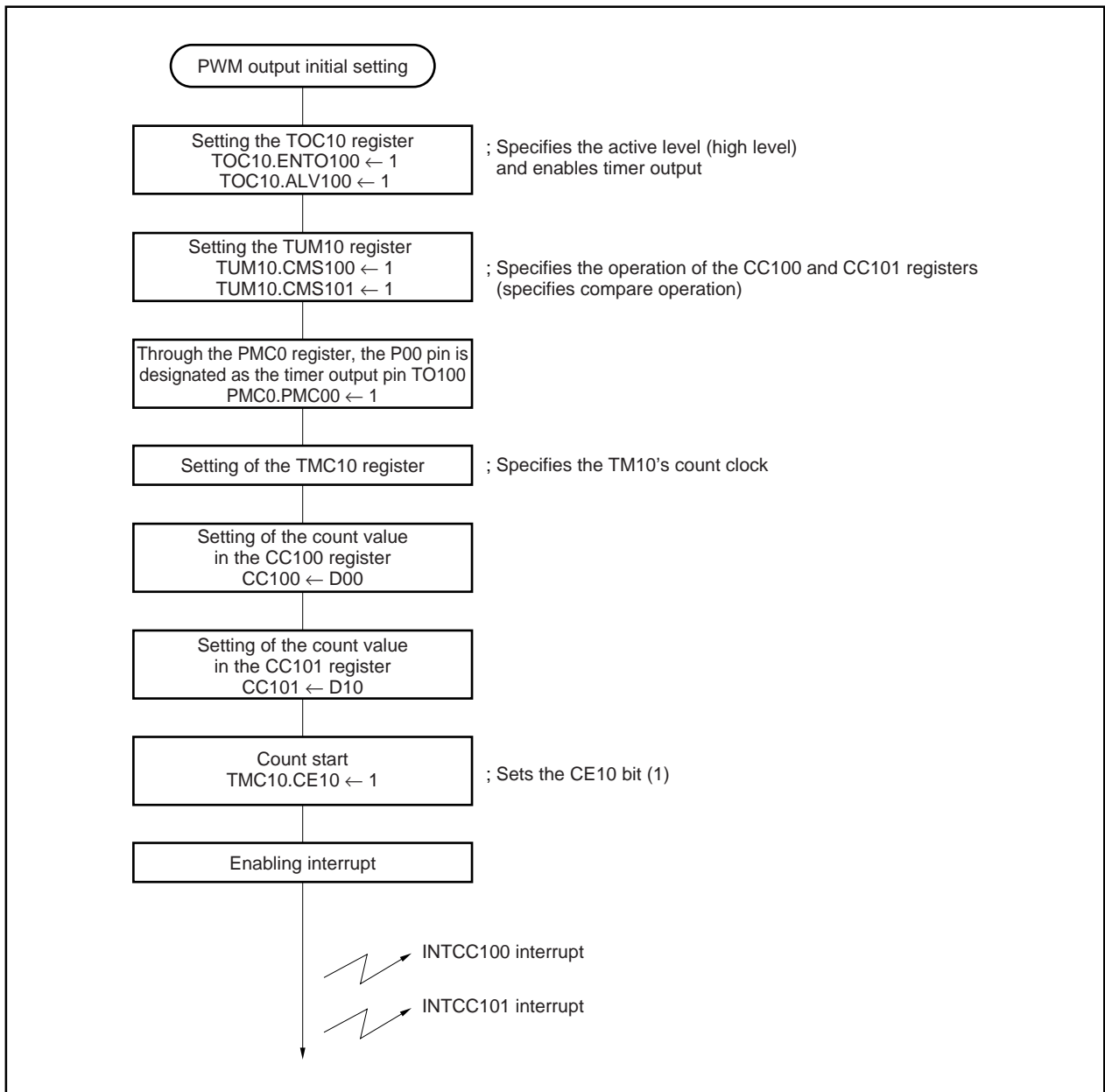
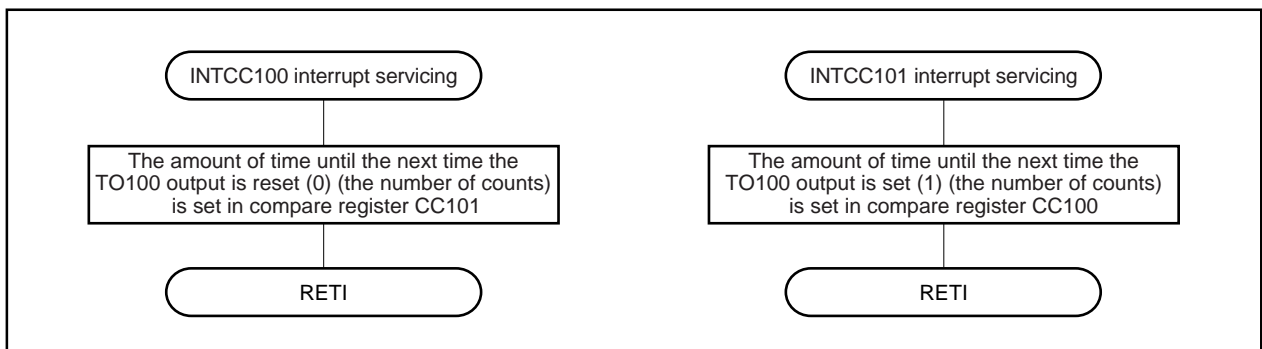


Figure 9-18. Example of Interrupt Request Servicing Routine for Rewriting Compare Value



**(4) Operation for frequency measurement (Timer 1)**

Timer 1 can measure the frequency of an external pulse's input to pins INTP1m0 to INTP1m3 and INTP130 (m = 0, 1).

Here, an example is shown where timer 1 and the capture/compare register CC110 are combined to measure the frequency of an external pulse input to the INTP110 pin with 16-bit precision.

The active edge of the INTP110 input signal is specified to be the rising edge by the INTM2 register.

The frequency is calculated by determining the difference between the TM11 count value ( $D_n$ ) captured in the CC110 register from the  $n$ th rising edge, and the count value ( $D_{n-1}$ ) captured from the rising edge the  $(n - 1)$ th time, then multiplying this value by the count clock.

**Figure 9-19. Example of Frequency Measurement Timing**

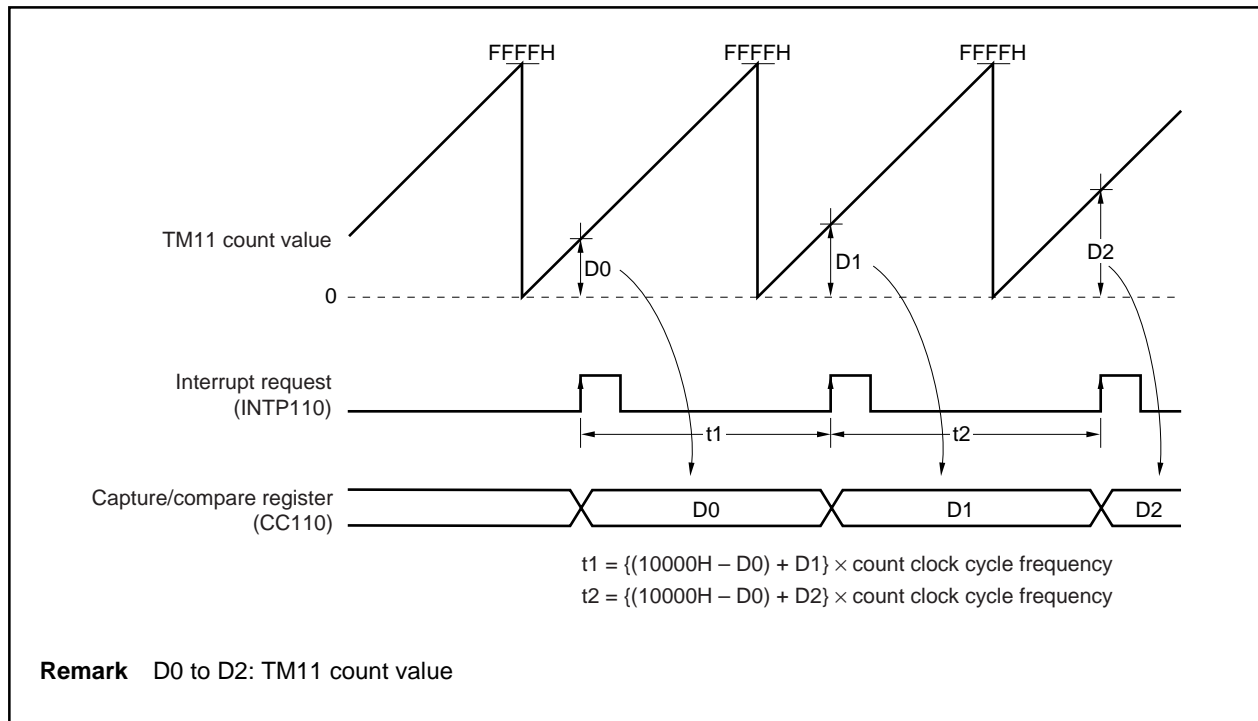


Figure 9-20. Example of Frequency Measurement Setting Procedure

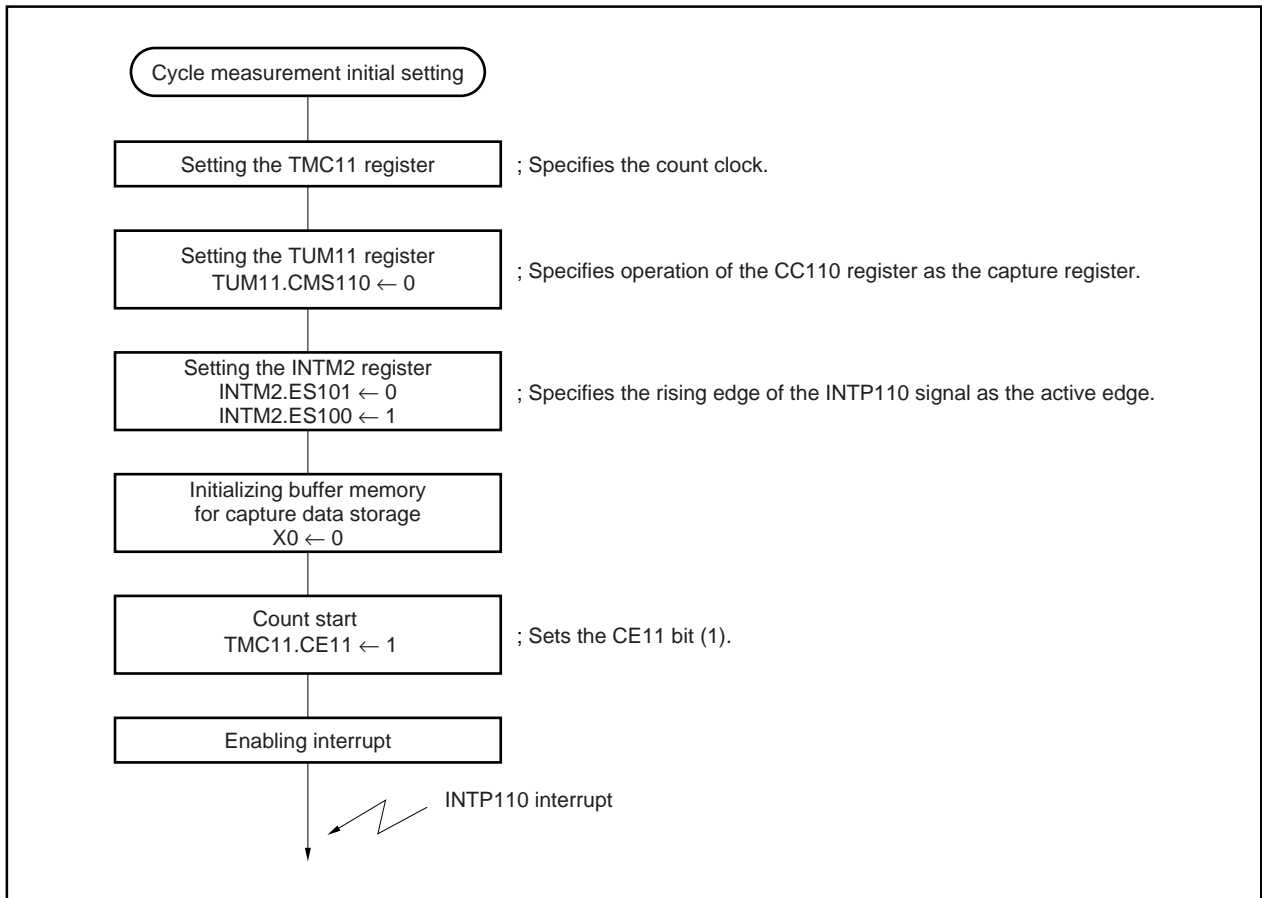
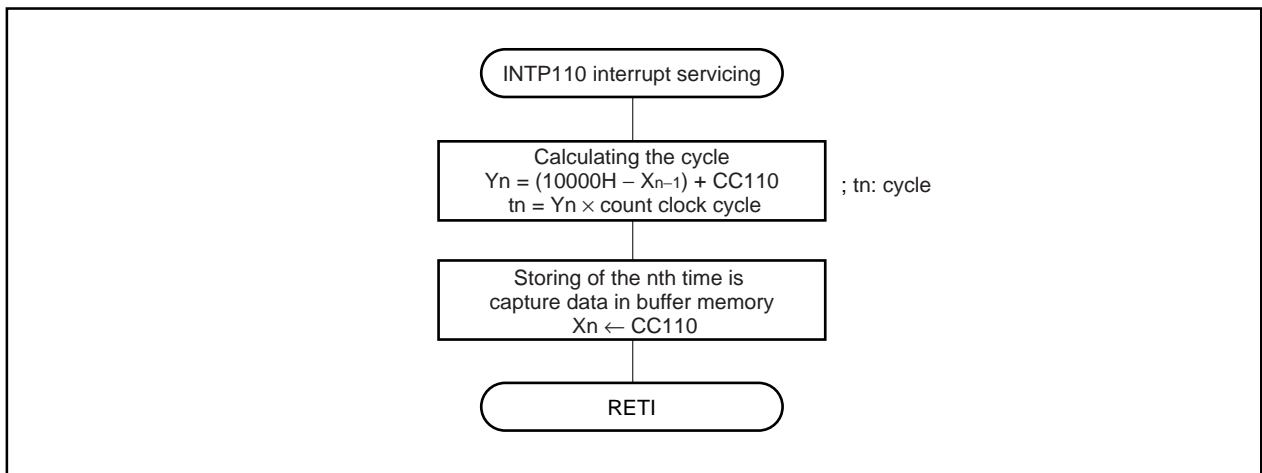


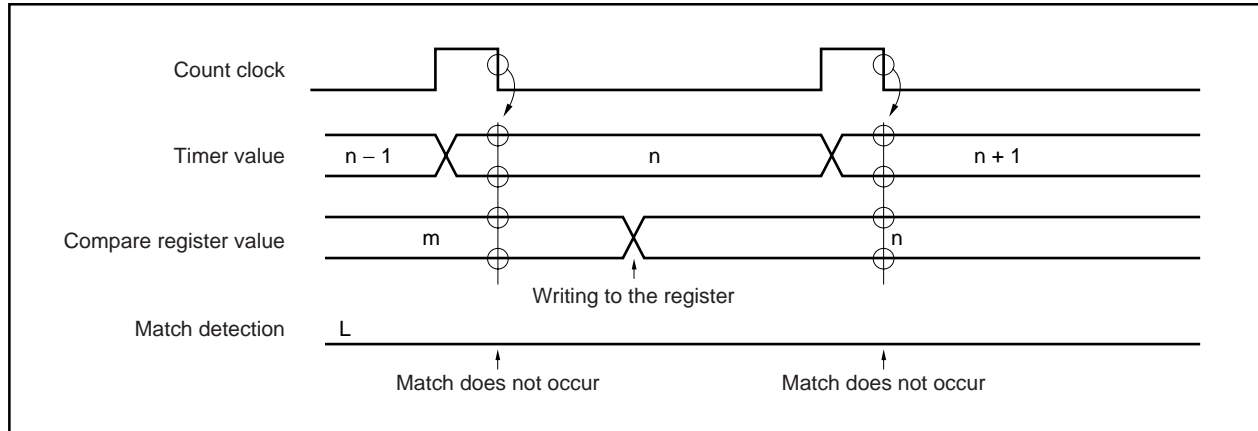
Figure 9-21. Example of Interrupt Request Servicing Routine Which Calculates the Frequency



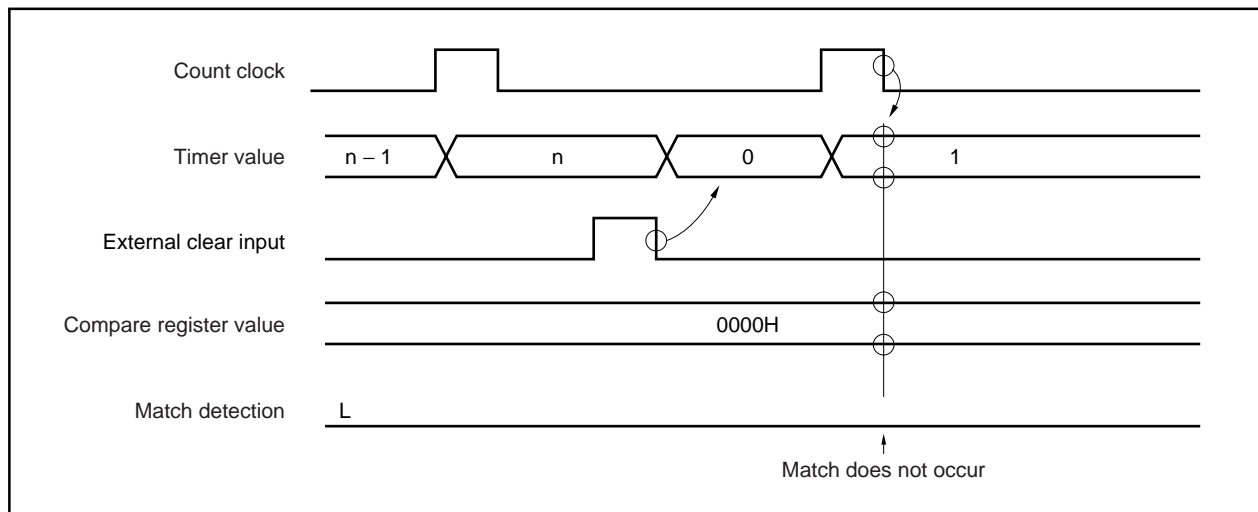
## 9.7 Precaution

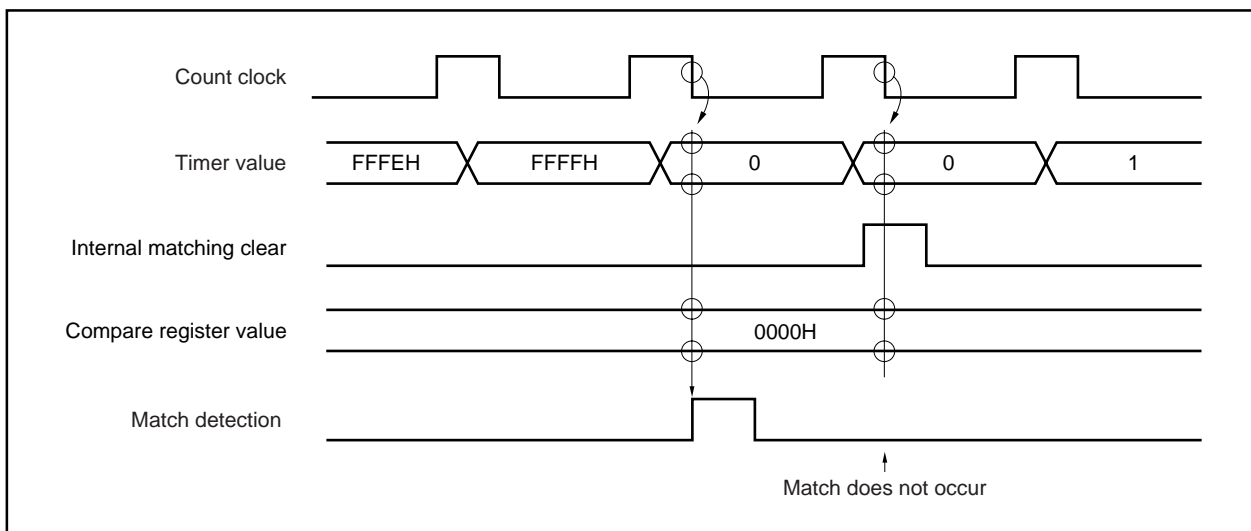
Match detection by the compare register is always performed immediately after timer count up. In the following cases, a match does not occur.

### (1) When rewriting the compare register (TM10 to TM13, TM40, TM41)

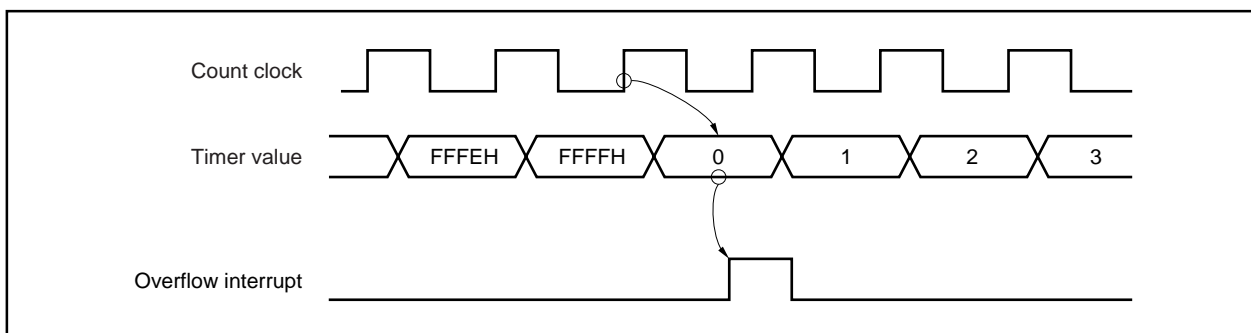


### (2) During external clear (TM10 to TM12)



**(3) When the timer is cleared (TM40, TM41)**

**Remark** When operating timer 1 as the free-running timer, the timer's value becomes 0 when timer overflow occurs.



[MEMO]



## CHAPTER 10 SERIAL INTERFACE FUNCTION

### 10.1 Features

Two types of serial interfaces with 4 transmit/receive channels are provided as the serial interface function, and up to 2 channels can be used simultaneously.

The following two types of interface configuration are provided.

- (1) Asynchronous serial interface (UART0, UART1): 2 channels
- (2) Clocked serial interface (CSI0, CSI1): 2 channels

UART0 and UART1 use the method of transmitting and receiving 1 byte of serial data following the start bit, and full duplex communication is possible.

CSI0 and CSI1 carry out data transfer with 3 types of signal lines, a serial clock ( $\overline{\text{SCK0}}$ ,  $\overline{\text{SCK1}}$ ), serial input (SI0, SI1), and serial output (SO0, SO1) (3-wire serial I/O).

**Caution** UART0 and CSI0, and UART1 and CSI1 share the same pins, the use of which is specified with the ASIM00 and ASIM10 registers.

## 10.2 Asynchronous Serial Interfaces 0, 1 (UART0, UART1)

### 10.2.1 Features

- Transfer rate 150 bps to 153,600 bps (using the exclusive baud rate generator when the internal system clock is 30 MHz)  
Maximum 3.75 Mbps (using the  $\phi/2$  clock when the internal system clock is 30 MHz)
- Full duplex communication On-chip receive buffer (RXBn)
- 2-pin configuration TXDn: Transmit data output pin  
RXDn: Receive data input pin
- Receive error detection functions
  - Parity error
  - Framing error
  - Overrun error
- Interrupt sources: 3 types
  - Receive error interrupt (INTSERn)
  - Reception complete interrupt (INTSRn)
  - Transmission complete interrupt (INTSTn)
- The character length of transmit/receive data is specified by the ASIMn0 and ASIMn1 registers.
- Character length 7, 8 bits  
9 bits (when adding an expansion bit)
- Parity function: odd, even, 0, none
- Transmission stop bit: 1, 2 bits
- On-chip dedicated baud rate generator
- Serial clock (SCKn) output function

**Remark** n = 0, 1

### 10.2.2 Configuration

UARTn is controlled by the asynchronous serial interface mode registers (ASIMn0, ASIMn1) and the asynchronous serial interface status registers (ASISn) (n = 0, 1). Receive data is held in the receive buffer (RXBn) and transmit data is written in the transmit shift registers (TXSn).

The asynchronous serial interface is configured as shown in Figure 10-1.

#### (1) Asynchronous serial interface mode registers (ASIM00, ASIM01, ASIM10, ASIM11)

The ASIMn0 and ASIMn1 registers are 8-bit registers that specify asynchronous serial interface operations.

#### (2) Asynchronous serial interface status registers (ASIS0, ASIS1)

The ASISn registers are registers of flags that show the contents of errors when a receive error occurs and transmission status flags. Each receive error flag is set (1) when a receive error occurs and is cleared (0) by reading of data from the receive buffer (RXBn) or reception of the next new data (if there is an error in the next data, that error flag will not be cleared (0) but left set (1)).

The transmit status flag is set (1) when transmission starts and is cleared (0) when transmission ends.

#### (3) Receive control parity check

Receive operations are controlled according to the contents set in the ASIMn0 and ASIMn1 registers. Also, errors such as parity errors are checked during receive operations. If an error is detected, a value corresponding to the error content is set in the ASISn register.

#### (4) Receive shift register

This is a shift register that converts serial data input to the RXDn pin to parallel data. When 1 byte of data is received, the receive data is transferred to the receive buffer.

This register cannot be directly manipulated.

#### (5) Receive buffers (RXB0, RXB0L, RXB1, RXB1L)

RXBn are 9-bit buffer registers that hold receive data, and when 7 or 8-bit character data is received, a 0 is stored in the higher bits.

During 16-bit access of these registers, specify RXB0 and RXB1, and during lower 8-bit access, specify RXB0L and RXB1L.

In the receive enabled state, 1 frame of receive data is transmitted to the receive buffer from the receive shift register in synchronization with the termination of shift-in processing.

Also, a reception complete interrupt request (INTSRn) is generated when data is transmitted to the receive buffer.

**(6) Transmit shift register (TXS0, TXS0L, TXS1, TXS1L)**

TXSn are 9-bit shift registers for transmit processing. Writing of data to these registers starts a transmit operation.

A transmission complete interrupt request (INTSTn) is generated in synchronization with termination of transmission of 1 frame, which includes TXSn data.

During 16-bit access of these registers, specify TXS0 and TXS1, and during lower 8-bit access, specify TXS0L and TXS1L.

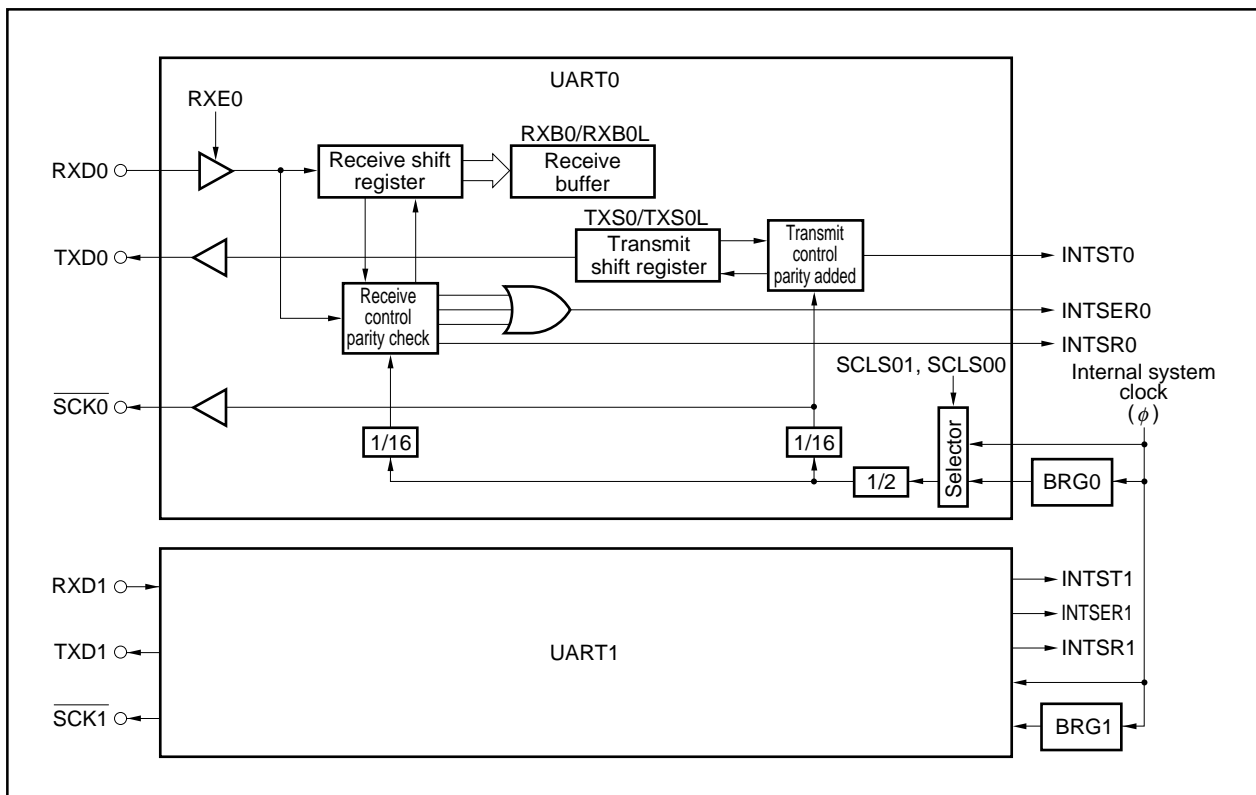
**(7) Adding transmit control parity**

In accordance with the contents set in the ASIMn0 and ASIMn1 registers, start bits, parity bits, stop bits, etc. are added to the data written to the TXSn or TXSnL register, and transmit operation control is carried out.

**(8) Selector**

This selects the serial clock source.

**Figure 10-1. Block Diagram of Asynchronous Serial Interface**



## 10.2.3 Control registers

## (1) Asynchronous serial interface mode registers 00, 01, 10, 11 (ASIM00, ASIM01, ASIM10, ASIM11)

These registers specify the UART0 and UART1 transfer mode.

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

(1/3)

	7	6	5	4	3	2	1	0		
ASIM00	TXE0	RXE0	PS01	PS00	CL0	SL0	SCLS01	SCLS00	Address FFFFFF0C0H	After reset 80H
ASIM10	TXE1	RXE1	PS11	PS10	CL1	SL1	SCLS11	SCLS10	FFFFFF0D0H	80H

Bit Position	Bit Name	Function
7, 6	TXEn, RXEn	Transmit/Receive Enable Specifies the transmission/reception enable status/disable status.

Bit Position	Bit Name	Function															
5, 4	PSn1, PSn0	<p>Parity Select Specifies the parity bit length.</p> <table border="1"> <thead> <tr> <th>PSn1</th><th>PSn0</th><th>Operation</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>No parity, expansion bit operation</td></tr> <tr> <td>0</td><td>1</td><td>Specifies 0 parity Transmission side → Transmits with parity bit at 0. Reception side → Does not generate parity errors during receiving.</td></tr> <tr> <td>1</td><td>0</td><td>Specifies odd parity.</td></tr> <tr> <td>1</td><td>1</td><td>Specifies even parity.</td></tr> </tbody> </table> <ul style="list-style-type: none"> <li> <b>Even parity</b>            If the number of bits whose values are 1 in the transmit data is odd, a parity bit is set (1). If the number of bits whose values are 1 is even, the parity bit is cleared (0). In this way, the number of bits in the transmit data and the parity bit which are 1 is controlled so that it is an even number. During receiving the number of bits in the receive data and parity bit which are 1 is counted, and if it is an odd number, a parity error is generated.         </li> <li> <b>Odd parity</b>            This is the opposite of even parity, with the number of bits in the transmit data and parity bit being controlled so that it is an odd number.            During receiving, if the number of bits in the receive data and parity bit which are 1 turns out to be an even number, a parity error is generated.         </li> <li> <b>0 parity</b>            During transmission, the parity bit is cleared (0) regardless of the transmit data.            During reception, since no parity bit check is performed, no parity error is generated.         </li> <li> <b>No parity</b>            No parity bit is added to transmit data.            During reception, data are received as having no parity bit. Since there is no parity bit, parity errors are not generated.            Expansion bit operations can be specified with the EBSn bit in the ASIMn1 register.         </li> </ul>	PSn1	PSn0	Operation	0	0	No parity, expansion bit operation	0	1	Specifies 0 parity Transmission side → Transmits with parity bit at 0. Reception side → Does not generate parity errors during receiving.	1	0	Specifies odd parity.	1	1	Specifies even parity.
PSn1	PSn0	Operation															
0	0	No parity, expansion bit operation															
0	1	Specifies 0 parity Transmission side → Transmits with parity bit at 0. Reception side → Does not generate parity errors during receiving.															
1	0	Specifies odd parity.															
1	1	Specifies even parity.															
3	CLn	<p>Character Length Specifies the character length of 1 frame.</p> <p>0: 7 bits 1: 8 bits</p>															

**Remark** n = 0, 1

Bit Position	Bit Name	Function																																				
2	SLn	Stop Bit Length Specifies the stop bit length. 0: 1 bit 1: 2 bits																																				
1, 0	SCLSn1, SCLSn0	Serial Clock Source Specifies the serial clock.																																				
		<table><tr><th>SCLSn1</th><th>SCLSn0</th><th>Serial Clock</th></tr><tr><td>0</td><td>0</td><td>Baud rate generator output</td></tr><tr><td>0</td><td>1</td><td><math>\phi/2</math> (<math>\times 16</math> sampling rate)</td></tr><tr><td>1</td><td>0</td><td><math>\phi/2</math> (<math>\times 8</math> sampling rate)</td></tr><tr><td>1</td><td>1</td><td><math>\phi/2</math> (<math>\times 4</math> sampling rate)</td></tr></table>	SCLSn1	SCLSn0	Serial Clock	0	0	Baud rate generator output	0	1	$\phi/2$ ( $\times 16$ sampling rate)	1	0	$\phi/2$ ( $\times 8$ sampling rate)	1	1	$\phi/2$ ( $\times 4$ sampling rate)																					
		SCLSn1	SCLSn0	Serial Clock																																		
		0	0	Baud rate generator output																																		
		0	1	$\phi/2$ ( $\times 16$ sampling rate)																																		
		1	0	$\phi/2$ ( $\times 8$ sampling rate)																																		
		1	1	$\phi/2$ ( $\times 4$ sampling rate)																																		
		<ul style="list-style-type: none"><li><b>In the case of SCLSn1, SCLSn0 = 00</b> <math>\phi/2</math> is selected as the serial clock source. (<math>\phi</math>: internal system clock). In the asynchronous mode, <math>\times 16</math>, <math>\times 8</math> and <math>\times 4</math> sampling rates are used, so the baud rate is expressed by the following formula. <math display="block">\text{Baud rate} = \frac{\phi/2}{\text{sampling rate}} \text{bps}</math> Based on the formula above, the baud rate value in the case where a representative clock is used is shown below.</li></ul>																																				
		<table><tr><th><div>Sampling Rate<sup>Note</sup></div><div>Internal System Clock (<math>\phi</math>)</div></th><th><math>\times 16</math> (01)</th><th><math>\times 8</math> (10)</th><th><math>\times 4</math> (11)</th></tr><tr><td>30 MHz</td><td>937.5 K</td><td>1,875 K</td><td>3,750 K</td></tr><tr><td>25 MHz</td><td>781 K</td><td>1,562 K</td><td>3,125 K</td></tr><tr><td>20 MHz</td><td>625 K</td><td>1,250 K</td><td>2,500 K</td></tr><tr><td>16 MHz</td><td>500 K</td><td>1,000 K</td><td>2,000 K</td></tr><tr><td>12.5 MHz</td><td>390 K</td><td>781 K</td><td>1,562 K</td></tr><tr><td>10 MHz</td><td>312 K</td><td>625 K</td><td>1,250 K</td></tr><tr><td>8 MHz</td><td>250 K</td><td>500 K</td><td>1,000 K</td></tr><tr><td>5 MHz</td><td>156 K</td><td>312 K</td><td>625 K</td></tr></table>	<div>Sampling Rate<sup>Note</sup></div> <div>Internal System Clock (<math>\phi</math>)</div>	$\times 16$ (01)	$\times 8$ (10)	$\times 4$ (11)	30 MHz	937.5 K	1,875 K	3,750 K	25 MHz	781 K	1,562 K	3,125 K	20 MHz	625 K	1,250 K	2,500 K	16 MHz	500 K	1,000 K	2,000 K	12.5 MHz	390 K	781 K	1,562 K	10 MHz	312 K	625 K	1,250 K	8 MHz	250 K	500 K	1,000 K	5 MHz	156 K	312 K	625 K
		<div>Sampling Rate<sup>Note</sup></div> <div>Internal System Clock (<math>\phi</math>)</div>	$\times 16$ (01)	$\times 8$ (10)	$\times 4$ (11)																																	
30 MHz	937.5 K	1,875 K	3,750 K																																			
25 MHz	781 K	1,562 K	3,125 K																																			
20 MHz	625 K	1,250 K	2,500 K																																			
16 MHz	500 K	1,000 K	2,000 K																																			
12.5 MHz	390 K	781 K	1,562 K																																			
10 MHz	312 K	625 K	1,250 K																																			
8 MHz	250 K	500 K	1,000 K																																			
5 MHz	156 K	312 K	625 K																																			
<b>Note</b> Values in ( ) are the set values for the SCLSn1 and SCLSn0 bits																																						
<ul style="list-style-type: none"><li><b>If SCLSn1, SCLSn0 = 00</b> The baud rate generator output is selected as the serial clock source. For details concerning the baud rate generator, refer to <b>10.4 Dedicated Baud Rate Generators 0, 1 (BRG0, BRG1)</b>.</li></ul>																																						

**Caution** UARTn operation is not guaranteed if this register is changed during UARTn transmission or reception. Furthermore, if this register is changed during UARTn transmission or reception, a transmission complete interrupt (INTSTn) is generated during transmission, and a reception complete interrupt (INTSRn) is generated during reception.

**Remark** n = 0, 1

	7	6	5	4	3	2	1	0		
ASIM01	0	0	0	0	0	0	0	EBS0	Address FFFFF0C2H	After reset 00H
ASIM11	0	0	0	0	0	0	0	EBS1	FFFFF0D2H	00H

Bit Position	Bit Name	Function
0	EBSn	<p>Extended Bit Select</p> <p>Specifies transmit/receive data expansion bit operation when no parity operation is specified (PSn1, PSn0 = 00).</p> <p>0: Expansion bit operation disabled.</p> <p>1: Expansion bit operation enabled.</p> <p>When expansion bit is specified, 1 data bit is added to the highest bit of 8-bit transmit/receive data, and communications by 9-bit data are enabled.</p> <p>Expansion bit operation is enabled only in the case where no parity operations have been specified in the ASIMn0 register. If 0 parity, or even/odd parity operation is specified, the EBSn bit specification is made invalid and the expansion bit adding operation is not performed.</p>

**Caution** UARTn operation when this register has been changed during UARTn transmission/ reception is not guaranteed.

**Remark** n = 0, 1



## (2) Asynchronous serial interface status registers 0, 1 (ASIS0, ASIS1)

These registers are configured with 3-bit error flags (PE<sub>n</sub>, FE<sub>n</sub>, OVE<sub>n</sub>), which show the error status when UART<sub>n</sub> reception is terminated, and a transmit status flag (SOT<sub>n</sub>) (n = 0,1).

The status flag that shows a receive error always shows the state of the error that occurred most recently. That is, if the same error occurred several times before reading of receive data, this flag would hold the status of the error that occurred most recently.

If a receive error occurs, after reading the ASIS<sub>n</sub> register, read the receive buffer (RXB<sub>n</sub> or RXB<sub>n</sub>L) and clear the error flag.

These are read-only registers in 8- or 1-bit units.

	7	6	5	4	3	2	1	0		
ASIS0	SOT0	0	0	0	0	PE0	FE0	OVE0	Address FFFFF0C4H	After reset 00H
ASIS1	SOT1	0	0	0	0	PE1	FE1	OVE1	FFFFF0D4H	00H

Bit Position	Bit Name	Function
7	SOT <sub>n</sub>	Status Of Transmission This is a status flag that shows the transmission operation's state. Set (1): Transmission start timing (writing to the TXS <sub>n</sub> or TXS <sub>n</sub> L register) Clear (0): Transmission end timing (generation of the INTST <sub>n</sub> interrupt) When about to start serial data transmission, use this as a means of judging whether writing to the transmit shift register is enabled or not.
2	PE <sub>n</sub>	Parity Error This is a status flag that shows a parity error. Set (1): When transmit parity and receive parity do not match. Clear (0): Data are read from the receive buffer and processed.
1	FE <sub>n</sub>	Framing Error This is a status flag that shows a framing error. Set (1): When a stop bit was not detected. Clear (0): Data are read from the receive buffer and processed.
0	OVE <sub>n</sub>	Overrun Error This is a status flag that shows an overrun error. Set (1): When UART <sub>n</sub> has finished the next receiving processing before fetching receive data from the receive buffer. Clear (0): Data are read from the receive buffer and processed. Furthermore, due to the configuration where 1 frame at a tie is received, then the contents of the receive shift register are transmitted to the receive buffer, when an overrun error has occurred, the next receive data is written over the data existing in the receive buffer, and the previous receive data is discarded.

**Remark** n = 0, 1

**(3) Receive buffers 0, 0L, 1, 1L (RXB0, RXB0L, RXB1, RXB1L)**

RXBn are 9-bit buffer registers that hold receive data, with a 0 stored in the higher bits when 7 or 8-bit character data is received (n = 0, 1).

During 16-bit access of these registers, specify RXB0 and RXB1, and during lower 8-bit access, specify RXB0L and RXB1L.

While in the reception enabled state, receive data is transmitted from the receive shift register to the receive buffer in synchronization with the end of shift-in processing of 1 frame.

Also, a reception complete interrupt request (INTSRn) is generated by transfer of receive data to the receive buffer.

In the reception disabled state, transmission of receive data to the receive buffer is not performed even if shift-in processing of 1 frame is completed, and the contents of the receive buffer are held.

Also, a reception complete interrupt request is not generated.

RXB0 and RXB1 are read-only registers in 16-bit units, and RXB0L and RXB1L are read-only registers in 8- or 1-bit units.

RXB0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address FFFFF0C8H	After reset Undefined
	0	0	0	0	0	0	0	RXE0	RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0		
									7	6	5	4	3	2	1	0		
								RXB0L	RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0	FFFFF0CAH	Undefined
RXB1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address FFFFF0D8H	Undefined
	0	0	0	0	0	0	0	RXE1	RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0		
									7	6	5	4	3	2	1	0		
								RXB1L	RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0	FFFFF0DAH	Undefined

Bit Position	Bit Name	Function
8	RXEBn	Receive Extended Buffer This is the expansion bit during reception of 9-bit/character data. A 0 can be read in reception of 7 and 8-bit/character data.
7 to 0	RXBn7 to RXBn0	Receive Buffer This stores receive data. A 0 can be read when RXBn7 is receiving 7-bit/character data.

**Remark** n = 0, 1

#### (4) Transmit shift registers 0, 0L, 1, 1L (TXS0, TXS0L, TXS1, TXS1L)

TXSn are 9-bit shift registers for transmission processing and when transmission is enabled, transmission operations are started ( $n = 0, 1$ ) by writing of data to these registers.

When transmission is disabled, the values are disregarded even if writing is performed.

A transmission complete interrupt request (INTSTn) is generated in synchronization with the end of transmission of 1 frame including TXS data.

During 16-bit access of these registers, specify TXS0 and TXS1, and during lower 8-bit access, specify TXS0L and TXS1L.

TXS0 and TXS1 are write-only registers in 16-bit units, and TXS0L and TXS1L are write-only registers in 8-bit units.

TXS0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address FFFFF0CCH	After reset Undefined
	0	0	0	0	0	0	0	TXED0	TXS07	TXS06	TXS05	TXS04	TXS03	TXS02	TXS01	TXS00		
									7	6	5	4	3	2	1	0		
								TXS0L	TXS07	TXS06	TXS05	TXS04	TXS03	TXS02	TXS01	TXS00	FFFFF0CEH	Undefined
TXS1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address FFFFF0DCH	After reset Undefined
	0	0	0	0	0	0	0	TXED1	TXS17	TXS16	TXS15	TXS14	TXS13	TXS12	TXS11	TXS10		
									7	6	5	4	3	2	1	0		
								TXS1L	TXS17	TXS16	TXS15	TXS14	TXS13	TXS12	TXS11	TXS10	FFFFF0DEH	Undefined

Bit Position	Bit Name	Function
8	TXEDn	Transmit Extended Data This is the expansion bit during transmission of 9-bit/character data.
7 to 0	TXSn7 to TXSn0 ( $n = 0, 1$ )	Transmit Shifter This writes transmission data.

- Cautions**
1. UARTn does not have a transmit buffer, so there is no interrupt request at the end of transmission (to the buffer), and an interrupt request (INTSTn) is generated in synchronization with the end of transmission of 1 frame of data.
  2. If the UARTn registers are changed during transmission, UARTn operation is not guaranteed.

**Remark**  $n = 0, 1$

### 10.2.4 Interrupt request

UARTn generates the following three types of interrupt requests ( $n = 0, 1$ ).

- Receive error interrupt (INTSERn)
- Reception complete interrupt (INTSRn)
- Transmission complete interrupt (INTSTn)

The priority order of these three interrupts is, from high to low: receive error interrupt, reception complete interrupt, transmission complete interrupt.

**Table 10-1. Default Priority of Interrupt**

Interrupt	Priority
Receive error	1
Reception complete	2
Transmission complete	3

#### (1) Receive error interrupt (INTSERn)

In the reception enabled state, a receive error interrupt is generated by ORing the three receive errors.

In the reception disabled state, no receive error interrupt is generated.

#### (2) Reception completion interrupt (INTSRn)

In the reception enabled state, a reception complete interrupt is generated when data is shifted into the receive shift register and transferred to the receive buffer.

This reception complete interrupt request is also generated when a receive error has occurred, but the receive error interrupt has a higher servicing priority.

In the reception disabled state, no reception complete interrupt is generated.

#### (3) Transmission completion interrupt (INTSTn)

As this UARTn has no transmit buffer, a transmission complete interrupt is generated when one frame of transmit data containing a 7-, 8-, or 9-bit character is shifted out of the transmit shift register.

A transmission complete interrupt is output at the start of transmission of the last bit of transmit data.

### 10.2.5 Operation

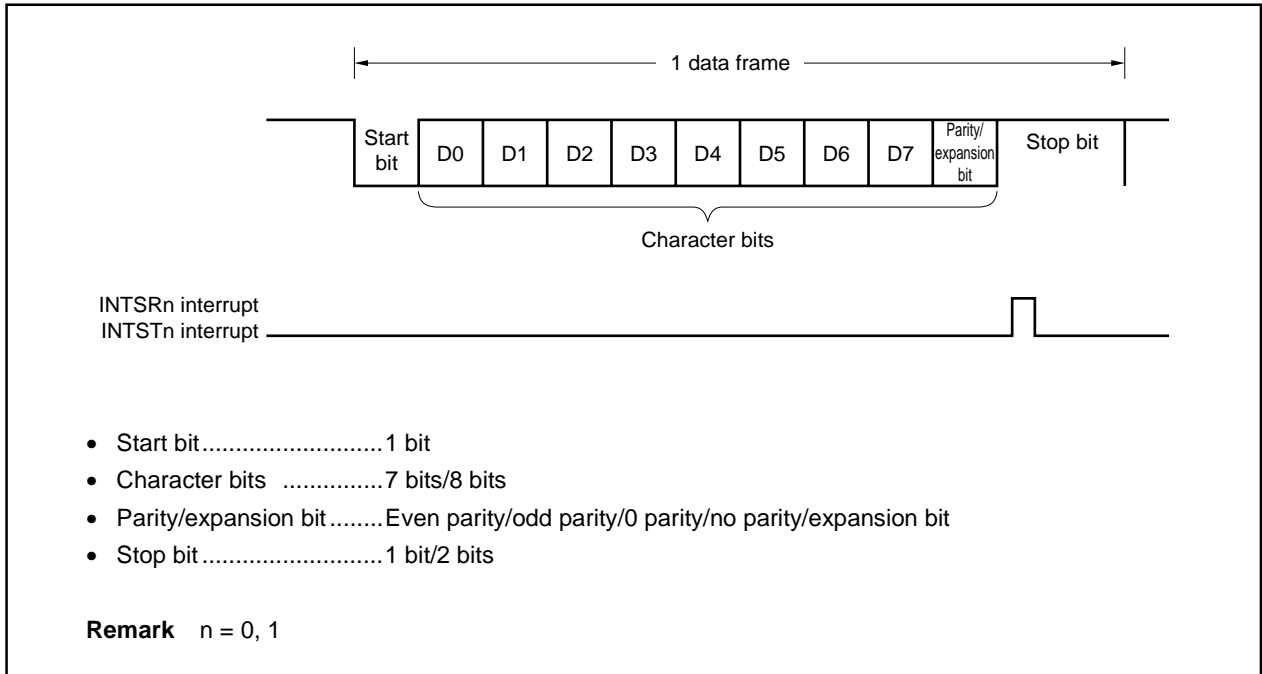
#### (1) Data format

Transmission and reception of full duplex serial data are performed.

As shown in Figure 10-2, 1 data frame consists of a start bit, character bits, a parity bit, and a stop bit as the format of transmit/receive data.

Specification of the character bit length within 1 data frame, parity selection and specification of the stop bit length are performed by the asynchronous serial interface mode register (ASIMn0, ASIMn1) ( $n = 0, 1$ ).

**Figure 10-2. Transmission/Reception Data Format of Asynchronous Serial Interface**



#### (2) Transmission

Transmission starts when data is written to the transmit shift register (TXSn or TXSnL). With the transmission complete interrupt (INTSTn) servicing routine, the next data is written to the TXSn or TXSnL register ( $n = 0, 1$ ).

##### (a) Transmit enable state

This is set with the TXEn bit of the ASIMn0 register.

TXEn = 1: Transmit enabled state

TXEn = 0: Transmit disabled state

However, when setting the transmit enabled state, be sure to set both the CTXEn and CRXEn bits of the clocked serial interface mode register (CSIMn) of the channel in use to 0.

Note that since UARTn does not have CTS (transmit enabled signal) input pins, when the opposite party wants to confirm the reception enabled state, use a port.

**(b) Starting a transmit operation**

In the transmit enabled state, if data is written to the transmit shift register (TXSn or TXSnL), the transmit operation starts. Transmit data is transmitted from the start bit to the LSB header. A start bit, parity/expansion bit and stop bit are added automatically.

In the transmit disabled state, data is not written to the transmit shift register. Even if writing is done, the values are disregarded.

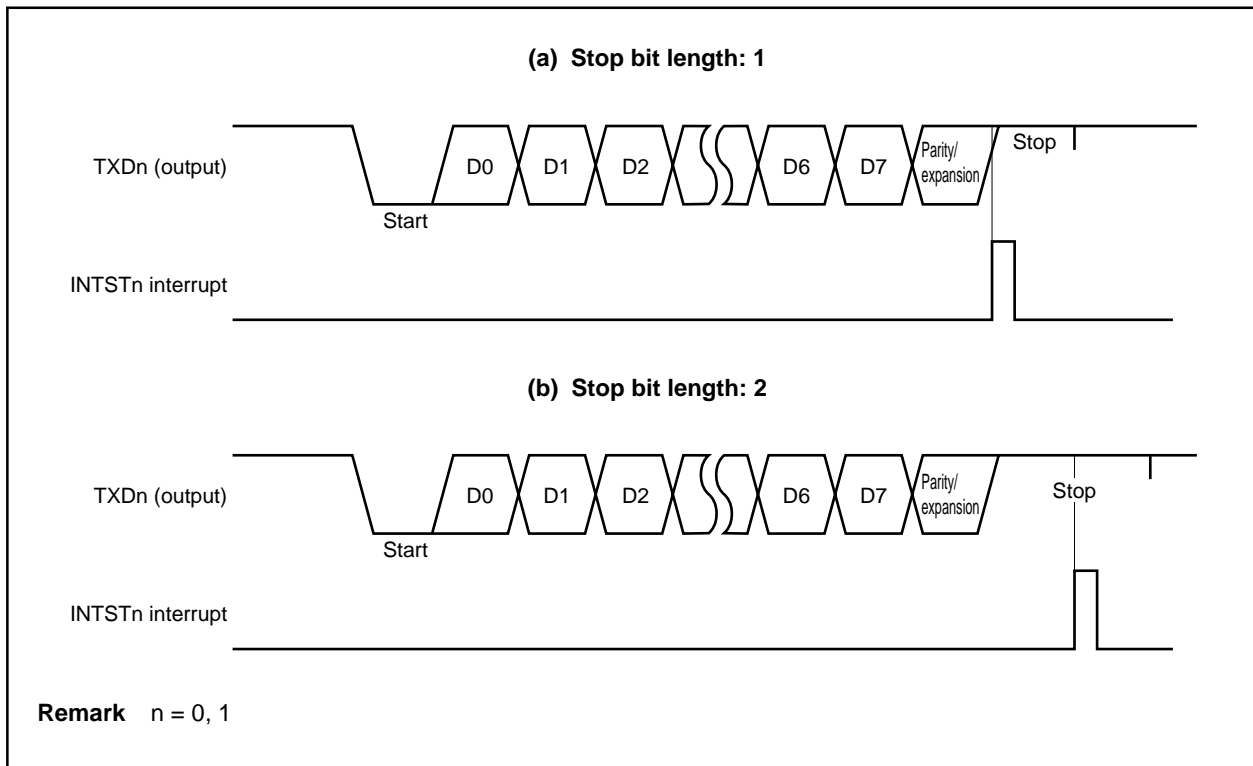
**(c) Transmission interrupt request**

If the transmit shift register (TXSn or TXSnL) becomes empty, a transmission complete interrupt request (INTSTn) is generated.

If the next transmit data is not written to the TXSn or TXSnL register, the transmit operation is interrupted. After 1 transmission is ended, the transmission rate drops if the next transmit data is not written to the TXSn or TXSnL register immediately.

- Cautions**
1. Normally, when the transmit shift register (TXSn or TXSnL) has become empty, a transmission complete interrupt (INTSTn) is generated. However, when  $\overline{\text{RESET}}$  is input, if the transmit shift register (TXSn or TXSnL) has become empty, a transmission complete interrupt (INTSTn) is not generated.
  2. During a transmit operation before INTSTn generation, even if data is written to the TXSn or TXSnL register, the written data is invalid.

**Figure 10-3. Asynchronous Serial Interface Transmission Completion Interrupt Timing**



**(3) Reception**

If reception is enabled, sampling of the RXDn pin is started and if a start bit is detected, data reception begins. When 1 frame of data reception is completed, the reception complete interrupt (INTSRn) is generated. Normally, with this interrupt servicing, receive data is transmitted from the receive buffer (RXBn or RXBnL) to memory (n = 0, 1).

**(a) Receive enabled state**

Reception is enabled when the RXEn bit of the ASIMn0 register is set to 1.

RXEn = 1: Receive enabled state

RXEn = 0: Receive disabled state

However, when reception is enabled, be sure to set both the CTXEn and CRXEn bits of the clocked serial interface mode register (CSIMn) of the channel in use to 0.

In the receive disabled state, the reception hardware stands by in the initial state.

At this time, no reception complete interrupts or reception error interrupts are generated, and the contents of the receive buffer are retained.

**(b) Start of receive operation**

The receive operation is started by detection of the start bit.

The RXDn pin is sampled using the serial clock from the baud rate generator (BRGn). When an RXDn pin low level is detected, the RXDn pin is sampled again after 8 serial clock cycles. If it is low this is recognized as a start bit, the receive operation is started and the RXDn pin input is subsequently sampled at intervals of 16 serial clock cycles.

If the RXDn pin input is found to be high when sampled again 8 serial clock cycles after an RXDn pin low level is detected, this low level is not recognized as a start bit, the operation is stopped by initializing the serial clock counter for sample timing generation, and the unit waits for the next low-level input.

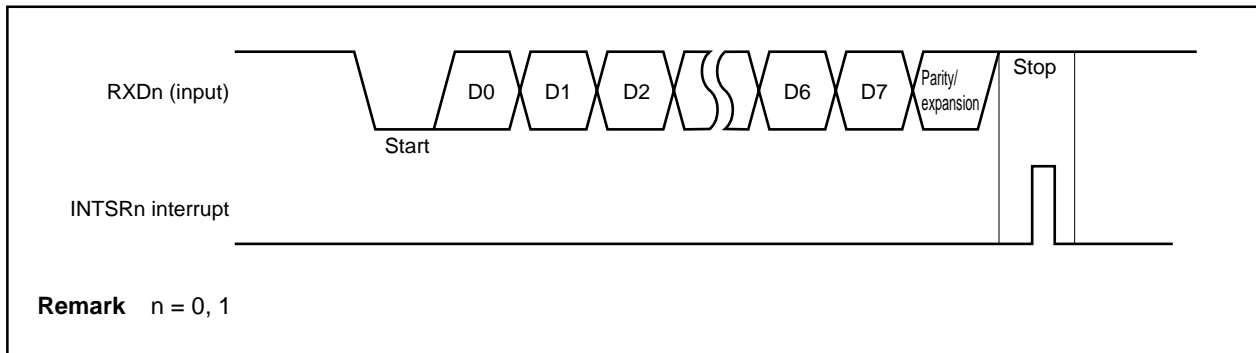
**(c) Reception complete interrupt request**

When RXEn = 1, after one frame of data has been received, the receive data in the shift register is transferred to RXBn and RXBnL a reception complete interrupt request (INTSRn) is generated.

Also, even if an error occurs, the receive data where the error occurred is transmitted to the receive buffer (RXBn or RXBnL) and a reception complete interrupt (INTSRn) and receive error interrupt (INTSERn) are generated simultaneously.

Furthermore, if the RXEn bit is reset (0) during a receive operation, the receive operation is stopped immediately. At this time, the contents of the receive buffer (RXBn or RXBnL) and the asynchronous serial interface status register (ASISn) do not change and the reception complete interrupt (INTSRn) and receive error interrupt (INTSERn) are not generated.

When RXEn = 0 and reception is disabled, a reception complete interrupt request is not generated.

**Figure 10-4. Asynchronous Serial Interface Reception Complete Interrupt Timing****(d) Receive error flag**

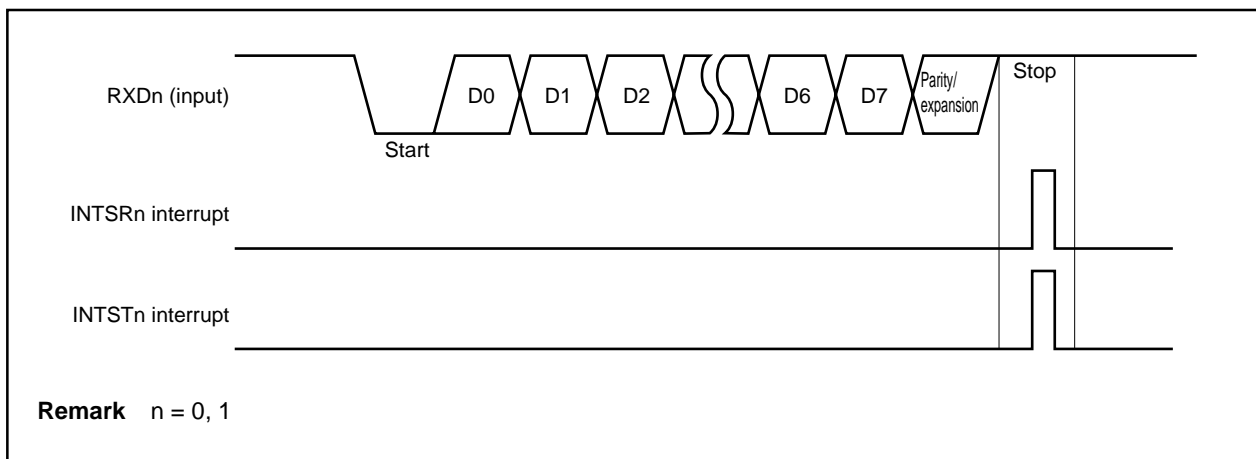
In synchronization with the receive operation, three types of error flags, the parity error flag, framing error flag, and overrun error flag, are affected.

A receive error interrupt request is generated by ORing these three error flags.

By reading out the contents of the ASISn register in the receive error interrupt (INTSRn), which error occurred during reception can be detected.

As for the contents of the ASISn register, either the receive buffer (RXBn or RXBnL) are read or it is reset (0) by reception of the next data (if there is an error in the next receive data, that error flag is set).

Receiving Error	Cause
Parity error	The parity specification during transmission does not match with the parity of the receive data.
Framing error	A stop bit was not detected.
Overrun error	Reception of the next data was completed before data was read from the receive buffer.

**Figure 10-5. Receive Error Timing**



### 10.3 Clocked Serial Interfaces 0, 1 (CSI0, CSI1)

#### 10.3.1 Features

- High transfer rate Max. 7.5 Mbps (when the internal system clock is operating at 30 MHz)
- Half-duplex communications
- Character length: 8 bits
- It is possible to switch MSB first or LSB first for data.
- Either external serial clock input or internal serial clock output can be selected.
- 3-wire type
  - SOn: Serial data output
  - SIn: Serial data input
  - $\overline{\text{SCKn}}$ : Serial clock input/output
- Interrupt source 1 type
  - Transmission/reception complete interrupt (INTCSIn)

**Remark** n = 0, 1

#### 10.3.2 Configuration

CSIn are controlled by the clocked serial interface mode registers (CSIMn). Transmission/reception data can be read from and written to the SIO<sub>n</sub> registers (n = 0, 1).

##### (1) Clocked serial interface mode registers (CSIM0, CSIM1)

The CSIMn registers are 8-bit registers that specify CSIn operations.

##### (2) Serial I/O shift registers (SIO0, SIO1)

The SIO<sub>n</sub> registers are 8-bit registers that convert serial data to parallel data. SIO<sub>n</sub> is used for both transmission and reception.

Data is shifted in (received) or shifted out (transmitted) either from the MSB side or the LSB side.

Actual transmitting/receiving operations are controlled by reading from or writing to SIO<sub>n</sub>.

##### (3) Selector

This selects the serial clock to be used.

##### (4) Serial clock controller

This performs control of supply to the serial clock shift register. Also, when the internal clock is used, it controls the clock that outputs to the  $\overline{\text{SCKn}}$  pin.

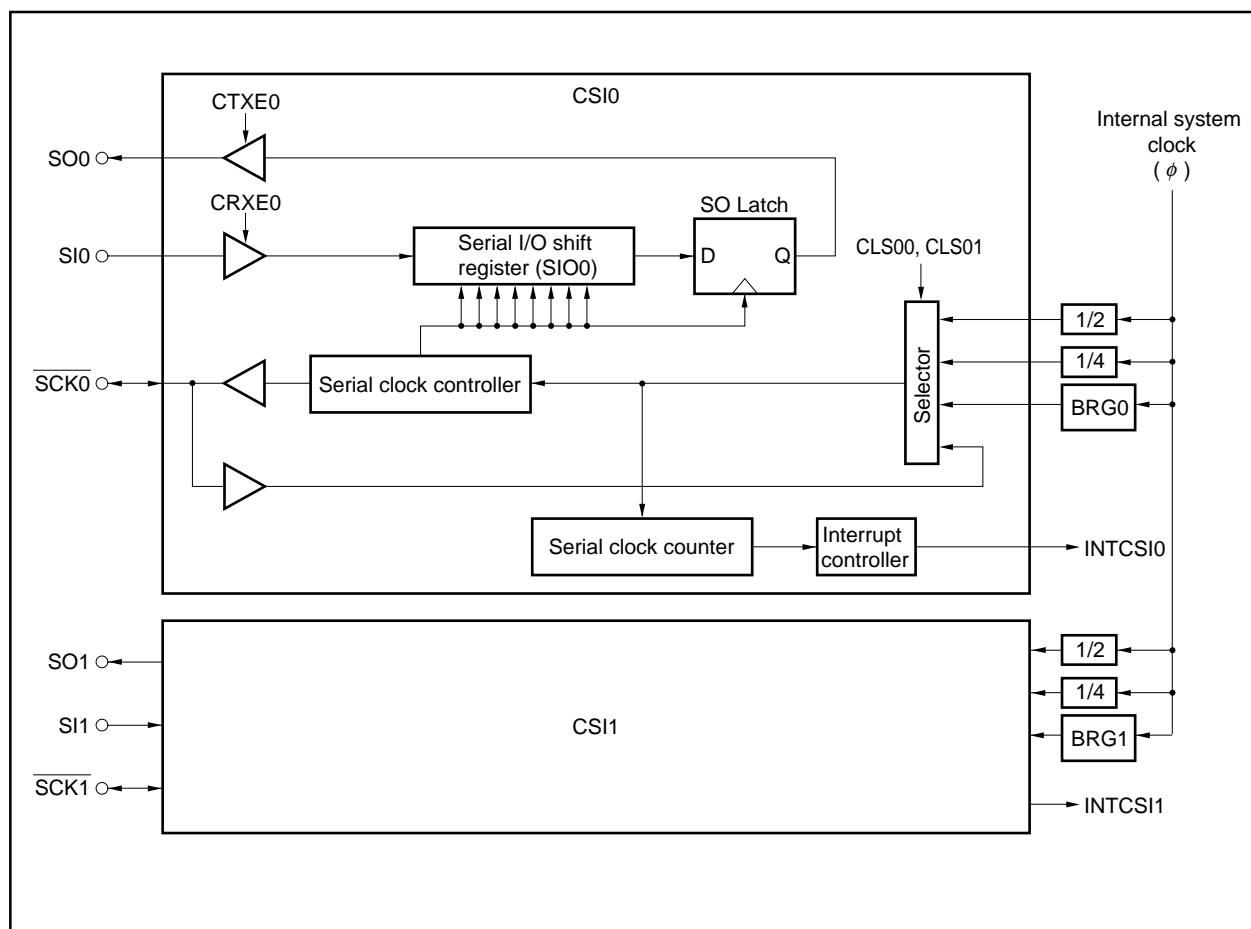
**(5) Serial clock counter**

Counts the serial clock that outputs, or is input during transmit/receive operations, and determines if 8-bit data were transmitted or received.

**(6) Interrupt control circuit**

This circuit controls whether or not an interrupt request is generated when the serial clock counter counts 8 clocks.

**Figure 10-6. Block Diagram of Clocked Serial Interface**



### 10.3.3 Control registers

#### (1) Clocked serial interface mode registers 0, 1 (CSIM0, CSIM1)

These registers specify the basic operation mode of CSI0 and CSI1.

These registers can be read/written in 8- or 1-bit units (however, for bit 5, only reading is possible).

(1/2)

	7	6	5	4	3	2	1	0		
CSIM0	CTXE0	CRXE0	CSOT0	0	0	MOD0	CLS01	CLS00	Address FFFFFF088H	After reset 00H
CSIM1	CTXE1	CRXE1	CSOT1	0	0	MOD1	CLS11	CLS10	FFFFFF098H	00H

Bit Position	Bit Name	Function
7	CTXEn	CSI Transmit Enable Specifies the transmit enabled state/disabled state. 0: Transmission disabled state 1: Transmission enabled state When CTXEn = 0, the impedance of both the SOn and SIn pins becomes high.
6	CRXEn	CSI Receive Enable Specifies the receive enabled/disabled state. 0: Reception disabled state 1: Reception enabled state When transmission is enabled (CTXEn = 1) and reception is disabled, if a serial clock is being input, 0 is input to the shift register. If reception is disabled (CRXEn = 0) while receiving data, the SIO register's contents become undefined.
5	CSOTn	CSI Status Of Transmission Shows that a transmit operation is in progress. Set (1): Transmit in progress (writing to the SIO register) Clear (0): Transmit end timing (INTCSIn generated) If set in the transmission enabled state (CTXEn = 1), when the attempt is made to start serial data transmission, this is used as a means of judging whether or not writing to serial I/O shift register n (SIO) is enabled.
2	MODn	Mode Specifies the operating mode. 0: MSB first 1: LSB first

**Remark** n = 0, 1

Bit Position	Bit Name	Function				
1, 0	CLS <sub>n</sub> 1, CLS <sub>n</sub> 0	Clock Source Specifies the serial clock.				
		CLS <sub>n</sub> 1	CLS <sub>n</sub> 0	Serial Clock Specification		SCK Pin
		0	0	External clock		Input
		0	1	Internal clock	Specified by the BPRM <sub>n</sub> register <sup>Note 1</sup>	Output
		1	0		$\phi/4$ <sup>Note 2</sup>	Output
		1	1		$\phi/2$ <sup>Note 2</sup>	Output
		<b>Notes 1.</b> Refer to <b>10.4 Dedicated Baud Rate Generators 0, 1 (BRG0, BRG1)</b> concerning setting of the BPRM <sub>n</sub> register.				
<b>2.</b> $\phi/4$ and $\phi/2$ are divider signals ( $\phi$ : Internal system clock).						

- Cautions**
1. When setting the CLS<sub>n</sub>1 and CLS<sub>n</sub>0 bits, do so in the transmission/reception disabled (CTXEn bit = CRXEn bit = 0) state. If the CLS<sub>n</sub>1 and CLS<sub>n</sub>0 bits are set in a state other than transmission/reception disabled, subsequent operation may not be normal.
  2. If the values set in bits 0 to 2 of these registers are changed while CSIn is transmitting or receiving, the operation of CSIn is not guaranteed.

**Remark** n = 0, 1

**(2) Serial I/O shift registers 0, 1 (SIO0, SIO1)**

These registers convert 8-bit serial data to 8-bit parallel data and convert 8-bit parallel data to 8-bit serial data.

The actual transmit/receive operation is controlled by reading from or writing to the SIO<sub>n</sub> registers.

Shift operation is performed when CTXEn = 1 or CRXEn = 1.

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

	7	6	5	4	3	2	1	0		
SIO0	SIO07	SIO06	SIO05	SIO04	SIO03	SIO02	SIO01	SIO00	Address FFFFFF08AH	After reset Undefined
SIO1	SIO17	SIO16	SIO15	SIO14	SIO13	SIO12	SIO11	SIO10	FFFFFF09AH	Undefined

Bit Position	Bit Name	Function
7 to 0	SIO <sub>n</sub> 7 to SIO <sub>n</sub> 0 (n = 0, 1)	Serial I/O Data shift in (receiving) or shift out (transmitting) from the MSB or from the LSB.

**Caution** CSIn operation is not guaranteed if this register is changed during CSIn operation.

## 10.3.4 Basic operation

## (1) Transfer format

CSIn transmits/receives data with three lines: one clock line and two data lines ( $n = 0, 1$ ).

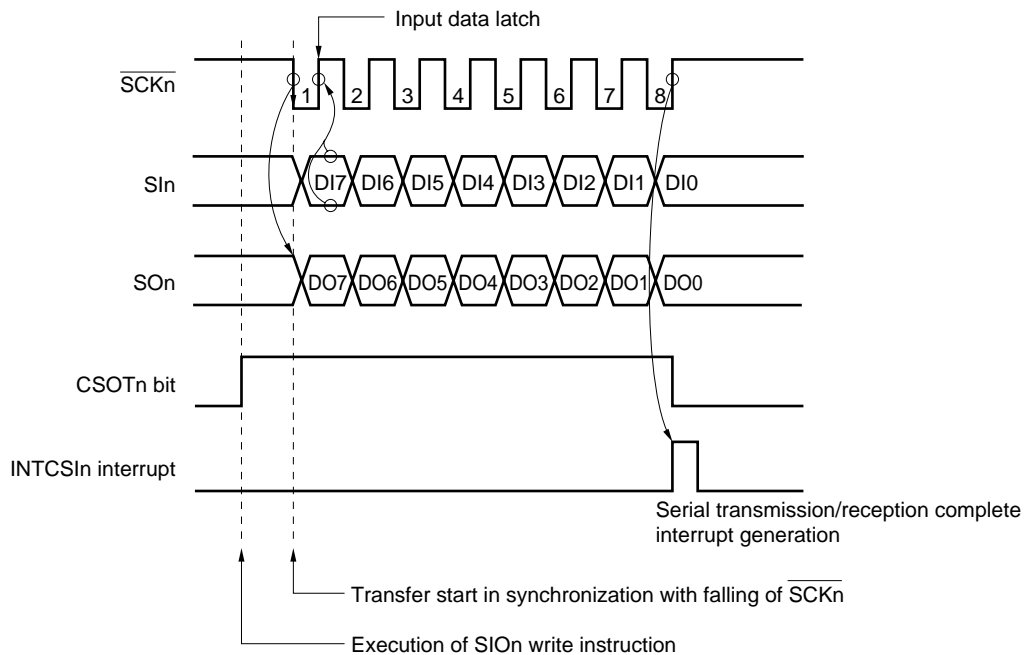
A serial transfer starts when an instruction that writes transfer data to the SIO<sub>n</sub> register is executed.

In the case of transmission, data is output from the SO<sub>n</sub> pin at each falling edge of  $\overline{\text{SCK}}_n$ .

In the case of reception, data is latched through the SI<sub>n</sub> pin at each rising edge of  $\overline{\text{SCK}}_n$ .

$\overline{\text{SCK}}_n$  stops when the serial clock counter overflows (at the rising edge of the 8th count), and  $\overline{\text{SCK}}_n$  remains high until the next data transmission or reception is started. At the same time, a transmission/reception complete interrupt (INTCSIn) is generated.

**Caution** Even if CTXEn bit is changed from 0 to 1 after the transmit data is written to the SIO<sub>n</sub>L registers, serial transfer will not begin.



**Remark**  $n = 0, 1$

**(2) Transmission/reception enabled**

CSIn each have only one 8-bit shift register and do not have any buffers, so basically, they conduct transmission and reception simultaneously ( $n = 0, 1$ ).

**(a) Transmission/reception enable conditions**

Setting of the CSIn transmission and reception enable conditions is accomplished by the CTXEn and CRXEn bits of the CSIMn registers.

However, it is necessary to set TXE0 bit = RXE0 bit = 0 in the ASIM00 register in the case of CSI0 and to set TXE1 bit = RXE1 bit = 0 in the ASIM10 register in the case of CSI1.

CTXEn	CRXEn	Transmit/Receive Operation
0	0	Transmission/reception disabled
0	1	Reception enabled
1	0	Transmission enabled
1	1	Transmission/reception enabled

**Remark**  $n = 0, 1$

**Remarks** 1. If the CTXEn bit = 0, CSIn becomes as follows.

- CSI0, CSI1: The serial output becomes high impedance or UARTn output (TXDn).
- CSI2, CSI3: The serial output becomes high impedance.

If the CTXEn bit = 1, the shift register data is output.

2. If the CRXEn bit = 0, the shift register input becomes 0.

If the CRXEn bit = 1, the serial input is input to the shift register.

3. In order to receive transmit data itself and check if a bus conflict is occurring, set CTXEn bit = CRXEn bit = 1.

**(3) Starting transmit/receive operations**

Transmit or receive operations are started by reading/writing the SIO<sub>n</sub> registers. Transmission/reception start control is carried out by setting the CTXEn and CRXEn bits of the CSIMn registers as shown below ( $n = 0, 1$ ).

CTXEn	CRXEn	Start Condition
0	0	Doesn't start
0	1	Reads the SIO <sub>n</sub> register
1	0	Writes to the SIO <sub>n</sub> register
1	1	Writes to the SIO <sub>n</sub> register
0	0 → 1	Rewrites the CRXEn bit

**Remark**  $n = 0, 1$

When the CTXEn bit is 0, the SIO<sub>n</sub> register is read/write, and even if it is set (1) afterward, transfer does not start.

Also, when the CTXEn bit is 0, if the CRXEn bit is changed from 0 to 1, the serial clock is generated and receive operation starts.

### 10.3.5 Transmission by CSIO, CSI1

After changing the settings to enable transmission by clocked serial interface mode register  $n$  (CSIM $n$ ), writing to the SIO $n$  registers starts the transmit operation ( $n = 0, 1$ ).

#### (1) Starting the transmit operation

Starting the transmit operation is accomplished by setting the CTXEn bit of clocked serial interface mode register  $n$  (CSIM $n$ ) (setting the CRXEn bit to 0), and writing transmit data to shift register  $n$  (SIO $n$ ).

Note that when the CTXEn bit = 0, the impedance of the SOn pin becomes high.

#### (2) Transmitting data in synchronization with the serial clock

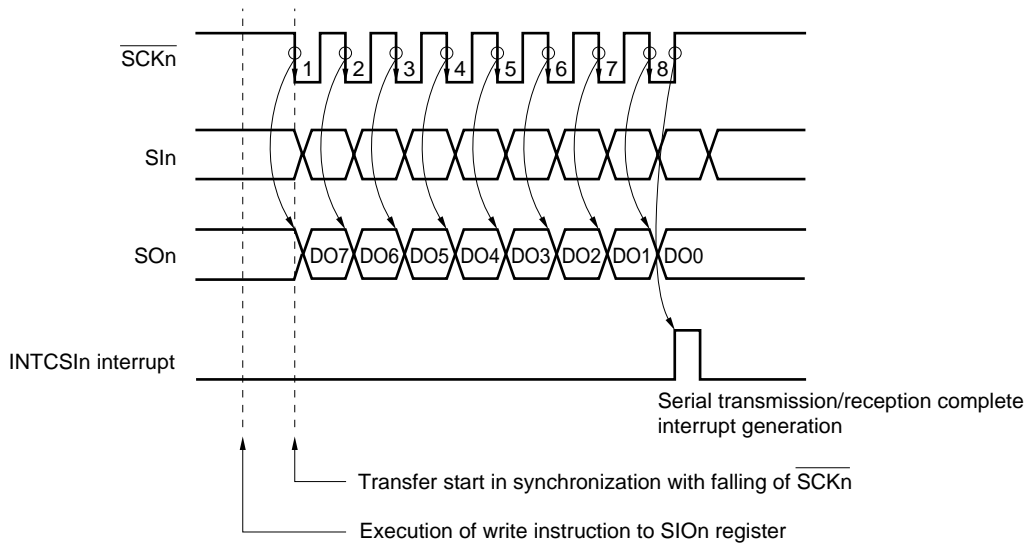
##### (a) If the internal clock is selected as the serial clock

When transmission is started, the serial clock is output from the  $\overline{\text{SCKn}}$  pin and at the same time, data from the SIO $n$  register is output sequentially to the SOn pin in synchronization with the fall of the serial clock.

##### (b) If an external clock is selected as the serial clock

When transmission is started, data from the SIO $n$  register is output sequentially to the SOn pin in synchronization with the fall of the serial clock input to the  $\overline{\text{SCKn}}$  pin after transmission starts. When transmission is not started, the shift operation is not performed even if the serial clock is input to the  $\overline{\text{SCKn}}$  pin and the SOn pin's output level does not change.

Figure 10-7. Timing of 3-Wire Serial I/O Mode (Transmission)



**Remark**  $n = 0, 1$



### 10.3.6 Reception by CSIO, CSI1

When the reception disabled setting is changed to reception enabled for clocked serial interface mode register  $n$  (CSIM $n$ ), and data is read from the SIO $n$  register in the reception enabled state, a receive operation is started ( $n = 0, 1$ ).

#### (1) Starting the receive operation

The following 2 methods can be used to start receive operations.

- <1> If the CRXEn bit of the CSIM $n$  register is changed from the reception disabled state (0) to the reception enabled state (1)
- <2> If the CRXEn bit of the CSIM $n$  register reads receive data from shift register  $n$  (SIO $n$ ) when in the reception enabled state (1)

When the CRXEn bit of the CSIM $n$  register is set (1), even if 1 is written again, a receive operation is not started. Note that when the CRXEn bit = 0, the shift register input becomes 0.

#### (2) Receiving data in synchronization with the serial clock

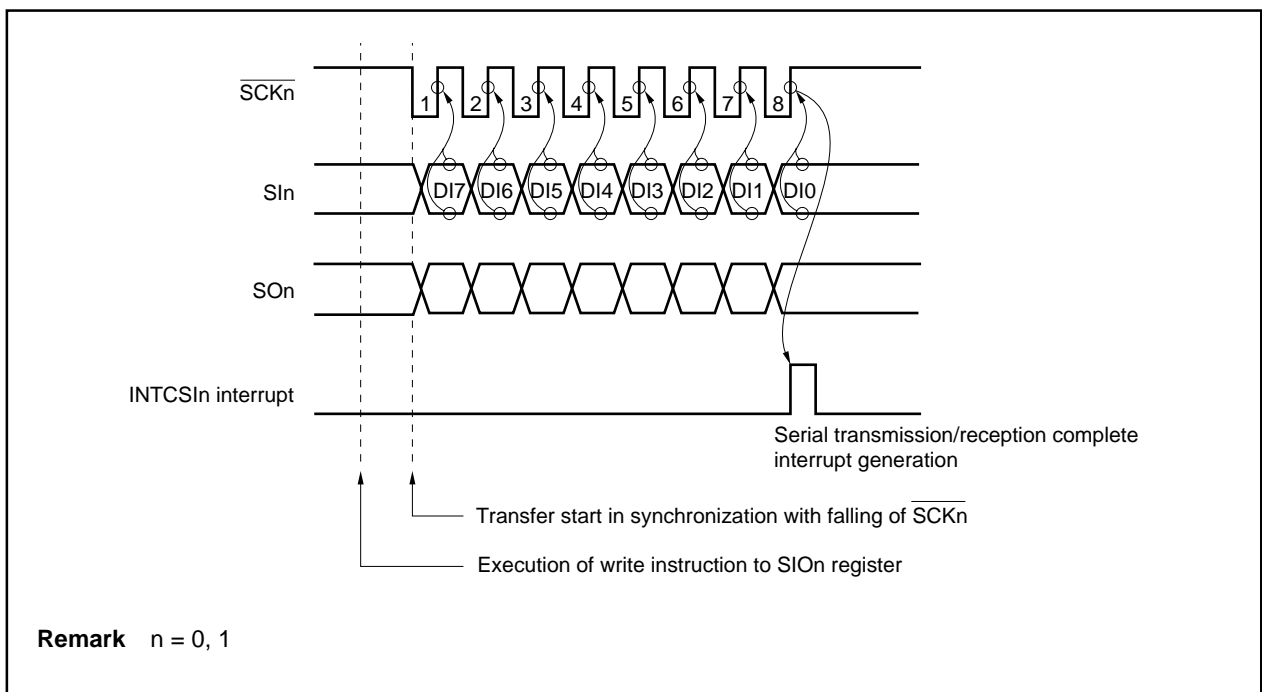
##### (a) If the internal clock is selected as the serial clock

When reception is started, the serial clock is output from the  $\overline{\text{SCKn}}$  pin and at the same time, data from the SIn pin is fetched sequentially to the SIO $n$  register in synchronization with the rise of the serial clock.

##### (b) If an external clock is selected as the serial clock

When reception is started, data from the SIn pin is fetched sequentially to the SIO $n$  register in synchronization with the rise of the serial clock input to the  $\overline{\text{SCKn}}$  pin after reception starts. When reception has not started, the shift operation is not performed even if the serial clock is input to the  $\overline{\text{SCKn}}$  pin.

Figure 10-8. Timing of 3-Wire Serial I/O Mode (Reception)



### 10.3.7 Transmission and reception by CSIO, CSI1

If both transmission and reception by clocked serial interface mode register n (CSIMn) are enabled, transmit and receive operations can be carried out simultaneously (n = 0, 1).

#### (1) Starting transmit and receive operations

When both the CTXEn bit and CRXEn bit of clocked serial interface mode register n (CSIMn) are set (1), both transmit operations and receive operations can be performed simultaneously (transmit/receive operations).

Transmit and receive operations are started when both the CTXEn and CRXEn bits of the CSIMn register are set to 1, enabling transmission and reception and when transmit data is written to shift register n (SIO<sub>n</sub>).

If the CRXEn bit of the CSIMn register is 1, even if data is written again, a transmit/receive operation is not started.

#### (2) Transmitting data in synchronization with the serial clock

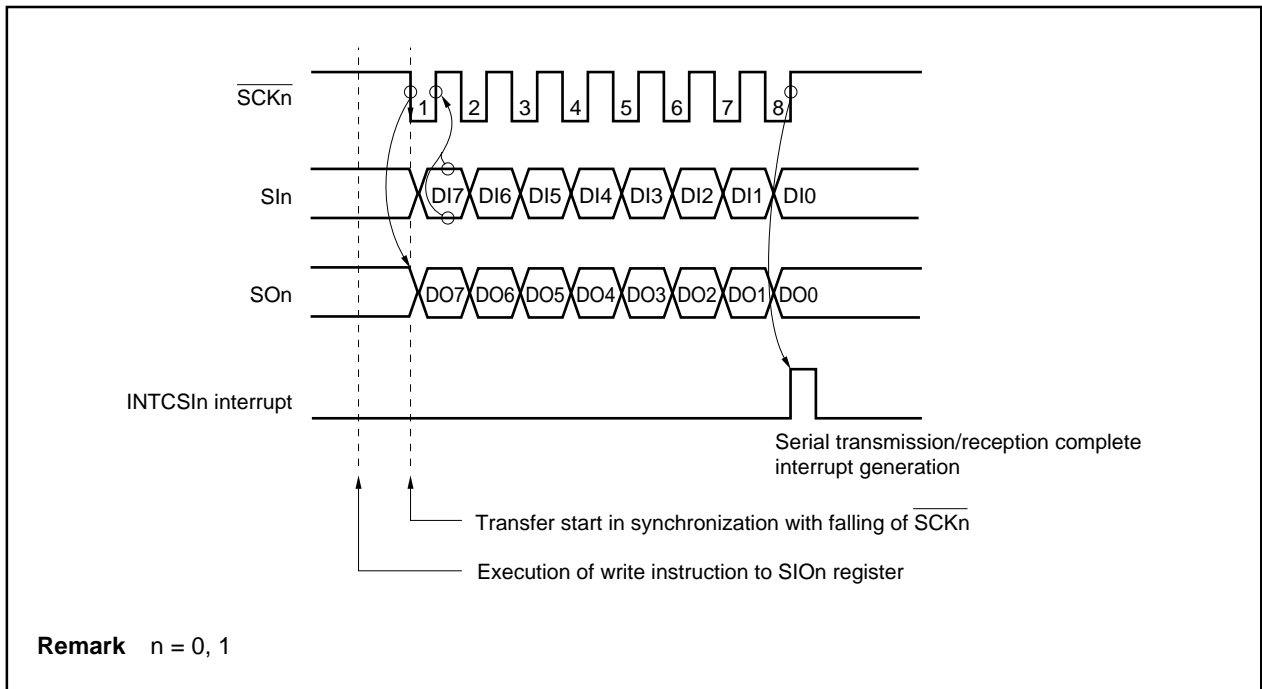
##### (a) If the internal clock is selected as the serial clock

When transmission/reception is started, the serial clock is output from the  $\overline{\text{SCKn}}$  pin and at the same time, data from the SIO<sub>n</sub> register is output sequentially to the SO<sub>n</sub> pin in synchronization with the fall of the serial clock. Also, data from the SIn pin is fetched sequentially to the SIO<sub>n</sub> register in synchronization with the rise of the serial clock.

##### (b) If an external clock is selected as the serial clock

When transmission/reception is started, data from the SIO<sub>n</sub> register is output sequentially to the SO<sub>n</sub> pin in synchronization with the fall of the serial clock input to the  $\overline{\text{SCKn}}$  pin after transmission/reception starts. Also, data from the SIn pin is fetched sequentially to the SIO<sub>n</sub> register in synchronization with the rise of the serial clock. When transmission/reception is not started, even if the serial clock is input to the  $\overline{\text{SCKn}}$  pin, shift operations are not performed and the output level of the SO<sub>n</sub> pin does not change.

Figure 10-9. Timing of 3-Wire Serial I/O Mode (Transmission/Reception)



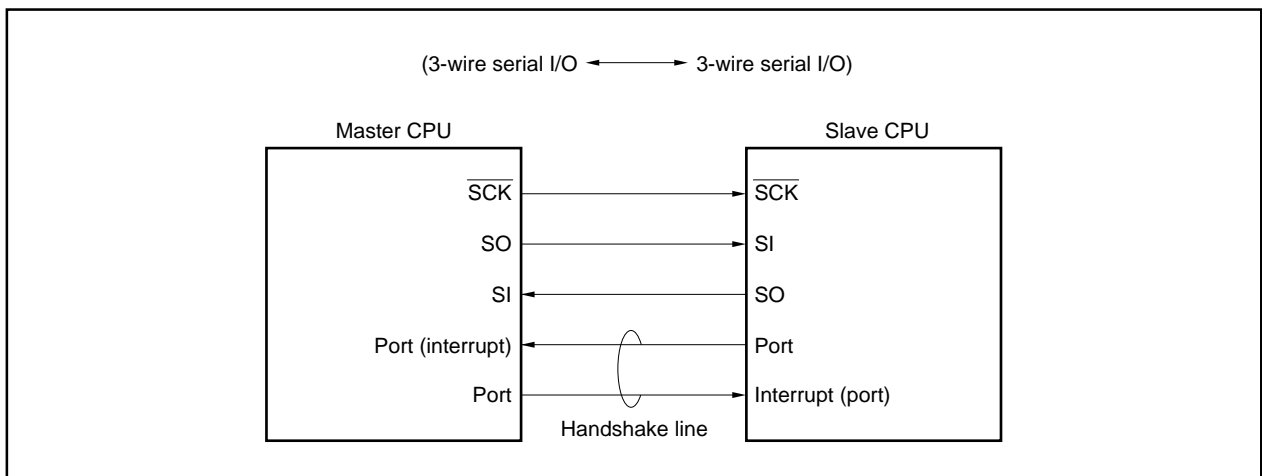
### 10.3.8 Example of system configuration

Using 3 signal lines, the serial clock ( $\overline{\text{SCKn}}$ ), serial input (SIn) and serial output (SOn), transfer of 8-bit data is carried out. This is effective in cases where connections are made to peripheral I/O with the old type of clocked serial interface built in, or with a display controller, etc. ( $n = 0, 1$ ).

If connecting to multiple devices, a line for handshake is necessary.

Since either the MSB or the LSB can be selected as the communication's header bit, it is possible to communicate with various types of device.

Figure 10-10. Example of CSI System Configuration



## 10.4 Dedicated Baud Rate Generators 0, 1 (BRG0, BRG1)

### 10.4.1 Configuration and function

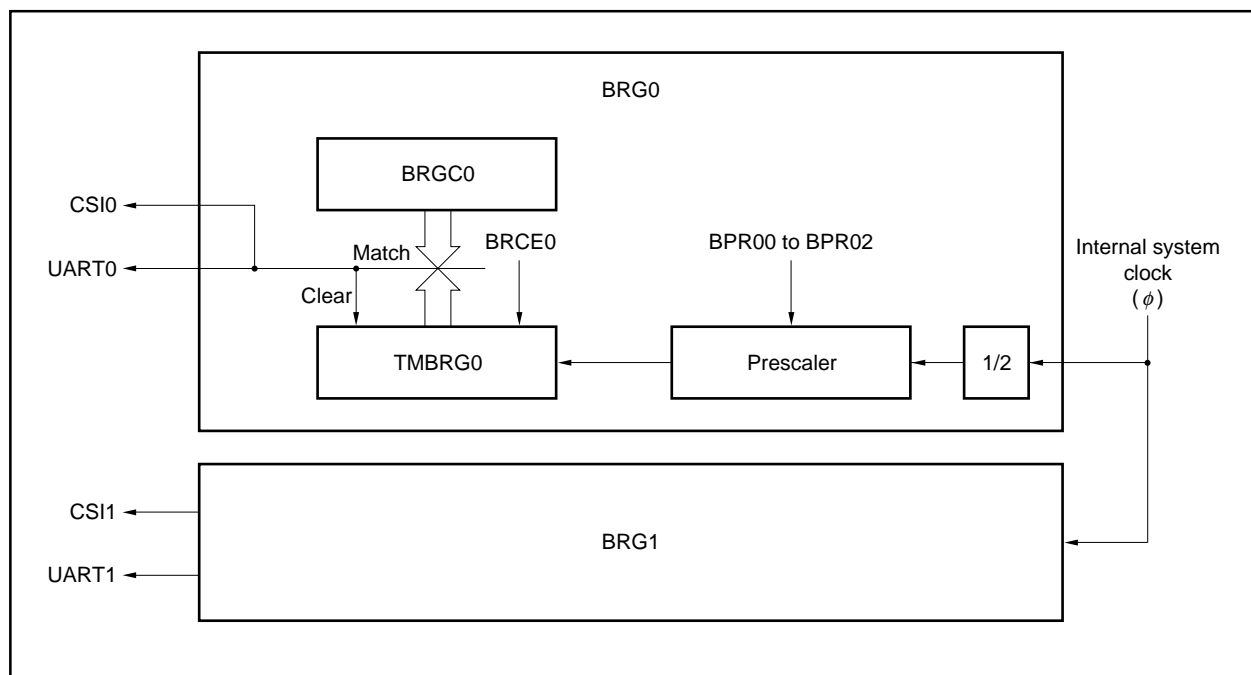
A dedicated baud rate generator output or the internal system clock ( $\phi$ ) can be selected for the serial interface serial clock for each channel.

The serial clock source is specified with the ASIM00 and ASIM10 registers for UART0 and UART1, and with the CSIM0, CSIM1 registers for CSI0, CSI1.

If the dedicated baud rate generator output is specified, BRG0 and BRG1 are selected as the clock source.

Since 1 serial clock is used in common for 1 channel of transmission and reception, the baud rate is the same for both transmission and for reception.

**Figure 10-11. Block Diagram of Dedicated Baud Rate Generator**



**(1) Dedicated baud rate generators 0, 1 (BRG0, BRG1)**

Dedicated baud rate generator BRGn (n = 0, 1) consists of a dedicated 8-bit timer (TMBRGn) which generates the transmission/reception shift clock plus a compare register (BRGCn) and prescaler.

**(a) Input clock**

Internal system clock ( $\phi$ ) is input to the BRGn.

**(b) Value set to BRGn**

**(i) UART0, UART1**

When the dedicated baud rate generator is specified as the serial source clock with the UART0, UART1, a sampling rate of  $\times 16$  is used, and therefore the baud rate is given by the following expression.

$$\text{Baud rate} = \frac{\phi}{2 \times j \times 2^k \times 16 \times 2} [\text{bps}]$$

$\phi$ : Internal system clock frequency [Hz]

j: Timer count value = BRGCn register setting value ( $1 \leq j \leq 256^{\text{Note}}$ )

k: Prescaler setting value = BPRMn register setting value (k = 0, 1, 2, 3, 4)

**Note** The j = 256 setting results in writing 0 to the BRGCn register.

**(ii) CSI0, CSI1**

If BRG0 and BRG1 are specified as the serial clock source in CSI0 and CSI1, the actual baud rate is expressed by the following formula.

$$\text{Baud rate} = \frac{\phi}{2 \times j \times 2^k \times 2} [\text{bps}]$$

$\phi$ : Internal system clock frequency [Hz]

j: Timer count value = BRGCn register setting value ( $1 \leq j \leq 256^{\text{Note}}$ )

k: Prescaler setting value = BPRMn register setting value (k = 0, 1, 2, 3, 4)

**Note** The j = 256 setting results in writing 0 to the BRGCn register.

BRGn setting values when representative clock frequencies are used are shown below.

Table 10-2. Baud Rate Generator Setup Values

Baud Rate [bps]		$\phi = 30$ MHz			$\phi = 25$ MHz			$\phi = 16$ MHz			$\phi = 12.5$ MHz		
UART0, UART1	CSI0, CSI1	BPR	BRG	Error	BPR	BRG	Error	BPR	BRG	Error	BPR	BRG	Error
110	1,760	—	—	—	4	222	0.02%	4	142	0.03%	3	222	0.02%
150	2,400	4	195	0.16%	4	163	0.15%	3	208	0.16%	3	163	0.15%
300	4,800	3	195	0.16%	3	163	0.15%	2	208	0.16%	2	163	0.15%
600	9,600	2	195	0.16%	2	163	0.15%	1	208	0.16%	1	163	0.15%
1,200	19,200	1	195	0.16%	1	163	0.15%	0	208	0.16%	0	163	0.15%
2,400	38,400	0	195	0.16%	0	163	0.15%	0	104	0.16%	0	81	0.47%
4,800	76,800	0	98	0.35%	0	81	0.47%	0	52	0.16%	0	41	0.76%
9,600	153,600	0	49	0.35%	0	41	0.76%	0	26	0.16%	0	20	1.73%
10,400	166,400	0	45	0.16%	0	38	1.16%	0	24	0.16%	0	19	1.16%
19,200	307,200	0	24	1.73 %	0	20	1.73%	0	13	0.16%	0	10	1.73%
38,400	614,400	0	12	1.73%	0	10	1.73%	0	7	6.99% <sup>Note</sup>	0	5	1.73%
76,800	1,228,800	0	6	1.73%	0	5	1.73%	—	—	—	0	3	15.2% <sup>Note</sup>
153,600	2,457,600	0	3	1.73%	0	2	27.2% <sup>Note</sup>	—	—	—	—	—	—

Baud Rate [bps]		$\phi = 20$ MHz			$\phi = 14.764$ MHz			$\phi = 12.288$ MHz		
UART0, UART1	CSI0, CSI1	BPR	BRG	Error	BPR	BRG	Error	BPR	BRG	Error
110	1,760	4	178	0.25%	4	131	0.07%	3	218	0.08%
150	2,400	4	130	0.16%	3	192	0.0%	3	160	0.0%
300	4,800	3	130	0.16%	2	192	0.0%	2	160	0.0%
600	9,600	2	130	0.16%	1	192	0.0%	1	160	0.0%
1,200	19,200	1	130	0.16%	0	192	0.0%	0	160	0.0%
2,400	38,400	0	130	0.16%	0	96	0.0%	0	80	0.0%
4,800	76,800	0	65	0.16%	0	48	0.0%	0	40	0.0%
9,600	153,600	0	33	1.36%	0	24	0.0%	0	20	0.0%
10,400	166,400	0	30	0.16%	0	22	0.7%	0	18	2.6%
19,200	307,200	0	16	1.73%	0	12	0.0%	0	10	0.0%
38,400	614,400	0	8	1.73%	0	6	0.0%	0	5	0.0%
76,800	1,228,800	0	4	1.73%	0	3	0.0%	0	3	16.7% <sup>Note</sup>
153,600	2,457,600	0	2	1.73%	0	2	25.0% <sup>Note</sup>	—	—	—

**Note** Cannot be used because the error is too great.

**Remark** BPR: Prescaler setting value (Set in the BPRMn register (n = 0, 1))

BRG: Timer count value (Set in the BRGCn register (n = 0, 1))

$\phi$ : Internal system clock frequency

**(c) Baud rate error**

The baud rate generator error 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:**  $(9,520/9,600 - 1) \times 100 = -0.833 \text{ [\%]}$   
 $(5,000/4,800 - 1) \times 100 = +4.167 \text{ [\%]}$

**(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.

**10.4.2 Baud rate generator compare registers 0, 1 (BRGC0, BRGC1)**

These are 8-bit compare registers used to set the timer count value for the BRG0 and BRG1.

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	FFFFF094H	Undefined

**Caution** Do not change the values in the BRGCn (n = 0, 1) register by software during a transmit/receive operation, because writing this register causes the internal timer (TMBRGn) to be cleared.

**10.4.3 Baud rate generator prescaler mode registers 0, 1 (BPRM0, BPRM1)**

These registers control BRG0, BRG1 timer count operations and select the count clock.

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

BPRM0	7	6	5	4	3	2	1	0	Address FFFFFF086H	After reset 00H
	BRCE0	0	0	0	0	BPR02	BPR01	BPR00		
BPRM1	7	6	5	4	3	2	1	0	Address FFFFFF096H	After reset 00H
	BRCE1	0	0	0	0	BPR12	BPR11	BPR10		

Bit Position	Bit Name	Function																								
7	BRCEn	Baud Rate Generator Count Enable Controls the BRGn count operations. 0: Stops count operations in the cleared state. 1: Enables the count operation.																								
2 to 0	BPRn2 to BPRn0	Baud Rate Generator Prescaler Specifies the count clock input to the internal timer (TMBRGn). <table border="1"><thead><tr><th>BPRn2</th><th>BPRn1</th><th>BPRn0</th><th>Count Clock</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td><td><math>\phi/2</math> (m = 0)</td></tr><tr><td>0</td><td>0</td><td>1</td><td><math>\phi/4</math> (m = 1)</td></tr><tr><td>0</td><td>1</td><td>0</td><td><math>\phi/8</math> (m = 2)</td></tr><tr><td>0</td><td>1</td><td>1</td><td><math>\phi/16</math> (m = 3)</td></tr><tr><td>1</td><td>don't care</td><td>don't care</td><td><math>\phi/32</math> (m = 4)</td></tr></tbody></table> <p>m: Prescaler setting value      <math>\phi</math>: Internal system clock frequency</p>	BPRn2	BPRn1	BPRn0	Count Clock	0	0	0	$\phi/2$ (m = 0)	0	0	1	$\phi/4$ (m = 1)	0	1	0	$\phi/8$ (m = 2)	0	1	1	$\phi/16$ (m = 3)	1	don't care	don't care	$\phi/32$ (m = 4)
BPRn2	BPRn1	BPRn0	Count Clock																							
0	0	0	$\phi/2$ (m = 0)																							
0	0	1	$\phi/4$ (m = 1)																							
0	1	0	$\phi/8$ (m = 2)																							
0	1	1	$\phi/16$ (m = 3)																							
1	don't care	don't care	$\phi/32$ (m = 4)																							

**Caution** Do not change the count clock during a transmit/receive operation.

**Remark** n = 0, 1



## CHAPTER 11 A/D CONVERTER

### 11.1 Features

- Analog input: 4 channels
- 10-bit A/D converter
- On-chip A/D conversion result register (ADCR0 to ADCR3)  
10 bits × 4
- A/D conversion trigger mode
  - A/D trigger mode
  - Timer trigger mode
- Successive approximation method

### 11.2 Configuration

The A/D converter of the V850E/MS1 adopts the successive approximation method, and uses the A/D converter mode registers (ADM0, ADM1), and ADCRn register to perform A/D conversion operations (n = 0 to 3).

#### (1) Input circuit

Selects the analog input (ANI0 to ANI3) according to the mode set to the ADM0 and ADM1 registers and sends the input to the sample and hold register.

#### (2) Sample and hold circuit

The sample and hold circuit samples each of the analog input signals sequentially sent from the input circuit, and sends the sample to the voltage comparator. This circuit also holds the sampled analog input signal voltage during A/D conversion.

#### (3) Voltage comparator

The voltage comparator compares the analog input signal with the output voltage of the series resistor string.

#### (4) Series resistor string

The series resistor string is used to generate voltages to match analog inputs.

The series resistor string is connected between the reference voltage pin ( $AV_{REF}$ ) for the A/D converter and the GND pin ( $AV_{SS}$ ) for the A/D converter. To make 1,024 equal voltage steps between these 2 pins, it is configured from 1,023 equal resistors and 2 resistors with 1/2 of the resistance value.

The voltage tap of the series resistor string is selected by a tap selector controlled by a successive approximation register (SAR).

**(5) Successive approximation register (SAR)**

The SAR is a 10-bit register in which is set series resistor string voltage tap data, which have values that match analog input voltage values, 1 bit at a time beginning with the most significant bit (MSB).

If the data is set in the SAR all the way to the least significant bit (LSB) (A/D conversion completed), the contents of that SAR (conversion results) are held in the A/D conversion results register (ADCRn).

**(6) A/D conversion results register (ADCRn)**

The ADCR is a 10-bit register that holds A/D conversion results. Each time A/D conversion is completed, conversion results are loaded from the successive approximation register (SAR).

$\overline{\text{RESET}}$  input makes its contents undefined.

**(7) Controller**

Selects the analog input, generates the sample and hold circuit operation timing, and controls the conversion trigger according to the mode set to the ADM0 and ADM1 registers.

**(8) ANI0 to ANI3 pins**

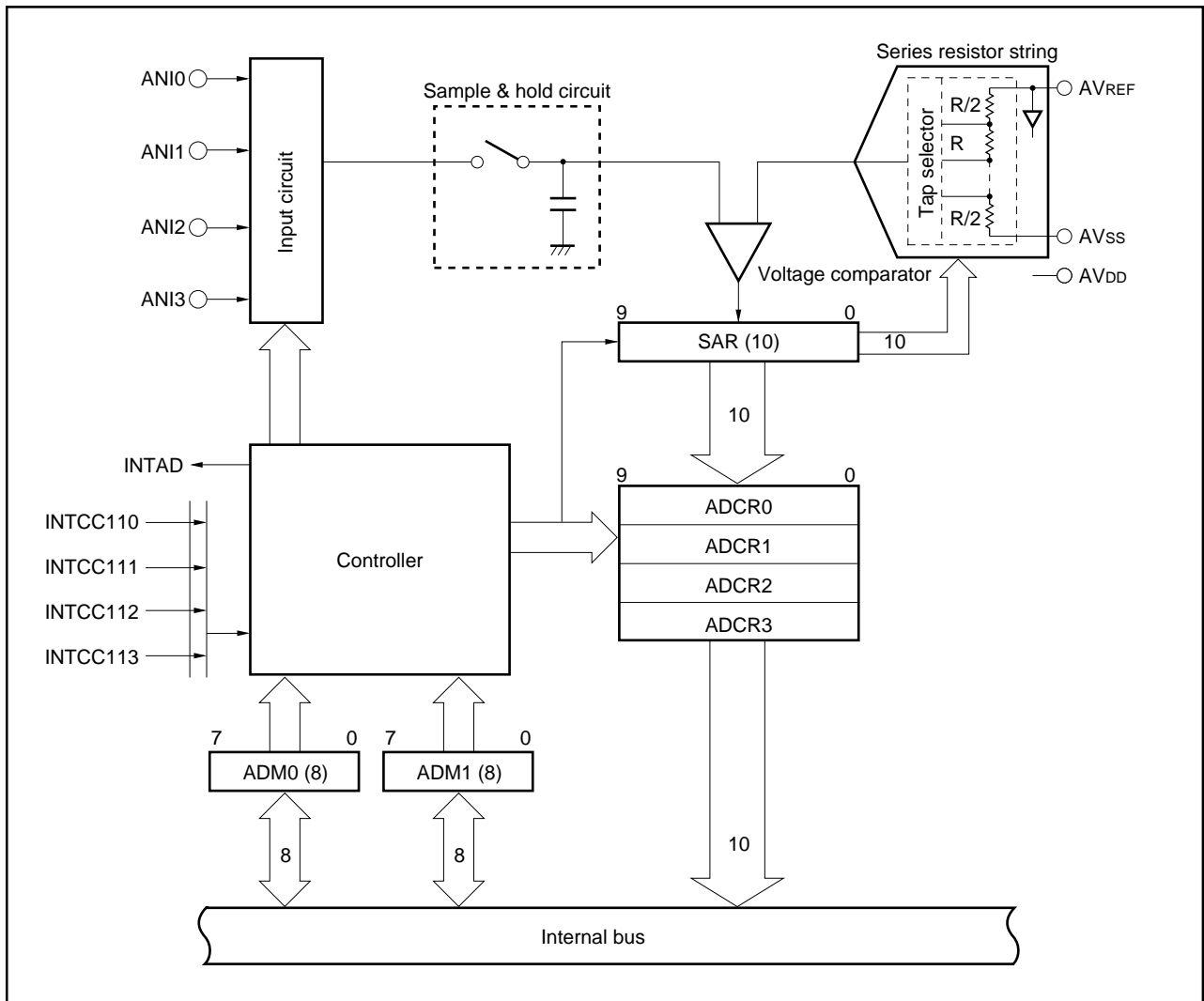
4-channel analog input pin for the A/D converter. Inputs the analog signal to be A/D converted.

**Caution** Make sure that the voltages input to ANI0 through ANI3 do not exceed the rated values. If a voltage higher than  $V_{DD}$  or lower than  $V_{SS}$  (even within the range of the absolute maximum ratings) is input to a channel, the conversion value of the channel is undefined, and the conversion values of the other channels may also be affected.

**(9)  $AV_{REF}$  pin**

Pin for inputting the reference voltage of the A/D converter. Converts signals input to the ANIn pin to digital signals based on the voltage applied between  $AV_{REF}$  and  $AV_{SS}$ .

Figure 11-1. A/D Converter Block Diagram



**Cautions** 1. When noise is generated from the analog input pins (ANI0 to ANI3) and the reference voltage input pin (AV<sub>REF</sub>), it may cause an illegal conversion result.

In order to avoid this illegal conversion result influencing the system, software processing is required.

An example of the necessary software processing is as follows.

- Use the average value of the A/D conversion results after obtaining several A/D conversion results.
- When an exceptional conversion result is obtained after performing A/D conversion several times consecutively, omit it and use the rest of the conversion results.
- When an A/D conversion result that indicates a system malfunction is obtained, be sure to recheck the abnormal generation before performing malfunction processing.

2. Make sure not to append the voltage that extends the value between AV<sub>SS</sub> to AV<sub>REF</sub> to the pins used as A/D converter input pins.

### 11.3 Control Registers

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

The ADM0 register is an 8-bit register that executes the selection of the analog input pin, specification of 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 and writing executed is ignored.

(1/2)

	7	6	5	4	3	2	1	0		
ADM0	CE	CS	BS	MS	0	0 <sup>Note 1</sup>	ANIS1	ANIS0	Address FFFFFF380H	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																																		
1, 0	ANIS1, ANIS0	Analog Input Select Specifies analog input pin to A/D convert. <table border="1"><thead><tr><th rowspan="2">ANIS1</th><th rowspan="2">ANIS0</th><th colspan="2">Select Mode</th><th colspan="2">Scan Mode</th></tr><tr><th>A/D Trigger Mode</th><th>Timer Trigger Mode</th><th>A/D Trigger Mode</th><th>Timer Trigger Mode<sup>Note 2</sup></th></tr></thead><tbody><tr><td>0</td><td>0</td><td>ANI0</td><td>ANI0</td><td>ANI0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>ANI1</td><td>ANI1</td><td>ANI0, ANI1</td><td>2</td></tr><tr><td>1</td><td>0</td><td>ANI2</td><td>ANI2</td><td>ANI0 to ANI2</td><td>3</td></tr><tr><td>1</td><td>1</td><td>ANI3</td><td>ANI3</td><td>ANI0 to ANI3</td><td>4</td></tr></tbody></table>	ANIS1	ANIS0	Select Mode		Scan Mode		A/D Trigger Mode	Timer Trigger Mode	A/D Trigger Mode	Timer Trigger Mode <sup>Note 2</sup>	0	0	ANI0	ANI0	ANI0	1	0	1	ANI1	ANI1	ANI0, ANI1	2	1	0	ANI2	ANI2	ANI0 to ANI2	3	1	1	ANI3	ANI3	ANI0 to ANI3	4
ANIS1	ANIS0	Select Mode			Scan Mode																															
		A/D Trigger Mode	Timer Trigger Mode	A/D Trigger Mode	Timer Trigger Mode <sup>Note 2</sup>																															
0	0	ANI0	ANI0	ANI0	1																															
0	1	ANI1	ANI1	ANI0, ANI1	2																															
1	0	ANI2	ANI2	ANI0 to ANI2	3																															
1	1	ANI3	ANI3	ANI0 to ANI3	4																															

**Notes** 1. Be sure to set this bit to 0.

2. In the timer trigger mode (4-trigger mode) during the scan mode, because the scanning sequence of the ANI0 to ANI3 pins is specified by the sequence in which the match signals are generated from the compare register, the number of trigger inputs should be specified instead of a certain analog input pin.

**Cautions** 1. When the CE bit is 1 in the timer 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 without clearing the CE bit, the trigger input standby state is set immediately after the change.

2. It takes 3 clocks for CS bit to become 1 after A/D conversion starts.

**(2) A/D converter mode register 1 (ADM1)**

The ADM1 register is an 8-bit register that 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 an A/D conversion operation, the conversion operation is initialized and conversion is executed from the beginning again.

	7	6	5	4	3	2	1	0		
ADM1	0	0 <sup>Note1</sup>	TRG1	TRG0	0	FR2	FR1	FR0	Address FFFFF382H	After reset 07H

Bit Position	Bit Name	Function																																																																		
5, 4	TRG1, TRG0	Trigger Mode Specifies trigger mode.																																																																		
		<table><tr><th>TRG1</th><th>TRG0</th><th>Trigger Mode</th></tr><tr><td>0</td><td>don't care</td><td>A/D trigger mode</td></tr><tr><td>1</td><td>0</td><td>Timer trigger mode (1-trigger mode)</td></tr><tr><td>1</td><td>1</td><td>Timer trigger mode (4-trigger mode)</td></tr><tr><td colspan="2">Other than above</td><td>Setting prohibited</td></tr></table>	TRG1	TRG0	Trigger Mode	0	don't care	A/D trigger mode	1	0	Timer trigger mode (1-trigger mode)	1	1	Timer trigger mode (4-trigger mode)	Other than above		Setting prohibited																																																			
		TRG1	TRG0	Trigger Mode																																																																
		0	don't care	A/D trigger mode																																																																
		1	0	Timer trigger mode (1-trigger mode)																																																																
		1	1	Timer trigger mode (4-trigger mode)																																																																
Other than above		Setting prohibited																																																																		
2 to 0	FR2 to FR0	Frequency Specifies conversion operation time. These bits control the conversion time to be same value irrespective of oscillation frequency.																																																																		
		<table><tr><th rowspan="2">FR2</th><th rowspan="2">FR1</th><th rowspan="2">FR0</th><th rowspan="2">Number of Conversion Clocks</th><th colspan="3">Conversion Operation Time (<math>\mu</math>s)<sup>Note 2</sup></th></tr><tr><th><math>\phi</math> = 30 MHz</th><th><math>\phi</math> = 25 MHz</th><th><math>\phi</math> = 16 MHz</th></tr><tr><td>0</td><td>0</td><td>0</td><td>48 clocks</td><td>—</td><td>—</td><td>—</td></tr><tr><td>0</td><td>0</td><td>1</td><td>72 clocks</td><td>—</td><td>—</td><td>—</td></tr><tr><td>0</td><td>1</td><td>0</td><td>96 clocks</td><td>—</td><td>—</td><td>6.00</td></tr><tr><td>0</td><td>1</td><td>1</td><td>120 clocks</td><td>—</td><td>—</td><td>7.50</td></tr><tr><td>1</td><td>0</td><td>0</td><td>168 clocks</td><td>5.59</td><td>6.72</td><td>—</td></tr><tr><td>1</td><td>0</td><td>1</td><td>192 clocks</td><td>6.39</td><td>7.68</td><td>—</td></tr><tr><td>1</td><td>1</td><td>0</td><td>240 clocks</td><td>7.99</td><td>9.60</td><td>—</td></tr><tr><td>1</td><td>1</td><td>1</td><td>336 clocks</td><td>—</td><td>—</td><td>—</td></tr></table>	FR2	FR1	FR0	Number of Conversion Clocks	Conversion Operation Time ( $\mu$ s) <sup>Note 2</sup>			$\phi$ = 30 MHz	$\phi$ = 25 MHz	$\phi$ = 16 MHz	0	0	0	48 clocks	—	—	—	0	0	1	72 clocks	—	—	—	0	1	0	96 clocks	—	—	6.00	0	1	1	120 clocks	—	—	7.50	1	0	0	168 clocks	5.59	6.72	—	1	0	1	192 clocks	6.39	7.68	—	1	1	0	240 clocks	7.99	9.60	—	1	1	1	336 clocks	—	—	—
		FR2					FR1	FR0	Number of Conversion Clocks	Conversion Operation Time ( $\mu$ s) <sup>Note 2</sup>																																																										
			$\phi$ = 30 MHz	$\phi$ = 25 MHz	$\phi$ = 16 MHz																																																															
		0	0	0	48 clocks	—	—	—																																																												
		0	0	1	72 clocks	—	—	—																																																												
		0	1	0	96 clocks	—	—	6.00																																																												
		0	1	1	120 clocks	—	—	7.50																																																												
		1	0	0	168 clocks	5.59	6.72	—																																																												
		1	0	1	192 clocks	6.39	7.68	—																																																												
		1	1	0	240 clocks	7.99	9.60	—																																																												
		1	1	1	336 clocks	—	—	—																																																												
		<b>Remark</b> $\phi$ : Internal system clock frequency —: Setting prohibited																																																																		

**Notes** 1. Be sure to set this bit to 0.

2. Figures under Conversion Operation Time are target values.

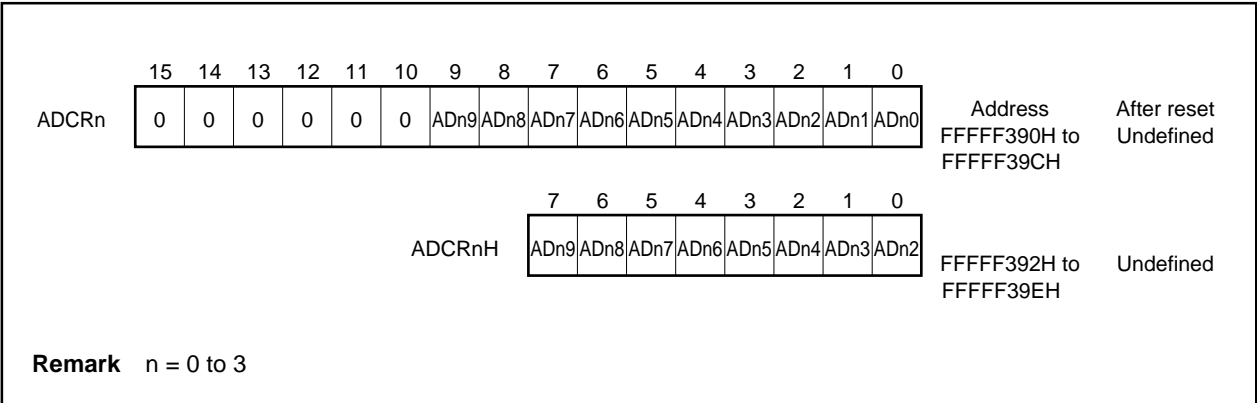
(3) A/D conversion result registers (ADCR0 to ADCR3, ADCR0H to ADCR3H)

The ADCRn register is a 10-bit register holding the A/D conversion results. It is provided with four 10-bit registers (n = 0 to 3).

This register is read-only, in 16- or 8-bit units.

During 16-bit access to this register, the ADCRn register is specified, and during higher 8-bit access, the ADCRnH register is specified.

When reading the 10-bit data of A/D conversion results from the ADCRn register, only the lower 10 bits are valid and the higher 6 bits are always read as 0.



The correspondence between each analog input pin and the ADCRn register (except the 4-buffer mode) is shown below.

Analog Input Pin	ADCRn Register
ANI0	ADCR0, ADCR0H
ANI1	ADCR1, ADCR1H
ANI2	ADCR2, ADCR2H
ANI3	ADCR3, ADCR3H





## 11.4 A/D Converter Operation

### 11.4.1 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, trigger mode, etc. should be set in the ADM0 and ADM1 registers<sup>Note 1</sup>.  
When the CE bit of the ADM0 register is set (1), A/D conversion starts in the A/D trigger mode. In the timer trigger mode, the trigger standby state<sup>Note 2</sup> is set.
- (2) The voltage generated from the voltage tap of the series resistor string and analog input are compared by the comparator.
- (3) When the comparison of the 10 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 3).

- Notes**
1. When the ADM0 and ADM1 registers are changed during an A/D conversion operation, the A/D conversion operation before the change is stopped and the conversion results are not stored in the ADCRn register.
  2. In the timer 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.

### 11.4.2 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 ADM0 and ADM1 registers.

The following shows the relationship between the operation mode and trigger mode.

Trigger Mode		Operation Mode		Setting Value		Analog Input
				ADM0 register	ADM1 register	
A/D trigger		Select	1 buffer	xx0100xxB	000x0xxxB	ANI0 to ANI3
			4 buffers	xx1100xxB	000x0xxxB	
		Scan		xxx000xxB	000x0xxxB	
Timer trigger	1 trigger	Select	1 buffer	xx0100xxB	00100xxxB	
			4 buffers	xx1100xxB	00100xxxB	
		Scan		xxx000xxB	00100xxxB	
	4 triggers	Select	1 buffer	xx0100xxB	00110xxxB	
			4 buffers	xx1100xxB	00110xxxB	
		Scan		xxx000xxB	00110xxxB	

#### (1) Trigger mode

There are two types of trigger modes that serve as the start timing of the A/D conversion processing: A/D trigger mode and timer trigger mode. The timer trigger mode consists of the 1-trigger mode and 4-trigger mode as the sub-trigger mode. These trigger modes are set by the ADM1 register.

##### (a) A/D trigger mode

Generates the conversion timing of the analog input for the ANI0 to ANI3 pins inside the A/D converter unit.

##### (b) Timer trigger mode

Specifies the conversion timing of the analog input set for the ANI0 to ANI3 pins using the values set to the TM11 compare register.

This register creates the analog input conversion timing by generating the match interrupts of the four capture/compare registers (CC110 to CC113) connected to the 16-bit TM11.

There are two types of sub-trigger modes: 1-trigger mode and 4-trigger mode.

- **1-trigger mode**

Mode that uses one match interrupt from timer 11 as the A/D conversion start timing.

- **4-trigger mode**

Mode that uses four match interrupts from timer 11 as the A/D conversion start timing.

**(2) Operation mode**

There are two types of operation modes that set the ANI0 to ANI3 pins: select mode and scan mode. The select mode has sub-modes including the 1-buffer mode and 4-buffer mode. These modes are set by the ADM0 register.

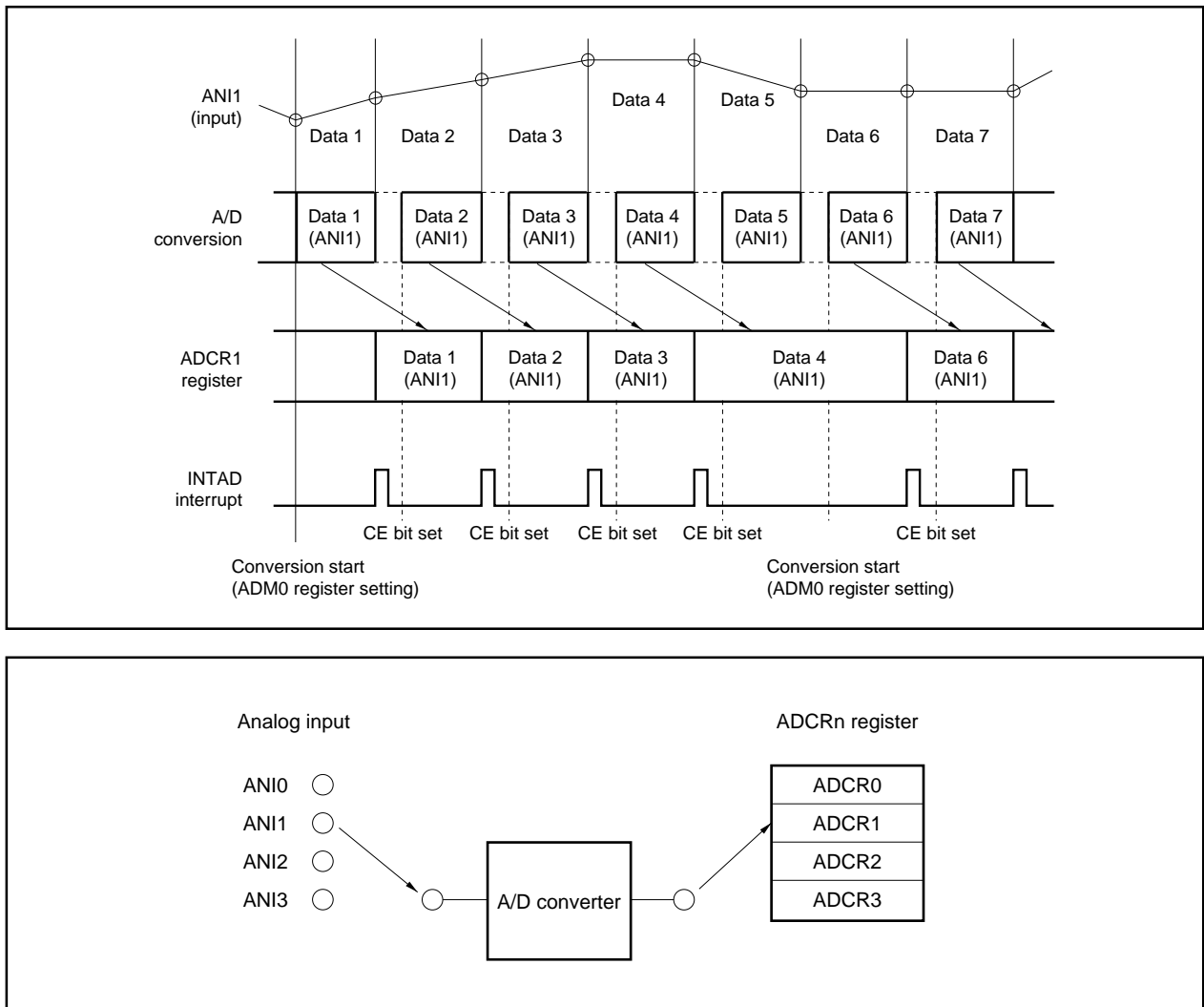
**(a) Select mode**

One analog input specified by the ADM0 register is A/D converted. The conversion results are stored in the ADCRn register corresponding to the analog input (ANIn). For this mode, the 1-buffer mode and 4-buffer mode are provided for storing the A/D conversion results ( $n = 0$  to 3).

- 1-buffer mode**

One analog input specified by the ADM0 register is A/D converted. The conversion results are stored in the ADCRn register corresponding to the analog input (ANIn). The ANIn and ADCRn registers correspond one to one, and an A/D conversion end interrupt (INTAD) is generated each time one A/D conversion ends.

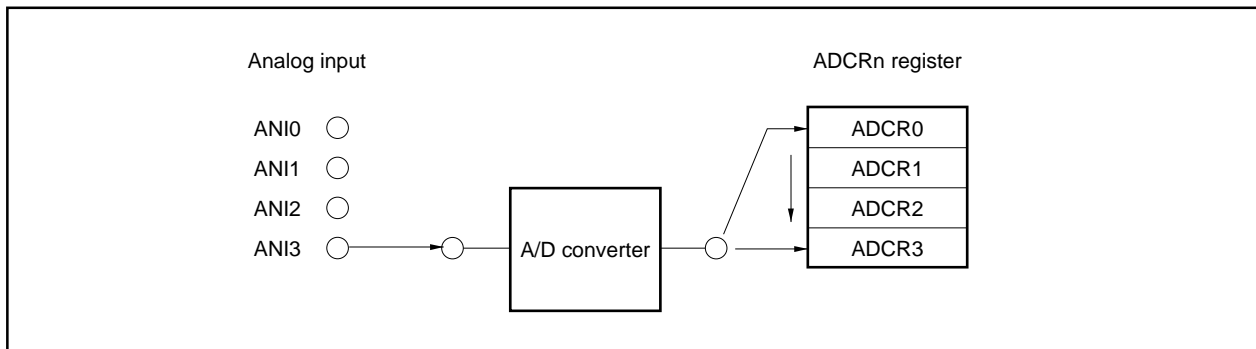
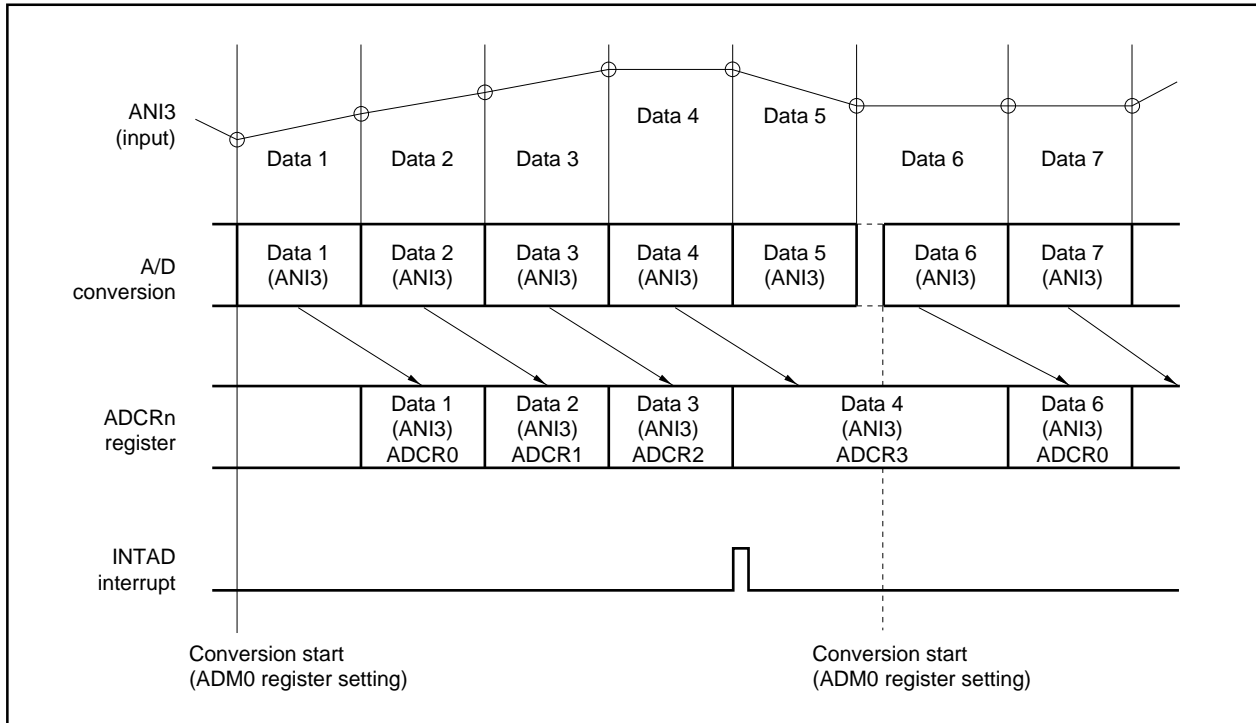
**Figure 11-3. Select Mode Operation Timing: 1-Buffer Mode (ANI1)**



- **4-buffer mode**

One analog input is A/D converted four times and the results are stored in the ADCR0 to ADCR3 registers. The A/D conversion end interrupt (INTAD) is generated when the four A/D conversions end.

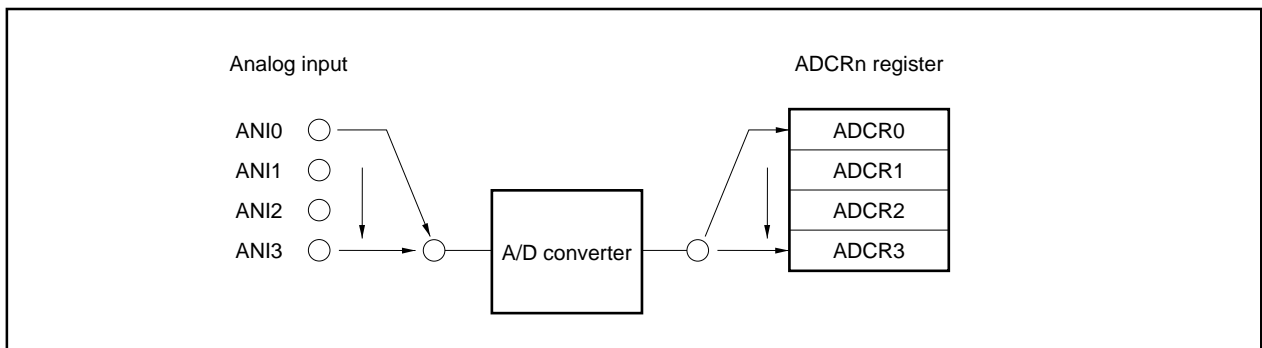
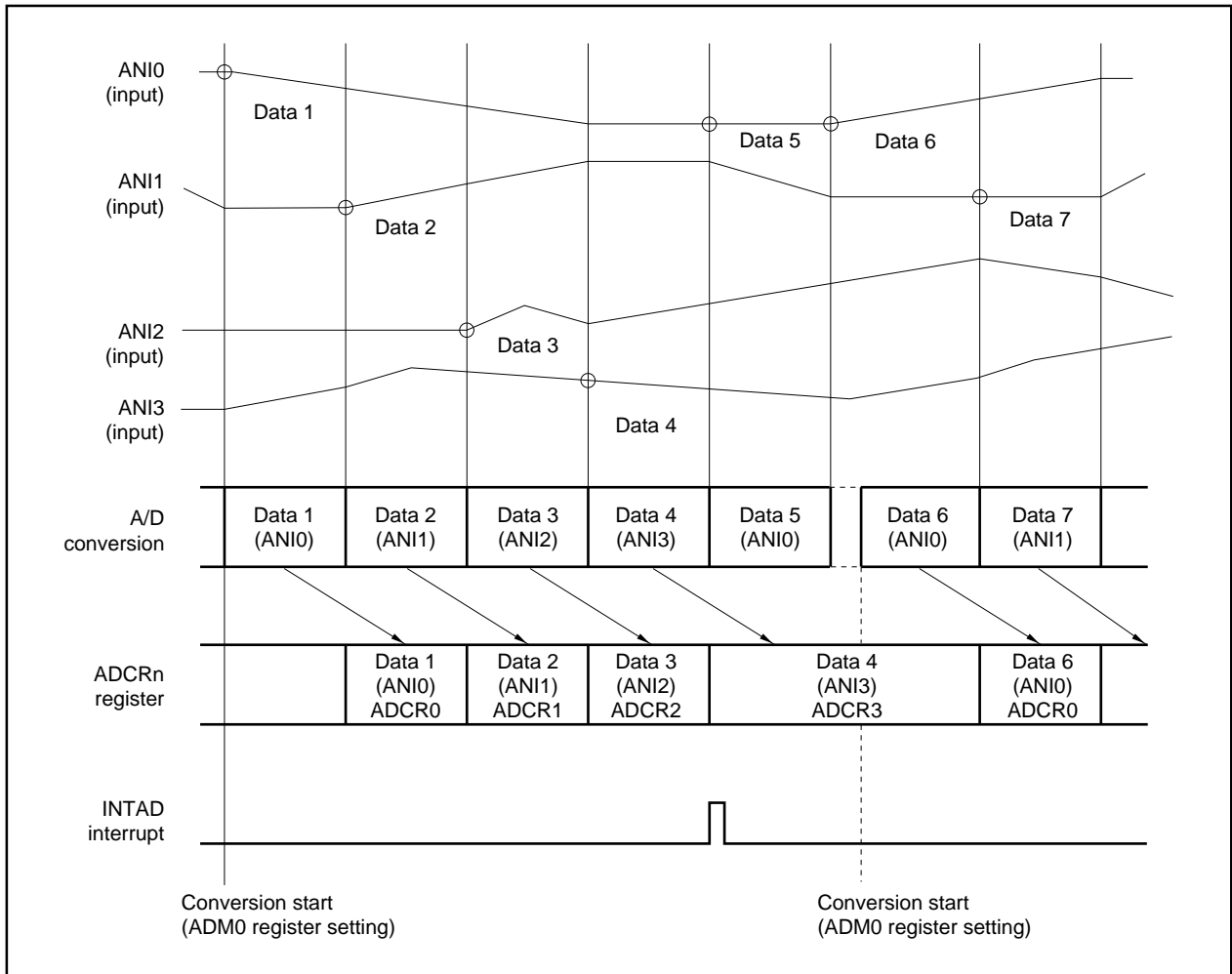
**Figure 11-4. Select Mode Operation Timing: 4-Buffer Mode (ANI3)**



**(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 ( $n = 0$  to 3). When the conversion of the specified analog input ends, the INTAD interrupt is generated.

**Figure 11-5. Scan Mode Operation Timing: 4-Channel Scan (ANI0 to ANI3)**



## 11.5 Operation in A/D Trigger Mode

When the CE bit of the ADM0 register is set to 1, A/D conversion starts.

### 11.5.1 Select mode operations

The analog input specified by the ADM0 register is A/D converted. The conversion results are stored in the ADCRn register corresponding to the analog input. For the select mode, the 1-buffer mode and 4-buffer mode are supported according to the storing method of the A/D conversion results ( $n = 0$  to 3).

#### (1) 1-buffer mode (A/D trigger select: 1-buffer)

One analog input is A/D converted once. The conversion results are stored in one ADCRn register. The analog input and ADCRn register correspond one to one.

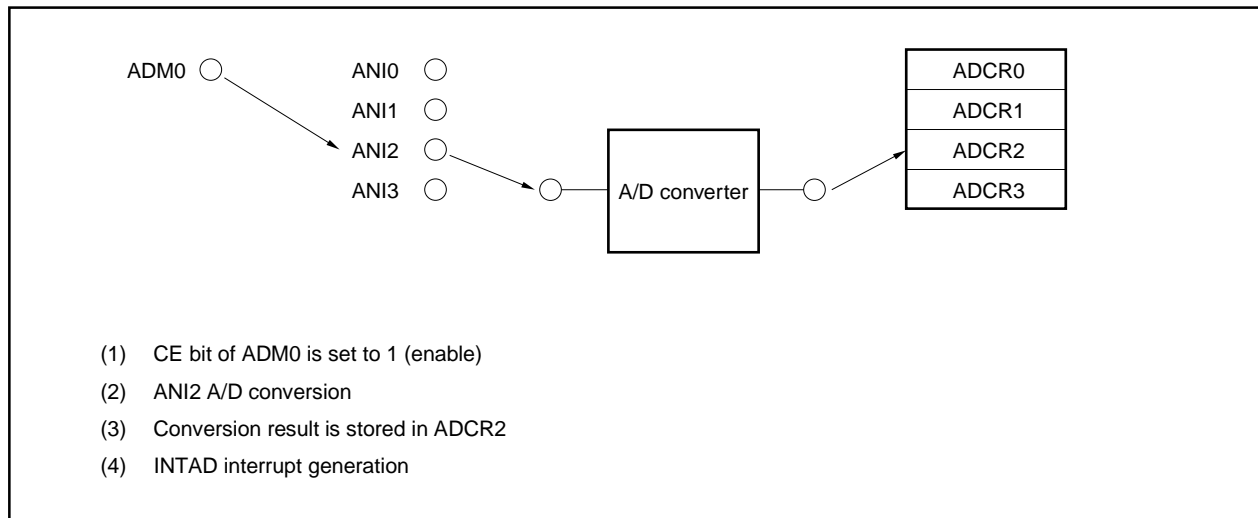
Each time an A/D conversion is executed, an INTAD interrupt is generated and the AD conversion terminates.

Analog Input	A/D Conversion Results Register
ANIn	ADCRn

( $n = 0$  to 3)

If 1 is written to the CE bit of the ADM0 register, A/D conversion can be restarted. This is most appropriate for applications in which the results of each first time A/D conversion are read.

**Figure 11-6. Example of 1-Buffer Mode (A/D Trigger Select 1-Buffer) Operation**



**(2) 4-buffer mode (A/D trigger select: 4-buffer)**

One analog input is A/D converted four times and the results are stored in the four ADCR0 to ADCR3 registers. When four A/D conversions end, an INTAD interrupt is generated and A/D conversion terminates.

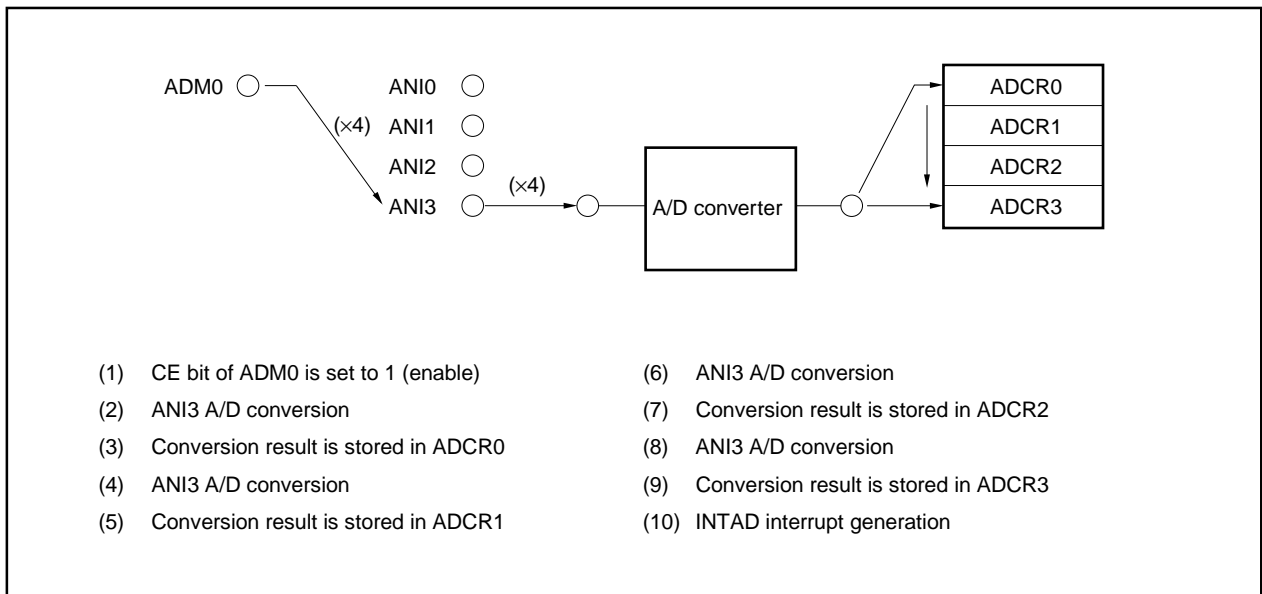
Analog Input	A/D Conversion Result Register
ANIn	ADCR0
ANIn	ADCR1
ANIn	ADCR2
ANIn	ADCR3

(n = 0 to 3)

If 1 is written in the CE bit of the ADM0 register, A/D conversion can be restarted.

This is most appropriate for applications that determine the average A/D conversion results.

**Figure 11-7. Example of 4-Buffer Mode (A/D Trigger Select 4-Buffer) Operation**



### 11.5.2 Scan mode operations

The analog inputs specified by the ADM0 register are selected 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 (n = 0 to 3).

When the conversion of all the specified analog input ends, the INTAD interrupt is generated, and A/D conversion terminates.

Analog Input	A/D Conversion Result Register
ANIn	ADCR0
ANIn <sup>Note</sup>	ADCRn

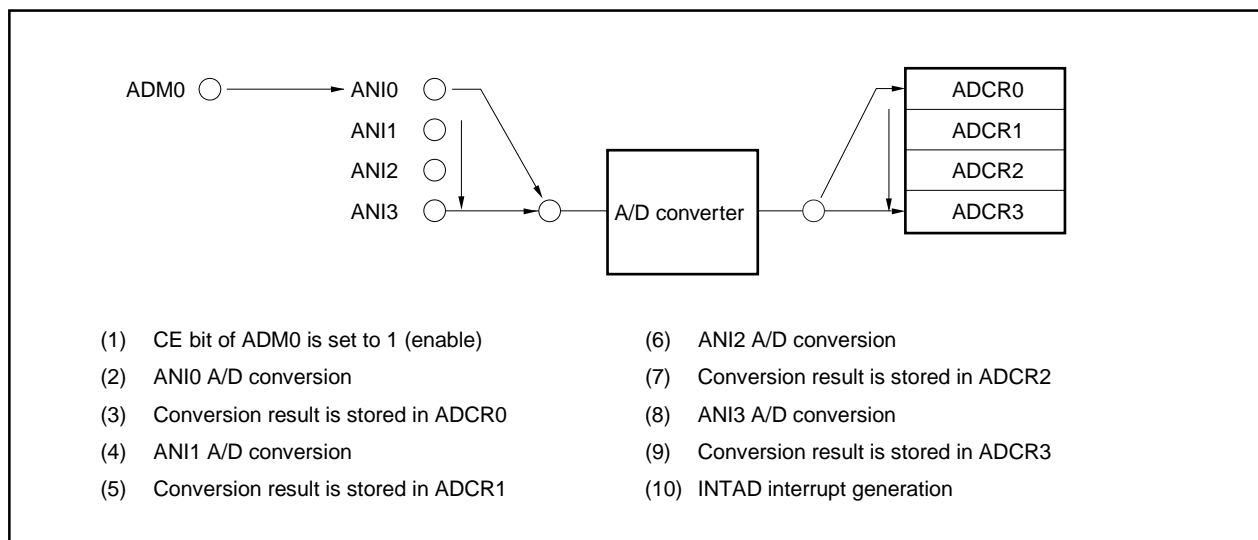
(n = 0 to 3)

**Note** Set in the ANIS0 to ANIS2 bits of the ADM0 register.

If 1 is written in the CE bit of the ADM0 register, A/D conversion can be restarted.

This is most appropriate for applications that are constantly monitoring multiple analog inputs.

**Figure 11-8. Example of Scan Mode (A/D Trigger Scan) Operation**





## 11.6 Operation in Timer Trigger Mode

The A/D converter is the match interrupt signal of the TM11 compare register, and can set conversion timings to a maximum of four channel analog inputs (ANI0 to ANI3).

TM11 and four capture/compare registers (CC110 to CC113) are used for the timer for specifying the analog conversion trigger.

The following two modes are provided according to the value set in the TUM11 register.

### (1) 1-shot mode

To use the 1-shot mode, the OST bit of the TUM11 register should be set to 1 (1-shot mode).

When the A/D conversion period is longer than the TM11 period, the TM11 generates an overflow, holds 0000H, and stops. Thereafter, TM11 does not output the match interrupt signal (A/D conversion trigger) of the compare register, and the A/D converter also enters the A/D conversion standby state. The TM11 count operation restarts when the valid edge of the TCLR11 pin input is detected or when 1 is written to the CE11 bit of the TMC11 register.

### (2) Loop mode

To use the loop mode, the OST bit of the TUM11 register should be set to 0 (normal mode).

When the TM11 generates an overflow, the TM11 starts counting from 0000H again, and the match interrupt signal (A/D conversion trigger) of the compare register is repeatedly output and A/D conversion is also repeated.

### 11.6.1 Select mode operations

One analog input (ANI0 to ANI3) specified by the ADM0 register is A/D converted. The conversion results are stored in the ADCRn register corresponding to the analog input. For the select mode, the 1-buffer mode and 4-buffer mode are provided according to the storing method of the A/D conversion results ( $n = 0$  to 3).

#### (1) 1-buffer mode operations (Timer trigger select: 1-buffer)

One analog input is A/D converted once and the conversion results are stored in one ADCRn register.

There are two modes in 1-buffer modes, the 1-trigger mode and 4-trigger mode, according to the number of triggers.

##### (a) 1-trigger mode (Timer trigger select: 1-buffer, 1-trigger)

One analog input is A/D converted once using the trigger of the match interrupt signal (INTCC110) and the results are stored in one ADCRn register.

An INTAD interrupt is generated for each A/D conversion and A/D conversion terminates.

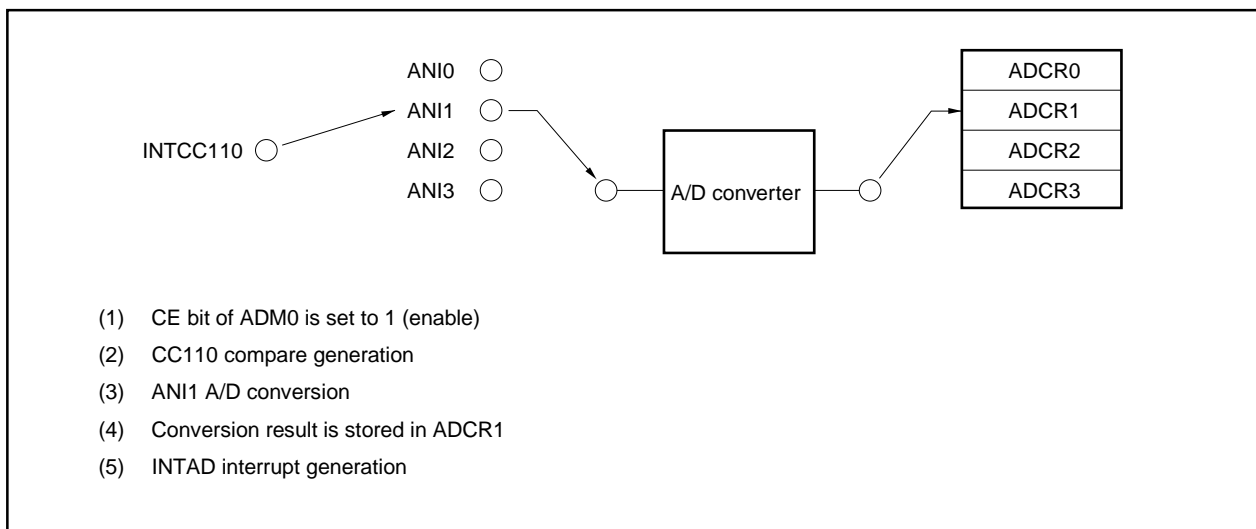
Trigger	Analog Input	A/D Conversion Result Register
INTCC110 interrupt	ANIn	ADCRn

( $n = 0$  to 3)

When the TM11 is set to the 1-shot mode, A/D conversion ends after one conversion. To restart A/D conversion, input the valid edge to the TCLR11 pin or write 1 to the CE11 bit of the TMC11 register.

When set to the loop mode, unless the CE bit of the ADM0 register is set to 0, A/D conversion is repeated each time the match interrupt is generated.

**Figure 11-9. Example of 1-Trigger Mode (Timer Trigger Select 1-Buffer 1-Trigger) Operation**



**(b) 4-trigger mode (Timer trigger select: 1-buffer, 4-trigger)**

One analog input is A/D converted four times using four match interrupt signals (INTCC110 to INTCC113) as triggers and the results are stored in one ADCRn register. The INTAD interrupt is generated with each A/D conversion, and the CS bit of the ADM0 register is reset (0). The results of one A/D conversion are held by the ADCRn register until the next A/D conversion ends. Perform transmission of the conversion results to the memory and other operations using the INTAD interrupt after each A/D conversion ends.

Trigger	Analog Input	A/D Conversion Result Register
INTCC110 interrupt	ANIn	ADCRn
INTCC111 interrupt	ANIn	ADCRn
INTCC112 interrupt	ANIn	ADCRn
INTCC113 interrupt	ANIn	ADCRn

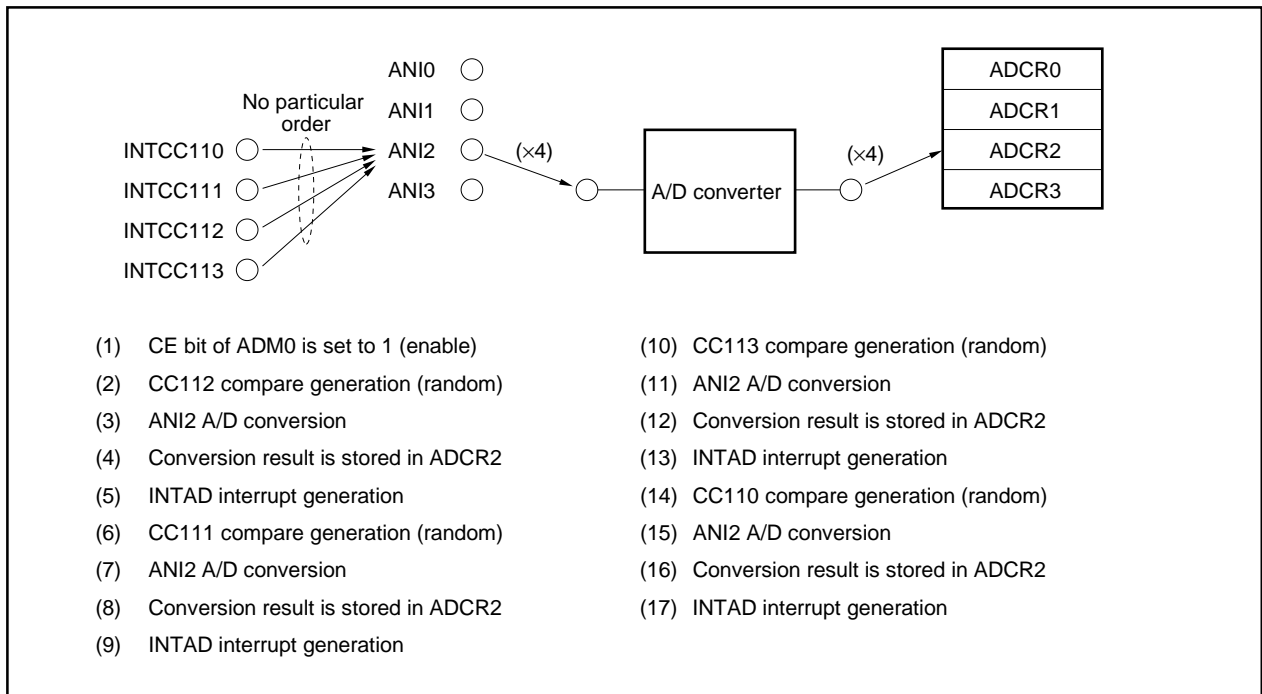
(n = 0 to 3)

When the TM11 is set to the 1-shot mode, A/D conversion ends after four conversions. To restart A/D conversion, input the valid edge to the TCLR11 pin or write 1 to the CE11 bit of the TMC11 register to restart the TM11. When the first match interrupt after TM11 is restarted is generated, the CS bit is set (1) and A/D conversion is started.

When set to the loop mode, unless the CE bit of the ADM0 register is set to 0, A/D conversion is repeated each time the match interrupt is generated.

The match interrupts (INTCC110 to INTCC113) can be generated in any order. The same trigger, even when it enters several times consecutively, is accepted as a trigger each time.

**Figure 11-10. Example of 4-Trigger Mode (Timer Trigger Select 1-Buffer 4-Trigger) Operation**



**(2) 4-buffer mode operations (Timer trigger select: 4-buffer)**

One analog input is A/D converted four times, and the results are stored in the ADCR0 to ADCR3 registers. There are two 4-buffer modes, 1-trigger mode and 4-trigger mode, according to the number of triggers. This mode is suitable for applications that calculate the average of the A/D conversion result.

**(a) 1-trigger mode**

One analog input is A/D converted four times using the match interrupt signal (INTCC110) as a trigger, and the results are stored in the ADCR0 to ADCR3 registers.

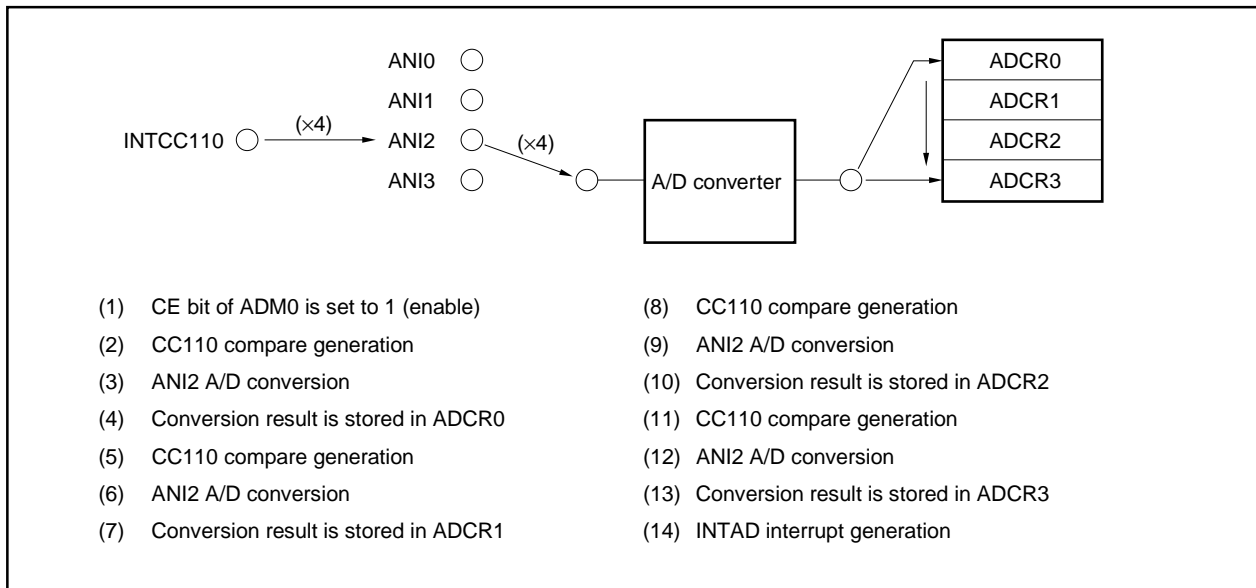
An INTAD interrupt is generated when the four A/D conversions end and A/D conversion terminates.

Trigger	Analog Input	A/D Conversion Result Register
INTCC110 interrupt	ANIn	ADCR0
INTCC110 interrupt	ANIn	ADCR1
INTCC110 interrupt	ANIn	ADCR2
INTCC110 interrupt	ANIn	ADCR3

(n = 0 to 3)

When the TM11 is set to the 1-shot mode, and less than four match interrupts are generated, if the CE bit is set to 0, the INTAD interrupt is not generated and the standby state is set.

**Figure 11-11. Example of 1-Trigger Mode (Timer Trigger Select 4-Buffer 1-Trigger) Operation**



**(b) 4-trigger mode**

One analog input is A/D converted four times using four match interrupt signals (INTCC110 to INTCC113) as triggers and the results are stored in four ADCRn registers. The INTAD interrupt is generated when the four A/D conversions end, the CS bit is reset (0), and A/D conversion terminates.

Trigger	Analog Input	A/D Conversion Result Register
INTCC110 interrupt	ANIn	ADCR0
INTCC111 interrupt	ANIn	ADCR1
INTCC112 interrupt	ANIn	ADCR2
INTCC113 interrupt	ANIn	ADCR3

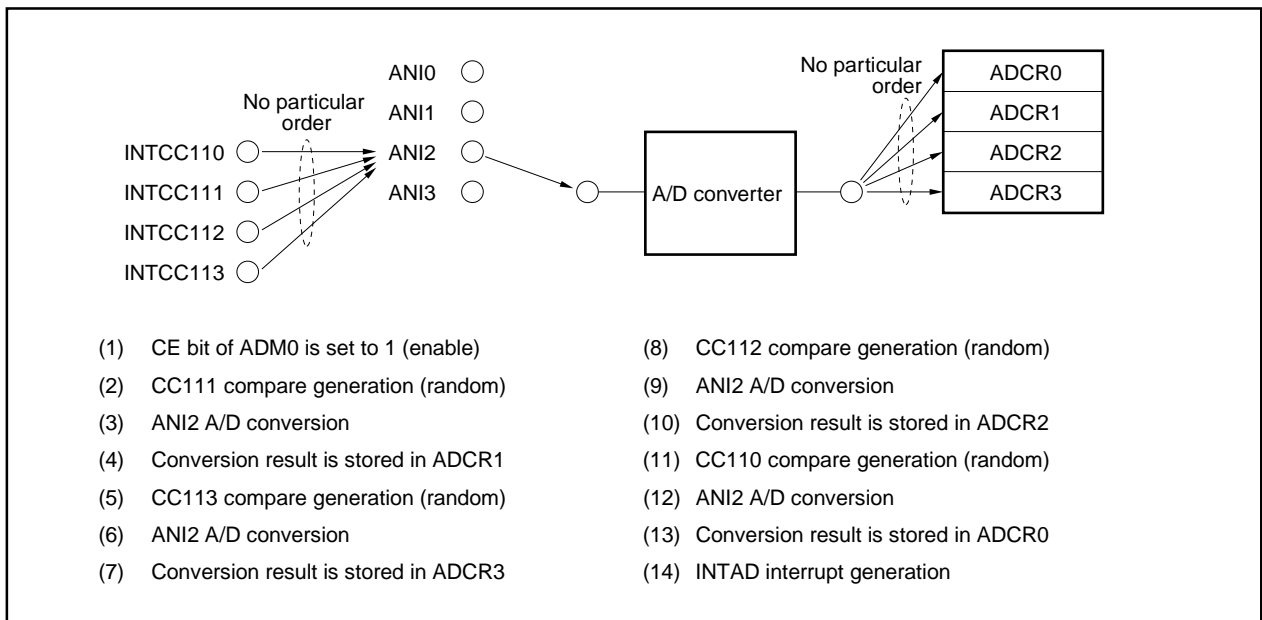
(n = 0 to 3)

When the TM11 is set to the 1-shot mode, A/D conversion ends after four conversions. To restart A/D conversion, input the valid edge to the TCLR11 pin or write 1 to the CE11 bit of the TMC11 register to restart the TM11. When the first match interrupt after TM11 is restarted is generated, the CS bit is set (1) and A/D conversion is started.

When set to the loop mode, unless the CE bit is set to 0, A/D conversion is repeated each time the match interrupt is generated.

Whichever the order of occurrence of match interrupts (INTCC110 to INTCC113), there is no problem, and the conversion results are stored in the ADCRn register corresponding to the input trigger. Also, even in cases where the same trigger is input continuously, it is received as a trigger.

**Figure 11-12. Example of 4-Trigger Mode (Timer Trigger Select 4-Buffer 4-Trigger) Operation**



### 11.6.2 Scan mode operations

The analog inputs specified by the ADM0 register are selected sequentially from the ANI0 pin and A/D converted for the specified number of times using the match interrupt signal as a trigger.

When the set number of A/D conversions ends, the INTAD interrupt is generated and A/D conversion ends.

There are two scan modes, 1-trigger mode and 4-trigger mode, according to the number of triggers.

This is most appropriate for applications that are constantly monitoring multiple analog inputs.

#### (1) 1-trigger mode (Timer trigger scan: 1-trigger)

The analog inputs are A/D converted for the specified number of times using the match interrupt signal (INTCC110) as a trigger.

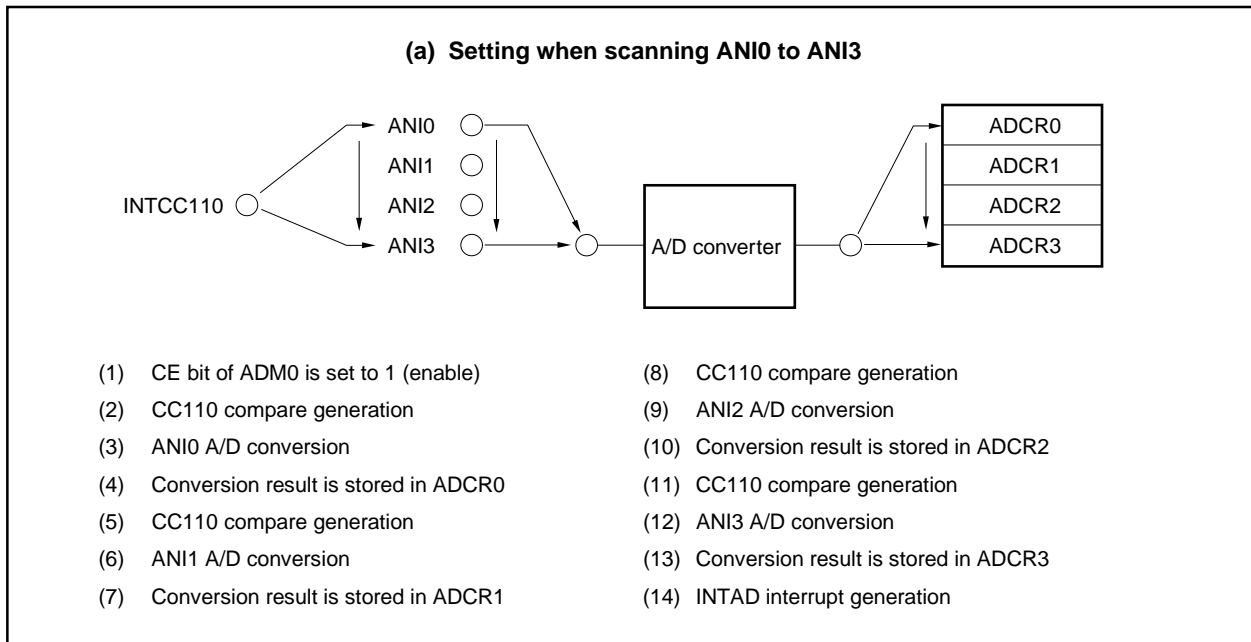
The analog input and ADCRn register correspond one to one.

When all the A/D conversions specified have ended, the INTAD interrupt is generated and A/D conversion ends.

Trigger	Analog Input	A/D Conversion Result Register
INTCC110 interrupt	ANI0	ADCR0
INTCC110 interrupt	ANI1	ADCR1
INTCC110 interrupt	ANI2	ADCR2
INTCC110 interrupt	ANI3	ADCR3

When the match interrupt is generated after all the specified A/D conversions end, A/D conversion is restarted. When the TM11 is set to the 1-shot mode, and less than a specified number of match interrupts are generated, if the CE bit is set to 0, the INTAD interrupt is not generated and the standby state is set.

**Figure 11-13. Example of 1-Trigger Mode (Timer Trigger Scan 1-Trigger) Operation**



**(2) 4-trigger mode**

The analog inputs are A/D converted for the number of times specified using the match interrupt signal (INTCC110 to INTCC113) as a trigger.

The analog input and ADCRn register correspond one to one.

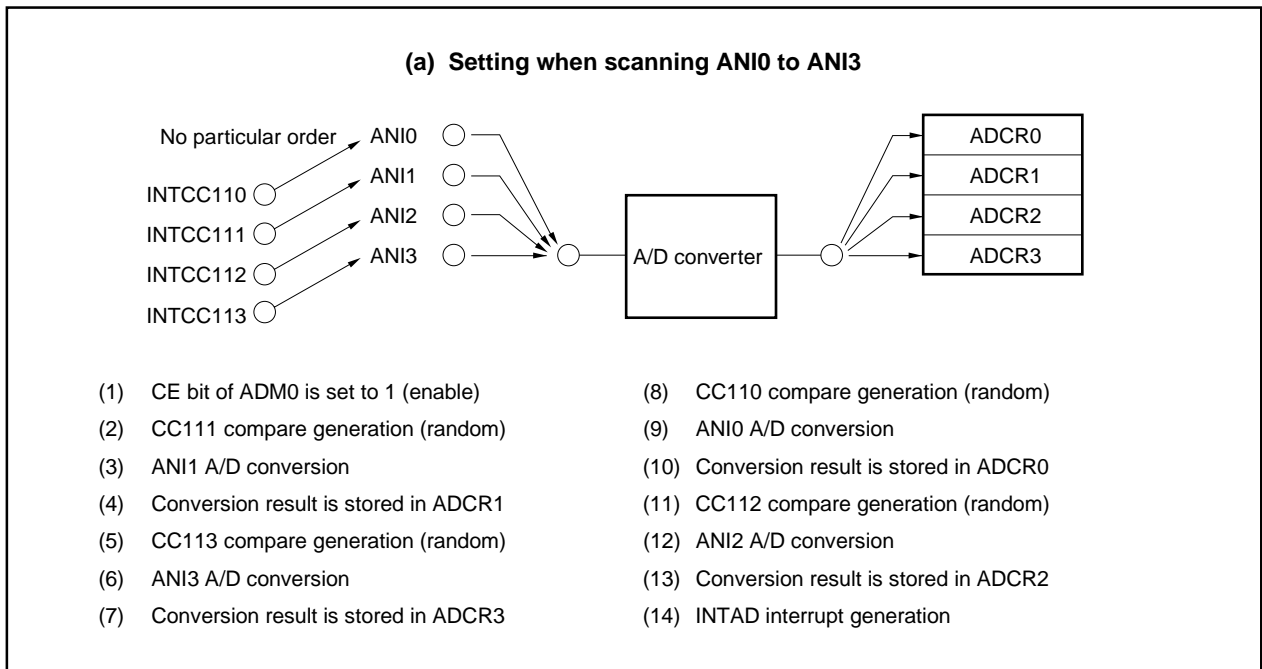
When all the A/D conversions specified have ended, the INTAD interrupt is generated and A/D conversion ends.

Trigger	Analog Input	A/D Conversion Result Register
INTCC110 interrupt	ANI0	ADCR0
INTCC111 interrupt	ANI1	ADCR1
INTCC112 interrupt	ANI2	ADCR2
INTCC113 interrupt	ANI3	ADCR3

To restart conversion when TM11 is set to the 1-shot mode, restart TM11. If set to the loop mode and the CE bit is 1, A/D conversion is restarted when a match interrupt is generated after conversion ends.

The match interrupt can be generated in any order. However, because the trigger signal and the analog input correspond one to one, the scanning sequence is determined according to the order in which the match signals of the compare register are generated.

**Figure 11-14. Example of 4-Trigger Mode (Timer Trigger Scan 4-Trigger) Operation**



## 11.7 Operating Precautions

### 11.7.1 Stopping conversion operation

When 0 is written to the CE bit of the ADM0 register during a conversion operation, the conversion operation stops and the conversion results are not stored in the ADCRn register ( $n = 0$  to 3).

### 11.7.2 Timer trigger interval

Set the interval (input time interval) of the trigger in the timer trigger mode longer than the conversion time specified by the FR2 to FR0 bits of the ADM1 register.

#### (1) When interval = 0

When several triggers are input simultaneously, the analog input with the smaller ANIn pin number is converted. The other trigger signals input simultaneously are ignored, and the number of trigger inputs is not counted. Therefore, the generation of interrupts and storage of results in the ADCRn register will become abnormal ( $n = 0$  to 3).

#### (2) When $0 < \text{interval} \leq \text{conversion operation time}$

When the timer trigger is input during a conversion operation, the conversion operation stops and the conversion starts according to the last timer trigger input.

When a conversion operation stops, the conversion results are not stored in the ADCRn register. However, the number of trigger inputs is counted, and when the interrupt is generated, the value at which conversion ended is stored in the ADCRn register.

### 11.7.3 Operation of standby mode

#### (1) HALT mode

The A/D conversion operation continues. When released by the NMI input, the ADM0 and ADM1 registers and ADCRn register hold the value ( $n = 0$  to 3).

#### (2) IDLE mode, STOP mode

As clock supply to the A/D converter is stopped, no conversion operations are performed. When these modes are released using the NMI input, the ADM0 and ADM1 registers and the ADCRn register hold the value. However, when the IDLE and software STOP modes are set during a conversion operation, the conversion operation stops. At this time, if released using the NMI input, the conversion operation resumes, but the conversion result written to the ADCRn register will become undefined.

In the IDLE and software STOP modes, operation of the comparator is also stopped to reduce the power consumption, and to further reduce current consumption, set the voltage of the AVREF to VSS.

### 11.7.4 Compare match interrupt when in timer trigger mode

The compare register's match interrupt becomes an A/D conversion start trigger and starts the conversion operation. When this happens, the compare register's match interrupt functions even if it is a compare register match interrupt directed to the CPU. In order to prevent match interrupts from the compare register being directed to the CPU, disable interrupts by the interrupt mask bits (P11MK0 to P11MK3) of the interrupt control register (P11IC0 to P11IC3).



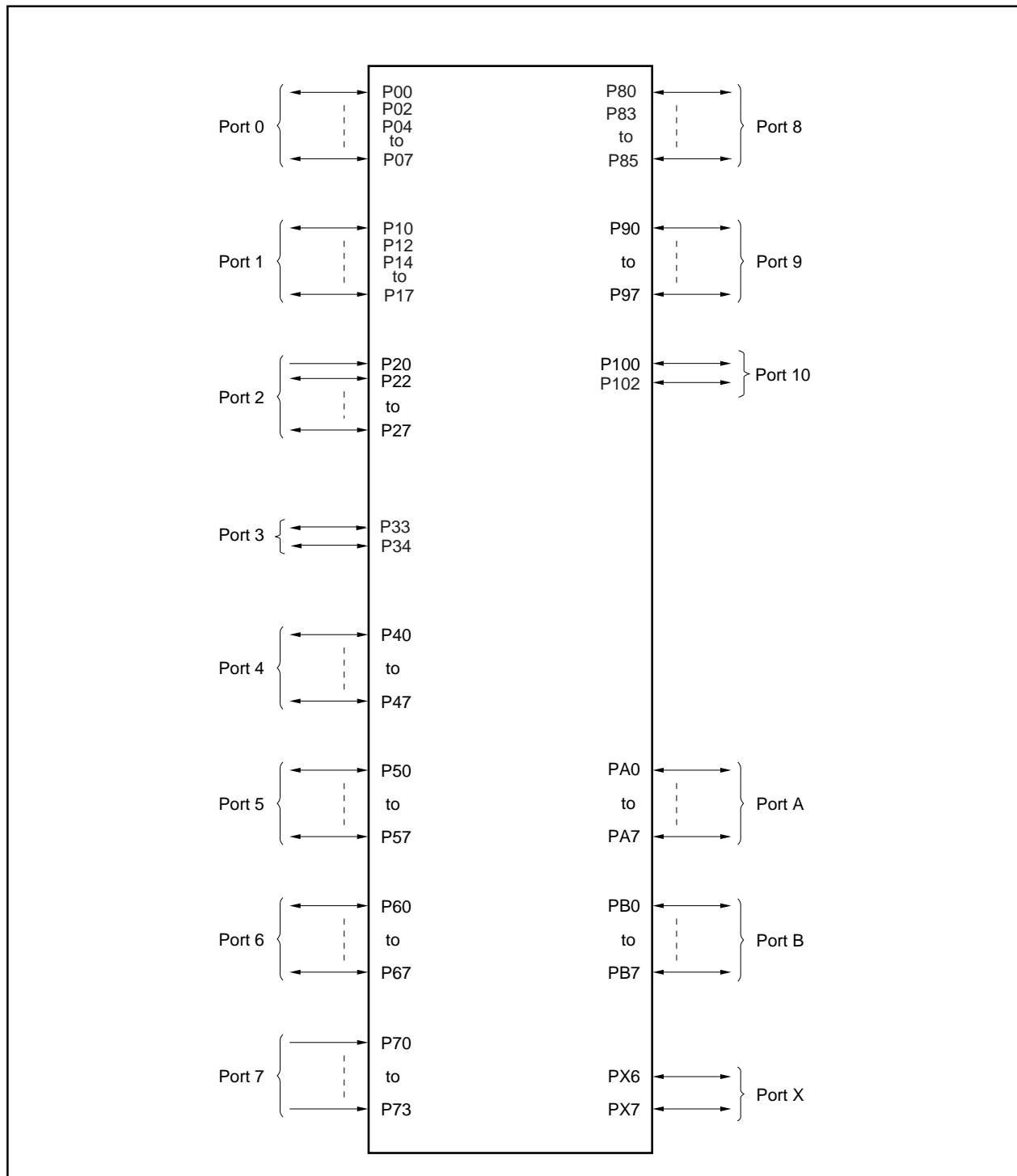
## CHAPTER 12 PORT FUNCTIONS

## 12.1 Features

- Number of ports    Input-only ports     5  
                             I/O ports              76
- Function alternately as the I/O pins of other peripheral functions.
- It is possible to specify input and output in bit units.

## 12.2 Port Configuration

The V850E/MS2 incorporates a total of 123 I/O ports (including 9 input-only ports) named ports 0 through 12, and A, B and X. The port configuration is shown below.



**(1) Function of each port**

The port functions of the V850E/MS2 are shown below.

8/1-bit operations are possible on all ports, allowing various kinds of control to be performed. In addition to their port functions, these pins also function as internal peripheral I/O input/output pins in the control mode.

Port Name	Pin Name	Port Function	Function in Control Mode	Block Type <sup>Note</sup>
Port 0	P00, P04 to P07	6-bit I/O	Input/output of real-time pulse unit (RPU) External interrupt input DMA control (DMAC) input	A, B, M
Port 1	P10, P12, P14 to P17	6-bit I/O	Input/output of real-time pulse unit (RPU) External interrupt input DMA control (DMAC) output	A, B, K
Port 2	P20, P22 to P27	1-bit input, 6-bit I/O	NMI input Serial interface (UART0/CSI0, UART1/CSI1) input/output	C, D, I, Q
Port 3	P33, P34	2-bit I/O	Input of real-time pulse unit (RPU) External interrupt input	B
Port 4	P40 to P47	8-bit I/O	External data bus (D0 to D7)	E
Port 5	P50 to P57	8-bit I/O	External data bus (D8 to D15)	E
Port 6	P60 to P67	8-bit I/O	External address bus (A16 to A23)	F
Port 7	P70 to P73	4-bit input	A/D converter (ADC) analog input	G
Port 8	P80, P83 to P85	4-bit I/O	External bus interface control signal output	O, P
Port 9	P90 to P97	8-bit I/O	External bus interface control signal input/output	H, O
Port 10	P100, P102	2-bit I/O	Input/output of real-time pulse unit (RPU)	A, B
Port A	PA0 to PA7	8-bit I/O	External address bus (A0 to A7)	F
Port B	PB0 to PB7	8-bit I/O	External address bus (A8 to A15)	F
Port X	PX6, PX7	2-bit I/O	Wait insertion signal input Internal system clock output	A, D

**Note** Refer to 12.2 (3) Block diagram of port.

**Caution** When switching to the control mode, be sure to set ports that operate as output pins, or as input/output pins in the control mode, by the following procedure.

- <1> Set the inactive level for the signal output in the control mode in the relevant bits of port n (Pn) (n = 0 to 6, 8 to 10, A, B, X).
- <2> Switch to the control mode from the port n mode control register (PMCn).

If <1> above is not performed, when switching from the port mode to the control mode, the contents of port n (Pn) will be output instantaneously.

(2) Function when each port's pins are reset and register which sets the port/control mode

(1/2)

Port Name	Pin Name	Pin Function After Reset		Register Which Sets the Mode
		ROM-less Mode 0	ROM-less Mode 1	
Port 0	P00/TO100	P00 (input mode)		PMC0
	P02/TCLR10	P02 (input mode)		
	P04/INTP100/DMARQ0	P04 (input mode)		PMC0, PCS0 <sup>Note</sup>
	P05/INTP101/DMARQ1	P05 (input mode)		
	P06/INTP102/DMARQ2	P06 (input mode)		
	P07/INTP103/DMARQ3	P07 (input mode)		
Port 1	P10/TO110	P10 (input mode)		PMC1
	P12/TCLR11	P12 (input mode)		
	P14/INTP110/DMAAK0	P14 (input mode)		PMC1, PCS1 <sup>Note</sup>
	P15/INTP111/DMAAK1	P15 (input mode)		
	P16/INTP112/DMAAK2	P16 (input mode)		
	P17/INTP113/DMAAK3	P17 (input mode)		
Port 2	P20/NMI	NMI		—
	P22/TXD0/SO0	P22 (input mode)		PMC2, ASIM00
	P23/RXD0/SI0	P23 (input mode)		
	P24/SCK0	P24 (input mode)		PMC2 <sup>Note</sup>
	P25/TXD1/SO1	P25 (input mode)		PMC2, ASIM10
	P26/RXD1/SI1	P26 (input mode)		
	P27/SCK1	P27 (input mode)		PMC2 <sup>Note</sup>
Port 3	P33/TI13	P33 (input mode)		PMC3
	P34/INTP130	P34 (input mode)		
Port 4	P40/D0 to P47/D7	D0 to D7		MM
Port 5	P50/D8 to P57/D15	D8 to D15	P50 to P57 (input mode)	MM
Port 6	P60/A16 to P67/A23	A16 to A23		MM
Port 7	P70/ANI0 to P73/ANI3	P70/ANI0 to P73/ANI3		—
Port 8	P80/CS0	CS0		PMC8
	P83/CS3/RAS3	CS3/RAS3		
	P84/CS4/RAS4/IOWR	CS4/RAS4		PMC8, PCS8 <sup>Note</sup>
	P85/CS5/RAS5/IORD	CS5/RAS5		

**Note** Selects the pin function when in the control mode.

(2/2)

Port Name	Pin Name	Pin Function After Reset		Register Which Sets the Mode
		ROM-less Mode 0	ROM-less Mode 1	
Port 9	P90/ $\overline{\text{LCAS}}$ / $\overline{\text{LWR}}$	$\overline{\text{LCAS}}$ / $\overline{\text{LWR}}$		PMC9
	P91/ $\overline{\text{UCAS}}$ / $\overline{\text{UWR}}$	$\overline{\text{UCAS}}$ / $\overline{\text{UWR}}$		
	P92/ $\overline{\text{RD}}$	$\overline{\text{RD}}$		
	P93/ $\overline{\text{WE}}$	$\overline{\text{WE}}$		
	P94/ $\overline{\text{BCYST}}$	$\overline{\text{BCYST}}$		PMC9
	P95/ $\overline{\text{OE}}$	$\overline{\text{OE}}$		PMC9
	P96/ $\overline{\text{HLD\!AK}}$	$\overline{\text{HLD\!AK}}$		
	P97/ $\overline{\text{HLDRQ}}$	$\overline{\text{HLDRQ}}$		
Port 10	P100/ $\text{TO120}$	P100 (input mode)		PMC10
	P102/ $\text{TCLR12}$	P102 (input mode)		
Port A	PA0/A0 to PA7/A7	A0 to A7		MM
Port B	PB0/A8 to PB7/A15	A8 to A15		MM
Port X	PX6/ $\overline{\text{WAIT}}$	$\overline{\text{WAIT}}$		PMCX
	PX7/ $\text{CLKOUT}$	$\text{CLKOUT}$		

**Note** Selects the pin function when in the control mode.

## (3) Block diagram of port

Figure 12-1. Type A Block Diagram

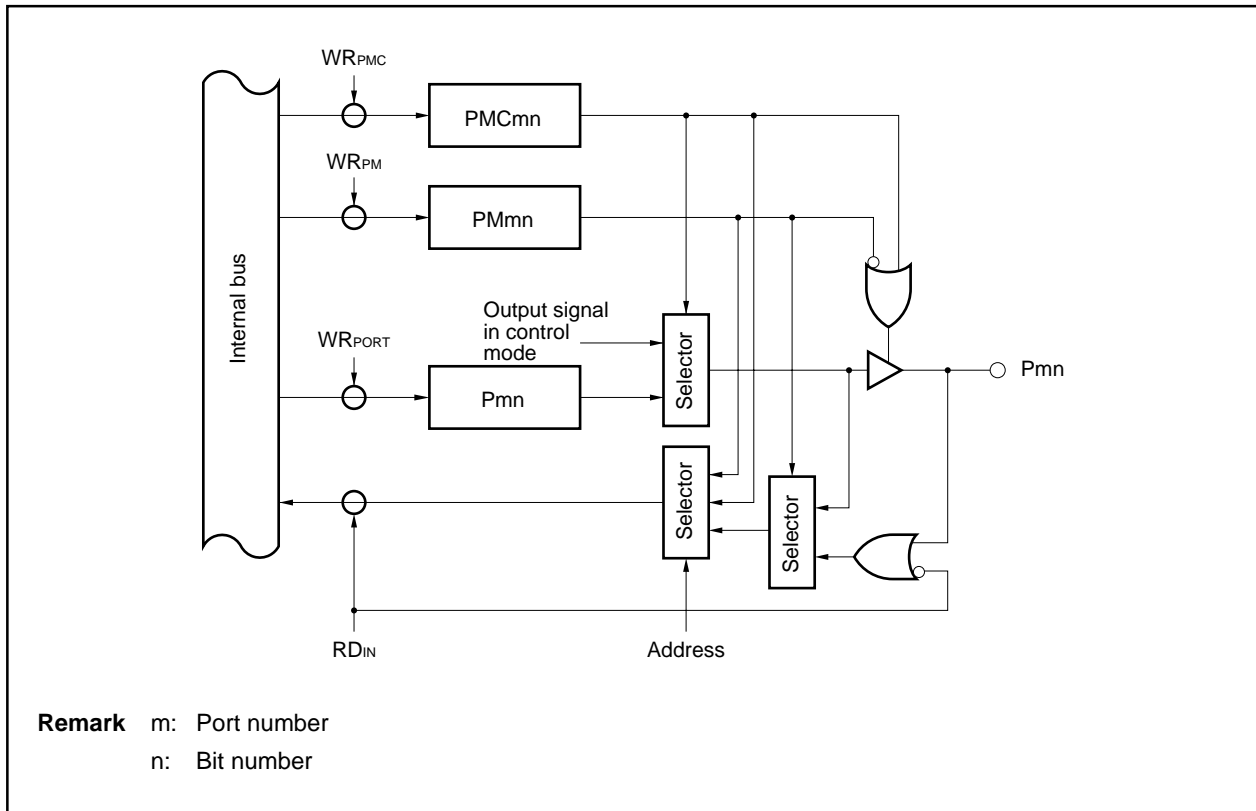


Figure 12-2. Type B Block Diagram

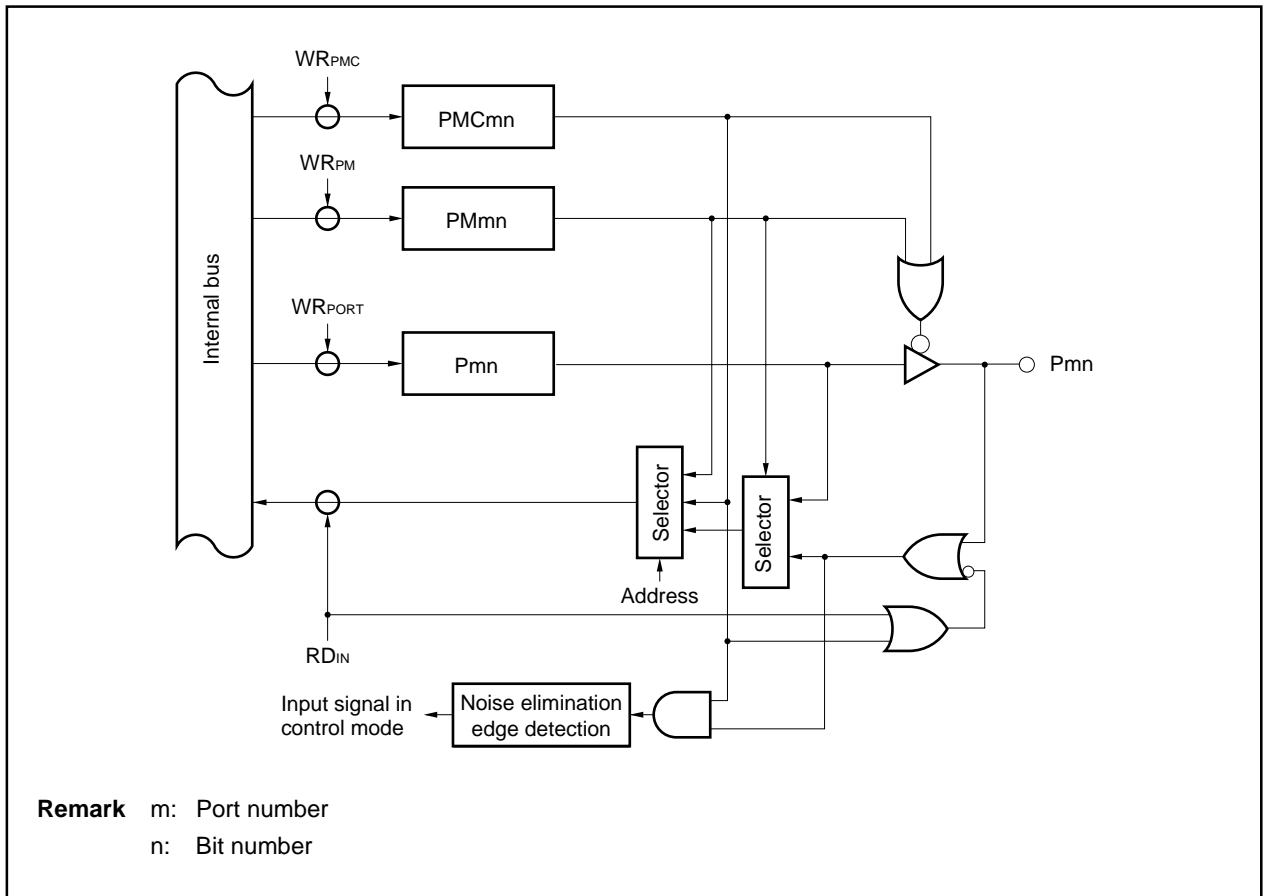
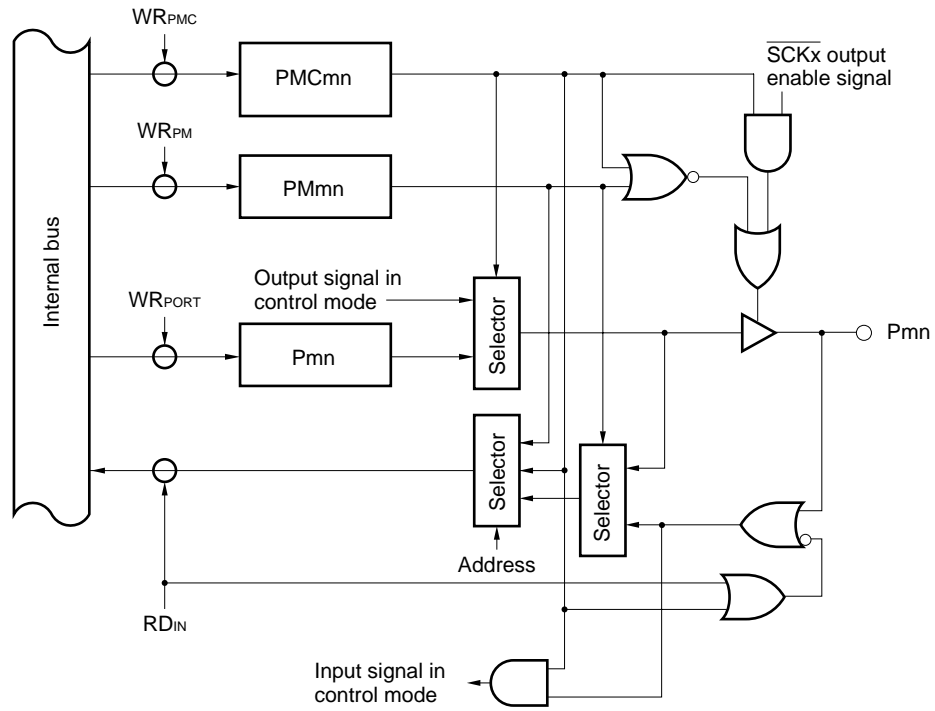


Figure 12-3. Type C Block Diagram



**Remark** mn: 24, 27  
 x: 0 (when mn = 24), 1 (when mn = 27)



Figure 12-4. Type D Block Diagram

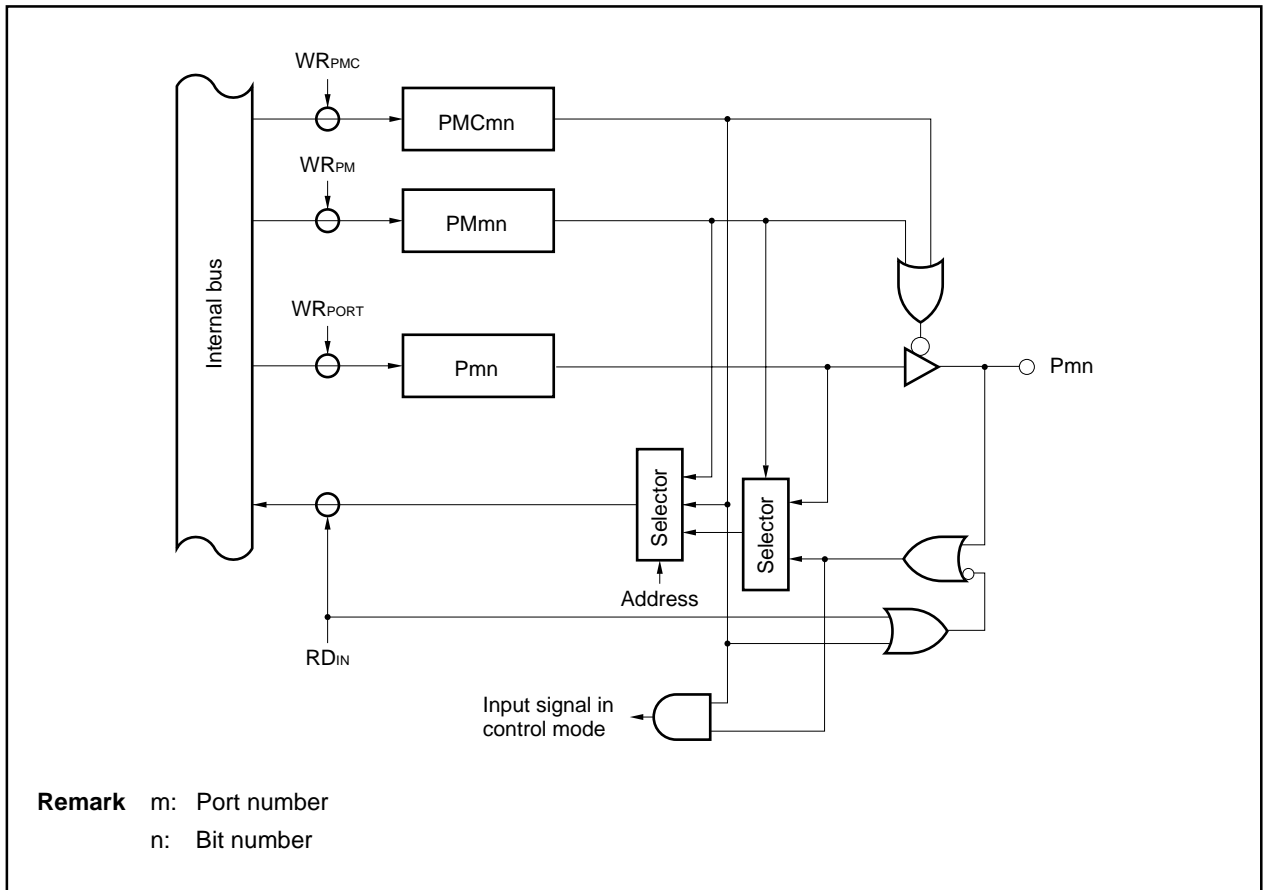
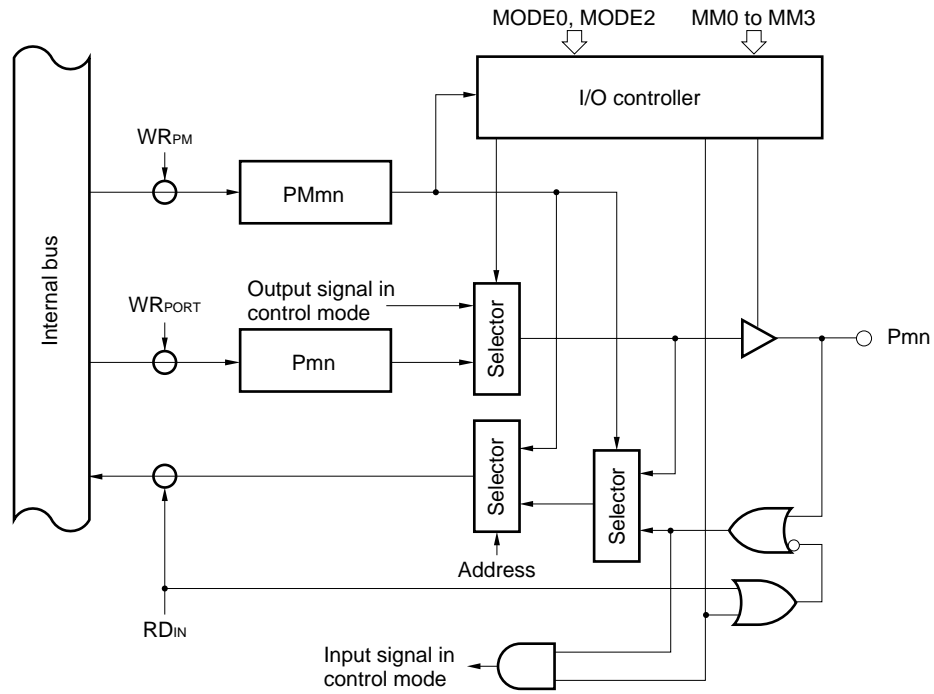
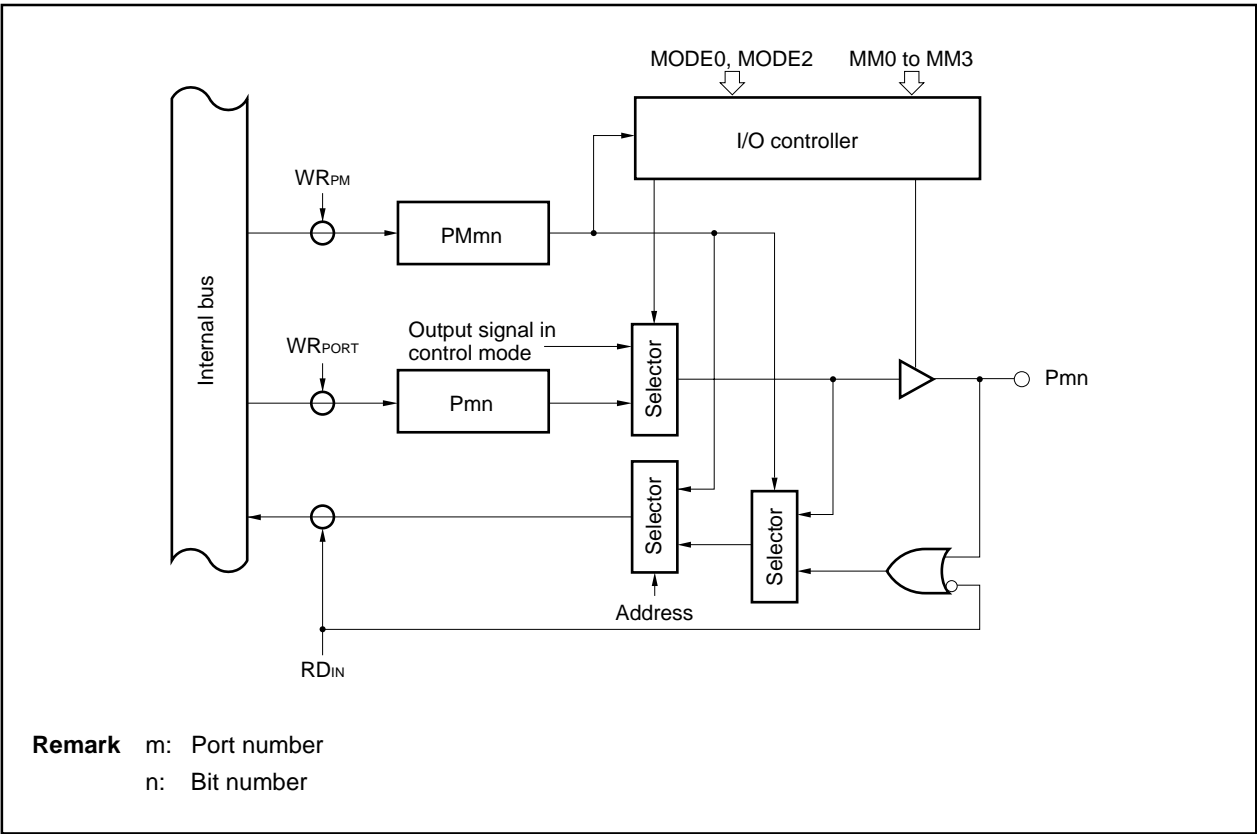


Figure 12-5. Type E Block Diagram



**Remark** m: Port number  
n: Bit number

**Figure 12-6. Type F Block Diagram**



**Figure 12-7. Type G Block Diagram**

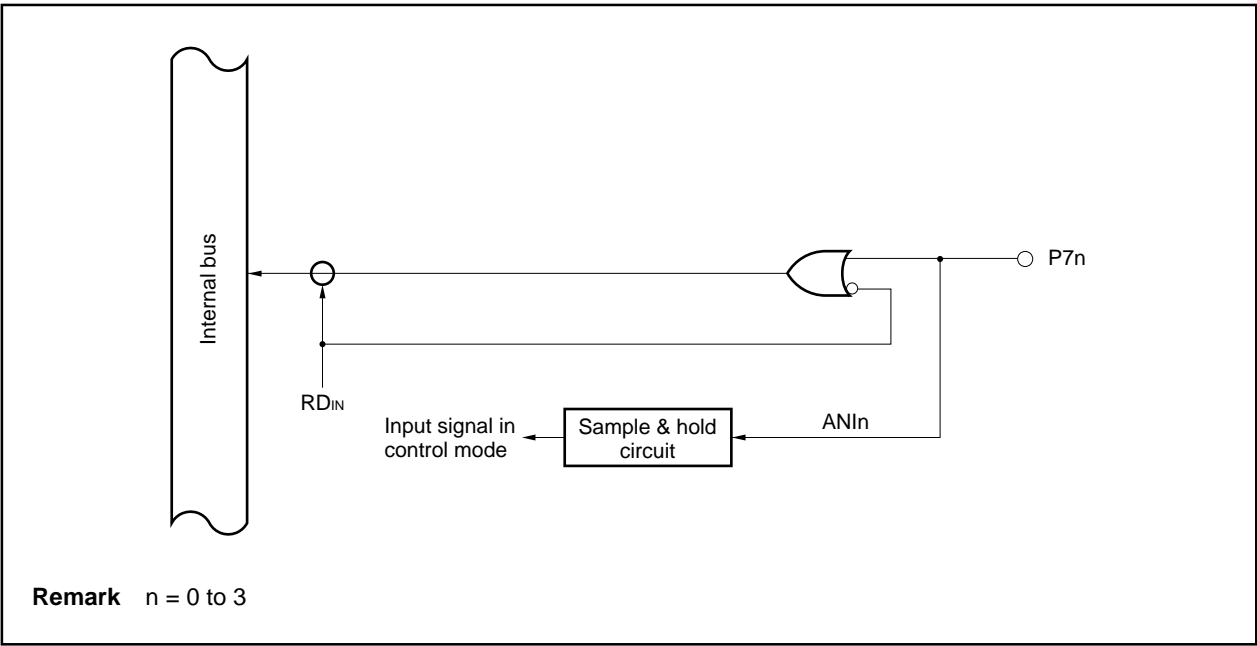


Figure 12-8. Type H Block Diagram

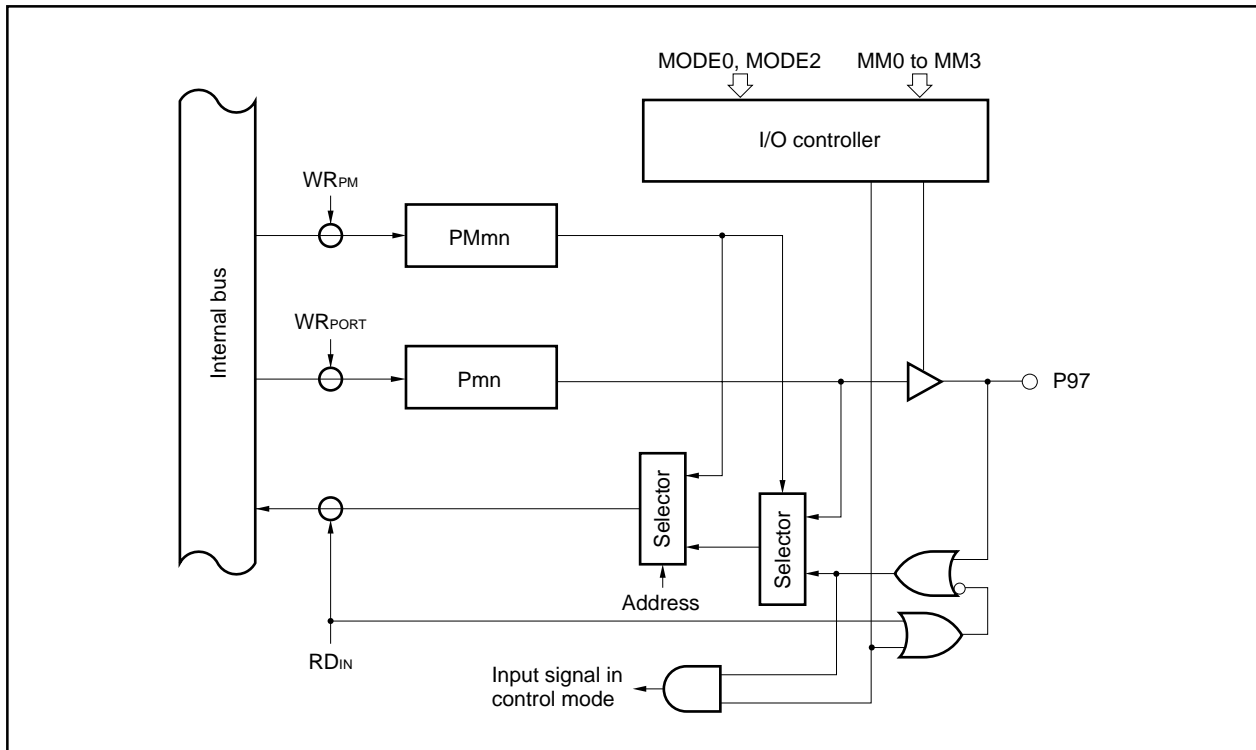
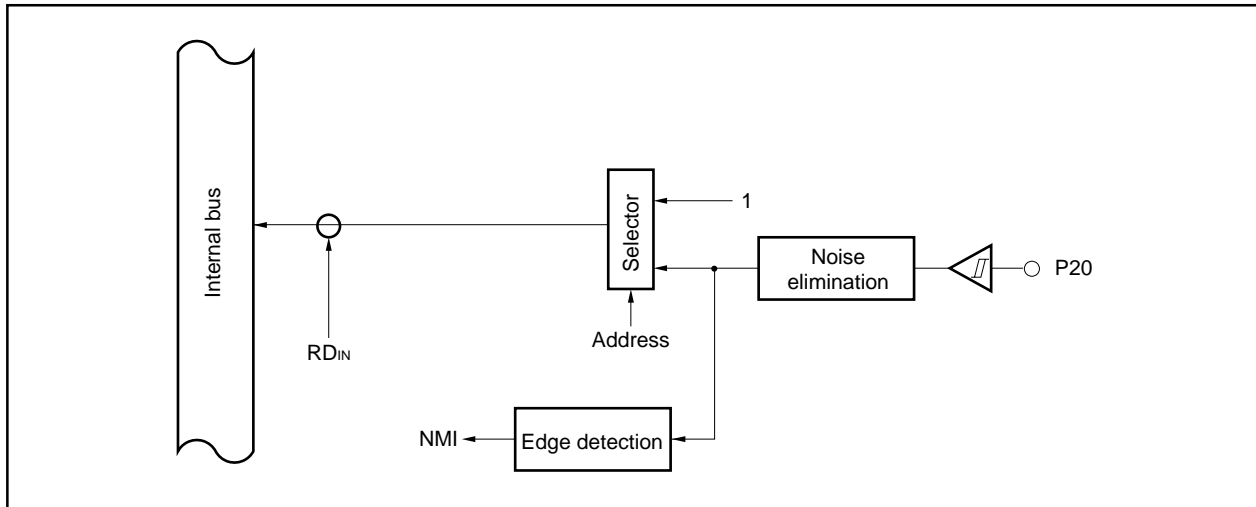
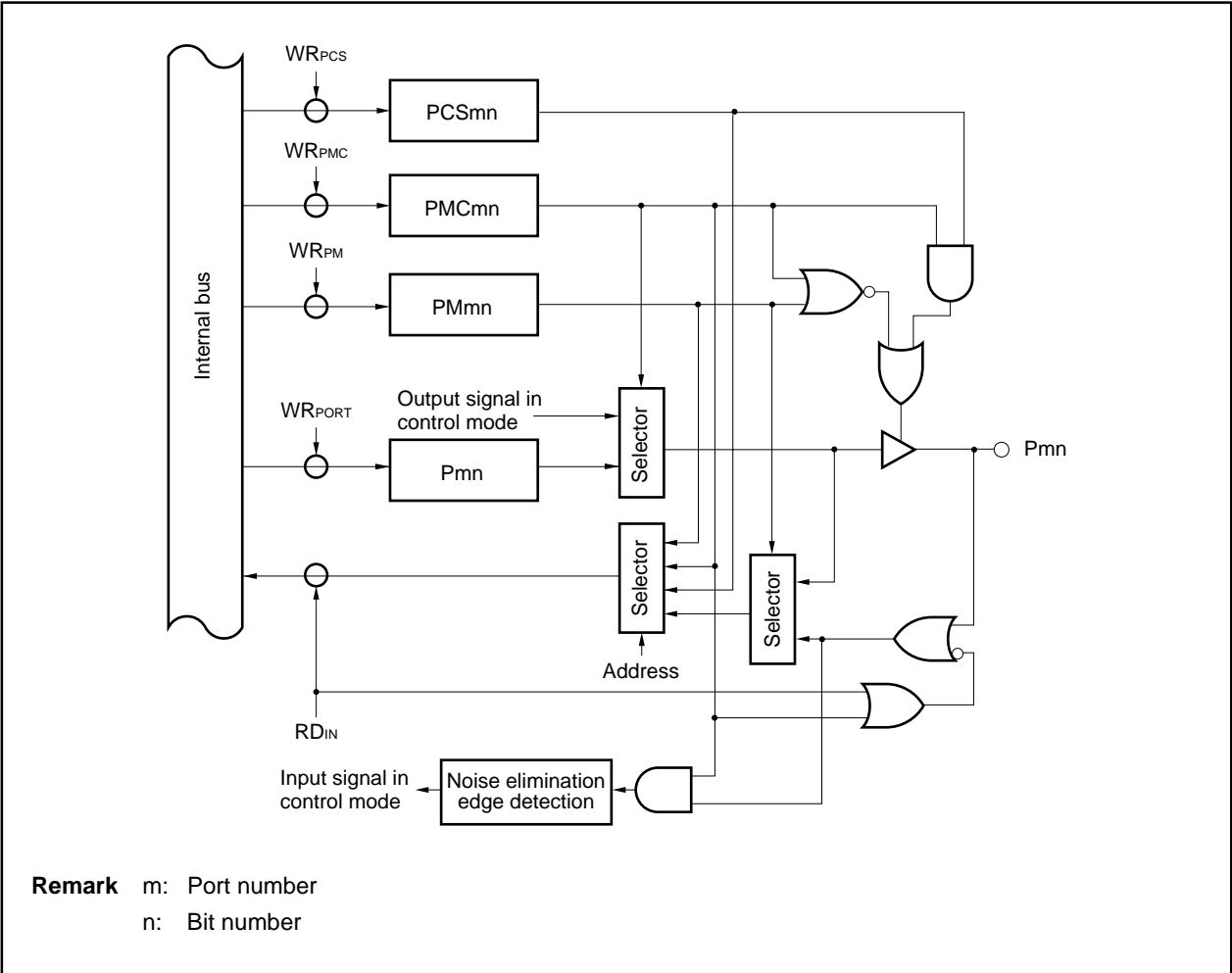


Figure 12-9. Type I Block Diagram

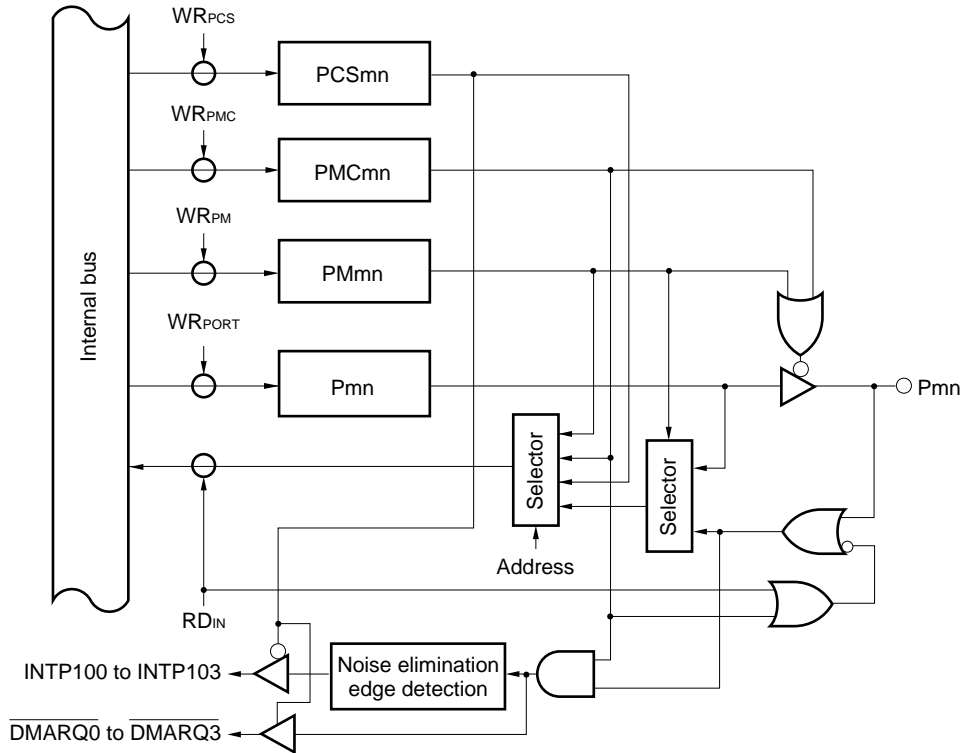


**Figure 12-10. Type K Block Diagram**



**Remark**    m: Port number  
              n: Bit number

Figure 12-11. Type M Block Diagram



**Remark** mn: 04 to 07

Figure 12-12. Type O Block Diagram

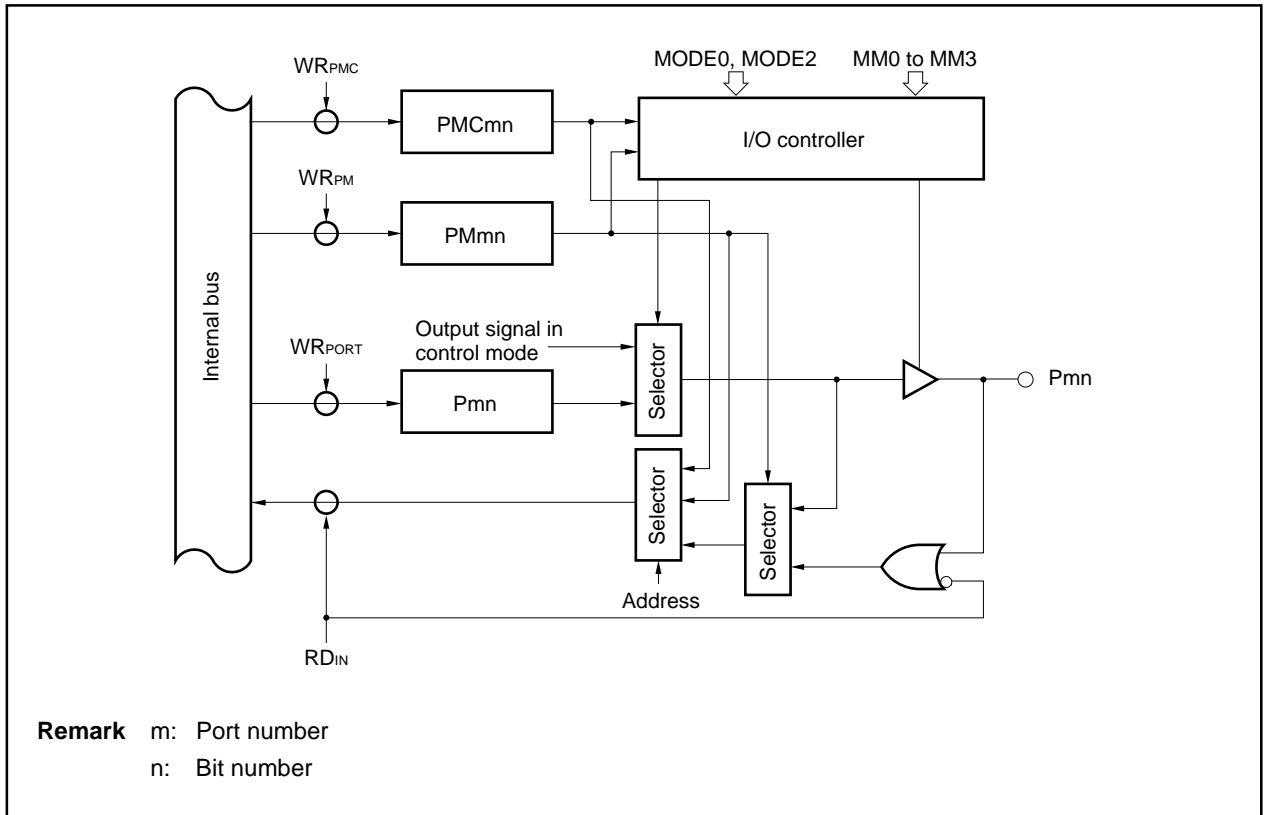
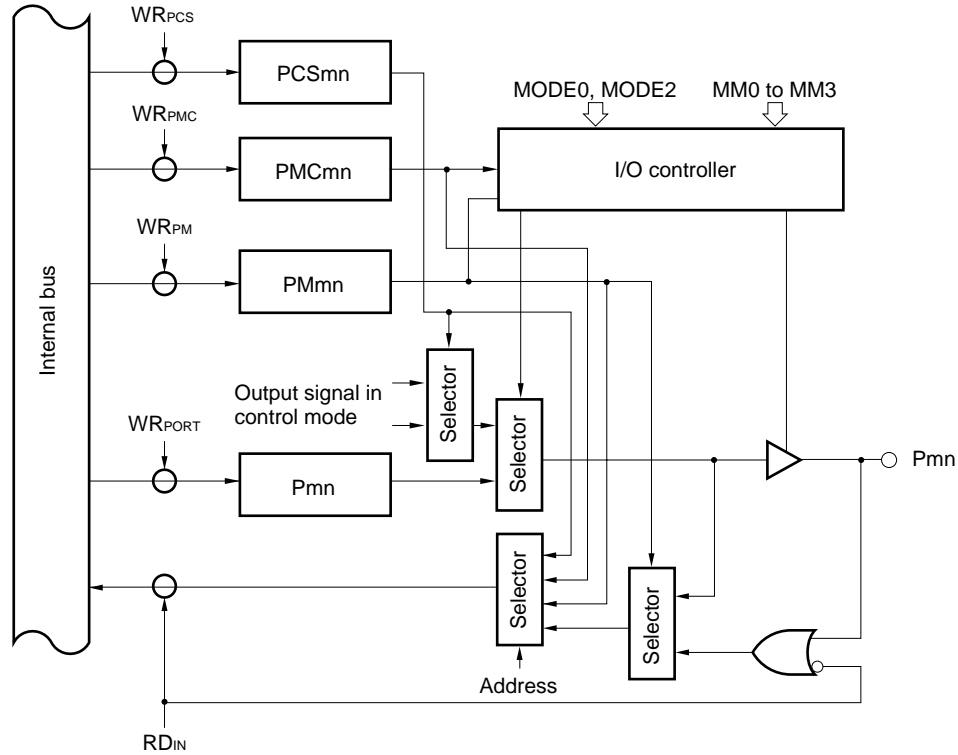


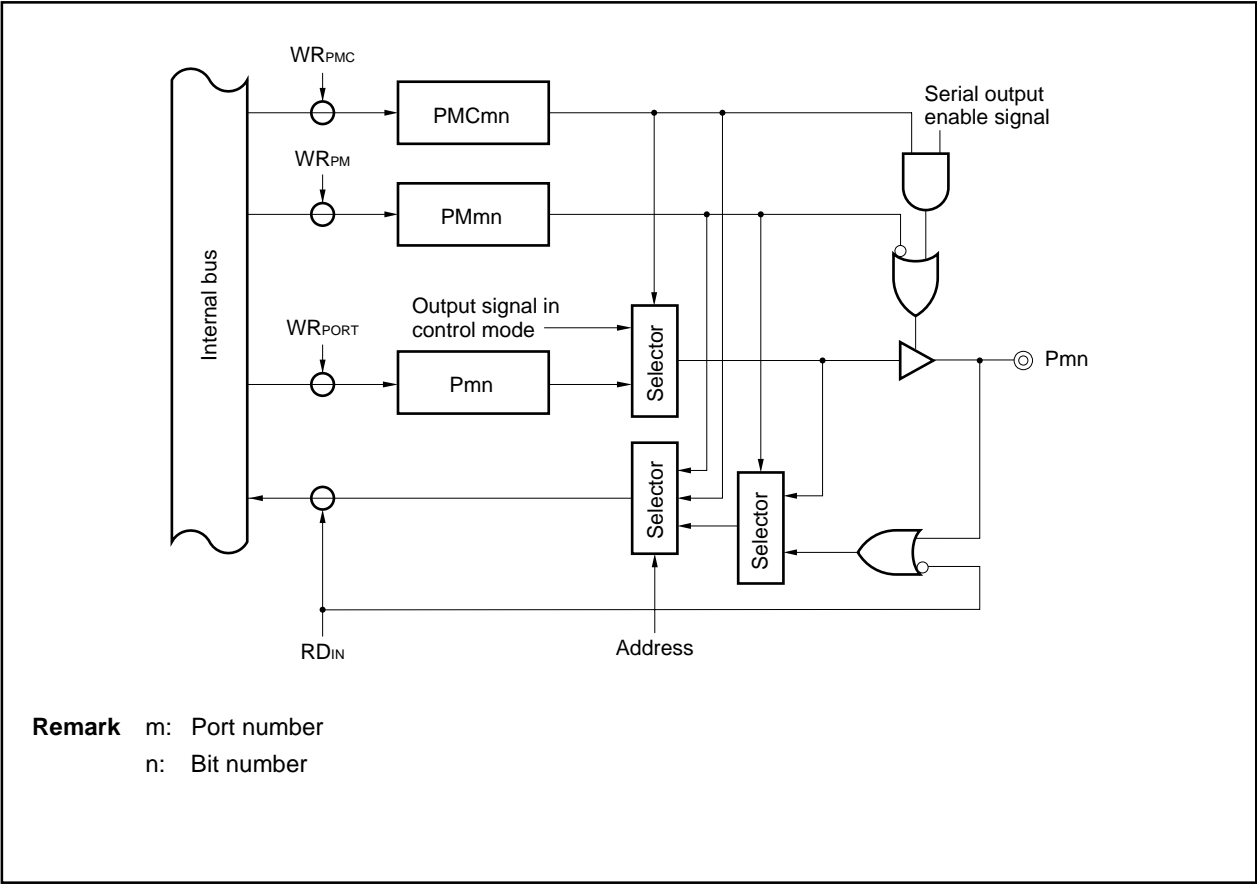
Figure 12-13. Type P Block Diagram



**Remark** m: Port number  
n: Bit number



Figure 12-14. Type Q Block Diagram



## 12.3 Port Pin Functions

### 12.3.1 Port 0

Port 0 is a 6-bit input/output port that can be set to input or output in 1-bit units.

P0	7	6	5	4	3	2	1	0	Address FFFFFF00H	After reset Undefined
	P07	P06	P05	P04	P03	P02	P01	P00		

Bit Position	Bit Name	Function
7 to 4, 2, 0	P0n (n = 7 to 4, 2, 0)	Port 0 Input/output port

In addition to their function as port pins, the port 0 pins can also operate as real-time pulse unit (RPU) inputs/outputs, external interrupt request inputs, and DMA request inputs in the control mode.

#### (1) Operation in control mode

Port	Control Mode	Remark	Block Type
Port 0	P00	TO100	Real-time pulse unit (RPU) output
	P02	TCLR10	Real-time pulse unit (RPU) input
	P04 to P07	INTP100/DMARQ0 to INTP103/DMARQ3	External interrupt request input/DMA request input

#### (2) Input/output mode/control mode setting

Port 0 input/output mode setting is performed by means of the port 0 mode register (PM0), and control mode setting is performed by means of the port 0 mode control register (PMC0) and port/control select register 0 (PCS0).

##### (a) Port 0 mode register (PM0)

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

PM0	7	6	5	4	3	2	1	0	Address FFFFFF020H	After reset FFH
	PM07	PM06	PM05	PM04	<sup>Note</sup> 1	PM02	<sup>Note</sup> 1	PM00		

Bit Position	Bit Name	Function
7 to 4, 2, 0	PM0n (n = 7 to 4, 2, 0)	Port Mode Specifies the input/output mode of pin P0n. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Note** Be sure to set these bits to 1.

**(b) 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	0 <sup>Note</sup>	PMC02	0 <sup>Note</sup>	PMC00	Address FFFFF040H	After reset 00H

Bit Position	Bit Name	Function
7 to 4	PMC0n (n = 7 to 4)	Port Mode Control Specifies the operation mode of pin P0n. Sets in combination with the PCS0 register. 0: Input/output port mode 1: External interrupt request (INTP103 to INTP100) input mode/DMA request (DMARQ3 to DMARQ0) input mode
2	PMC02	Port Mode Control Sets operation mode of P02 pin. 0: Input/output port mode 1: TCLR10 input mode
0	PMC00	Port Mode Control Sets operation mode of P00 pin. 0: Input/output port mode 1: TO100 output mode

**Note** Be sure to set these bits to 0.

**(c) Port/control select register 0 (PCS0)**

This register can be read/written in 8- or 1-bit units. However, bits 3 to 0 are fixed at 0, so even if 1 is written, it is disregarded.

	7	6	5	4	3	2	1	0		
PCS0	PCS07	PCS06	PCS05	PCS04	0	0	0	0	Address FFFF580H	After reset 00H

Bit Position	Bit Name	Function
7	PCS07	Port Control Select Specifies the operating mode when pin P07 is in the control mode. 0: INTP103 input mode 1: DMARQ3 input mode
6	PCS06	Port Control Select Specifies the operating mode when pin P06 is in the control mode. 0: INTP102 input mode 1: DMARQ2 Input mode
5	PCS05	Port Control Select Specifies the operating mode when pin P05 is in the control mode. 0: INTP101 input mode 1: DMARQ1 input mode
4	PCS04	Port Control Select Specifies the operating mode when pin P04 is in the control mode. 0: INTP100 input mode 1: DMARQ0 input mode

**Caution** When the port mode is specified by the PMC0 register, the settings of this register are ignored.

### 12.3.2 Port 1

Port 1 is a 6-bit input/output port that can be set to input or output 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 4, 2, 0	P1n (n = 7 to 4, 2, 0)	Port 1 Input/output port

In addition to their function as port pins, the port 1 pins can also operate as real-time pulse unit (RPU) inputs/outputs, external interrupt request inputs, and DMA acknowledge outputs in the control mode.

#### (1) Operation in control mode

Port		Control Mode	Remark	Block Type
Port 1	P10	TO110	Real-time pulse unit (RPU) output	A
	P12	TCLR11	Real-time pulse unit (RPU) input	B
	P14 to P17	INTP110/DMAAK0 to INTP113/DMAAK3	External interrupt input/DMA acknowledge output	K

#### (2) Input/output mode/control mode setting

Port 1 input/output mode setting is performed by means of the port 1 mode register (PM1), and control mode setting is performed by means of the port 1 mode control register (PMC1) and port/control select register 1 (PCS1).

##### (a) Port 1 mode register (PM1)

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

PM1	7	6	5	4	3	2	1	0	Address FFFFF022H	After reset FFH
	PM17	PM16	PM15	PM14	1 <sup>Note</sup>	PM12	1 <sup>Note</sup>	PM10		

Bit Position	Bit Name	Function
7 to 4, 2, 0	PM1n (n = 7 to 4, 2, 0)	Port Mode Sets P1n in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Note** Be sure to set these bits to 1.

**(b) Port 1 mode control register (PMC1)**

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

	7	6	5	4	3	2	1	0		
PMC1	PMC17	PMC16	PMC15	PMC14	0 <sup>Note</sup>	PMC12	0 <sup>Note</sup>	PMC10	Address FFFFF042H	After reset 00H

Bit Position	Bit Name	Function
7 to 4	PMC1n (n = 7 to 4)	Port Mode Control Sets operation mode of P1n pin. Set in combination with PCS1. 0: Input/output port mode 1: External interrupt request (INTP113 to INTP110) input mode/ DMA acknowledge ( $\overline{\text{DMAAK3}}$ to $\overline{\text{DMAAK0}}$ ) output mode
2	PMC12	Port Mode Control Sets operation mode of P12 pin. 0: Input/output port mode 1: TCLR11 input mode
0	PMC10	Port Mode Control Sets operation mode of P10 pin. 0: Input/output port mode 1: TO110 output mode

**Note** Be sure to set these bits to 0.

**(c) Port/control select register 1 (PCS1)**

This register can be read/written in 8- or 1-bit units. However, bits 3 to 0 are fixed at 0, so even if 1 is written, it is disregarded.

	7	6	5	4	3	2	1	0		
PCS1	PCS17	PCS16	PCS15	PCS14	0	0	0	0	Address FFFF582H	After reset 00H

Bit Position	Bit Name	Function
7	PCS17	Port Control Select Specifies the operating mode when pin P17 is in the control mode. 0: <u>INTP113</u> input mode 1: <u>DMAAK3</u> output mode
6	PCS16	Port Control Select Specifies the operating mode when pin P16 is in the control mode. 0: <u>INTP112</u> input mode 1: <u>DMAAK2</u> output mode
5	PCS15	Port Control Select Specifies the operating mode when pin P15 is in the control mode. 0: <u>INTP111</u> input mode 1: <u>DMAAK1</u> output mode
4	PCS14	Port Control Select Specifies the operating mode when pin P14 is in the control mode. 0: <u>INTP110</u> input mode 1: <u>DMAAK0</u> output mode

**Caution** When the port mode is specified by the PMC1 register, the settings of this register are ignored.

### 12.3.3 Port 2

Port 2 is a 7-bit input/output port that can be set to input or output in 1-bit units. However, P20 always operates as an NMI input if the edge is input.

	7	6	5	4	3	2	1	0		
P2	P27	P26	P25	P24	P23	P22	P21	P20	Address FFFFF004H	After reset Undefined

Bit Position	Bit Name	Function
7 to 2	P2n (n = 7 to 2)	Port 2 Input/output port
0	P20	Fix to NMI input mode.

In addition to their function as port pins, the port 2 pins can also operate as serial interface (UART0/CSI0, UART1/CSI1) inputs/outputs in the control mode.

#### (1) Operation in control mode

Port	Control Mode	Remark	Block Type
Port 2	P20	NMI	I
	P22	TXD0/SO0	Q
	P23	RXD0/SI0	D
	P24	$\overline{\text{SCK0}}$	C
	P25	TXD1/SO1	Q
	P26	RXD1/SI1	D
	P27	$\overline{\text{SCK1}}$	C



**(2) Input/output mode/control mode setting**

Port 2 input/output mode setting is performed by means of the port 2 mode register (PM2), and control mode setting is performed by means of the port 2 mode control register (PMC2). Pin P20 is fixed to NMI input mode.

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

This register can be read/written in 8- or 1-bit units. However, bit 0 is fixed at 1 by hardware, so writing 0 to this bit is ignored.

	7	6	5	4	3	2	1	0		
PM2	PM27	PM26	PM25	PM24	PM23	PM22	1 <sup>Note</sup>	1	Address FFFFF024H	After reset FFH

Bit Position	Bit Name	Function
7 to 2	PM2n (n = 7 to 2)	Port Mode Sets P2n in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Note** Be sure to set these bits to 1.

**Caution** When the serial interface is used, use the following bits in the state when they are set to 1 (initial value).

When UART0 is used: PM22

When UART1 is used: PM25

When CSI0 is used: PM24 to PM22

When CSI1 is used: PM27 to PM25

**(b) 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, so writing 0 to this bit is ignored. Bit 1 is fixed to 0, so writing 1 to this bit is ignored.

	7	6	5	4	3	2	1	0		
PMC2	PMC27	PMC26	PMC25	PMC24	PMC23	PMC22	0	1	Address FFFFF044H	After reset 01H

Bit Position	Bit Name	Function
7	PMC27	Port Mode Control Sets operation mode of P27 pin. 0: Input/output port mode 1: SCK1 input/output mode
6	PMC26	Port Mode Control Sets operation mode of P26 pin. 0: Input/output port mode 1: RXD1/SI1 input mode
5	PMC25	Port Mode Control Sets operation mode of P25 pin. 0: Input/output port mode 1: TXD1/SO1 output mode
4	PMC24	Port Mode Control Sets operation mode of P24 pin. 0: Input/output port mode 1: SCK0 input/output mode
3	PMC23	Port Mode Control Sets operation mode of P23 pin. 0: Input/output port mode 1: RXD0/SI0 input mode
2	PMC22	Port Mode Control Sets operation mode of P22 pin. 0: Input/output port mode 1: TXD0/SO0 output mode

**Remark** UART0 and CSI0, and UART1 and CSI1 share the same pins respectively. Either one of these is selected according to the ASIM00 and ASIM10 registers (refer to **10.2.3 Control registers**).

### 12.3.4 Port 3

Port 3 is a 2-bit input/output port that can be set to input or output in 1-bit units.

	7	6	5	4	3	2	1	0		
P3	P37	P36	P35	P34	P33	P32	P31	P30	Address FFFFF006H	After reset Undefined

Bit Position	Bit Name	Function
4, 3	P3n (n = 4, 3)	Port 3 Input/output port

In addition to their function as port pins, the port 3 pins can also operate as the input/output signals of the real-time pulse unit (RPU), the input signals of external interrupt, and the input/output lines of the serial interface (CSI2) when in the control mode.

#### (1) Operation in control mode

Port		Control Mode	Remark	Block Type
Port 3	P33	TI13	Real-time pulse unit (RPU) input	B
	P34	INTP130	External interrupt input	

**(2) Input/output mode/control mode setting**

Port 3 input/output mode setting is performed by means of the port 3 mode register (PM3), and control mode setting is performed by means of the port 3 mode control register (PMC3) and port/control select register 3 (PCS3).

**(a) 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 <sup>Note</sup>	1 <sup>Note</sup>	1 <sup>Note</sup>	PM34	PM33	1 <sup>Note</sup>	1 <sup>Note</sup>	1 <sup>Note</sup>	Address FFFFF026H	After reset FFH

Bit Position	Bit Name	Function
4, 3	PM3n (n = 4, 3)	Port Mode Sets P3n in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Note** Be sure to set these bits to 1.

**(b) Port 3 mode control register (PMC3)**

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

	7	6	5	4	3	2	1	0		
PMC3	0 <sup>Note</sup>	0 <sup>Note</sup>	0 <sup>Note</sup>	PMC34	PMC33	0 <sup>Note</sup>	0 <sup>Note</sup>	0 <sup>Note</sup>	Address FFFFF046H	After reset 00H

Bit Position	Bit Name	Function
4	PMC34	Port Mode Control Sets operation mode of P34 pin. 0: Input/output port mode 1: INTP130 input mode
3	PMC33	Port Mode Control Sets operation mode of P33 pin. 0: Input/output port mode 1: TI13 input mode

**Note** Be sure to set these bits to 0.

### 12.3.5 Port 4

Port 4 is an 8-bit input/output port that can be set to input or output in 1-bit units.

P4	7	6	5	4	3	2	1	0	Address FFFFF008H	After reset Undefined
	P47	P46	P45	P44	P43	P42	P41	P40		

Bit Position	Bit Name	Function
7 to 0	P4n (n = 7 to 0)	Port 4 Input/output port

In addition to their function as port pins, the port 4 pins can also operate in the control mode (external expansion mode) as a data bus used when memory is expanded externally.

#### (1) Operation in control mode

Port		Control Mode	Remark	Block Type
Port 4	P40 to P47	D0 to D7	Data bus in memory expansion	E

**(2) Input/output mode/control mode setting**

Port 4 input/output mode setting is performed by means of the port 4 mode register (PM4), and control mode (external expansion mode) setting is performed by means of the mode specification pins (MODE0, MODE2) and the memory expansion mode register (MM: refer to 3.4.6 (1)).

**(a) Port 4 mode register (PM4)**

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

PM4	7	6	5	4	3	2	1	0	Address FFFFF028H	After reset FFH
	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40		

Bit Position	Bit Name	Function
7 to 0	PM4n (n = 7 to 0)	Port Mode Sets P4n in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**(b) Operation mode of port 4**

Bit of MM Register				Operation Mode							
MM3	MM2	MM1	MM0	P40	P41	P42	P43	P44	P45	P46	P47
don't care	0	0	0	Port (P40 to P47)							
	0	0	1								
	0	1	0	Data bus (D0 to D7)							
	0	1	1								
	1	0	0								
	1	0	1								
	1	1	0								
	1	1	1								

For the details of mode selection by the MODE0 and MODE2 pins, refer to 3.3.2 Operating mode specification.

The MM0 to MM3 bits are initialized to 111× at system reset, enabling the external expansion mode. External expansion can be disabled by programming the MM0 to MM3 bits and setting the port mode. If MM0 to MM3 are set to 000×, the subsequent external instruction cannot be fetched.

**Remark** ×: don't care

### 12.3.6 Port 5

Port 5 is an 8-bit input/output port that can be set to input or output in 1-bit units.

	7	6	5	4	3	2	1	0		
P5	P57	P56	P55	P54	P53	P52	P51	P50	Address FFFFFF00AH	After reset Undefined

Bit Position	Bit Name	Function
7 to 0	P5n (n = 7 to 0)	Port 5 Input/output port

In addition to their function as port pins, the port 5 pins can also operate in the control mode (external expansion mode) as a data bus used when memory is expanded externally.

#### (1) Operation in control mode

Port		Control Mode	Remark	Block Type
Port 5	P50 to P57	D8 to D15	Data bus in memory expansion	E

**(2) Input/output mode/control mode setting**

Port 5 input/output mode setting is performed by means of the port 5 mode register (PM5), and control mode (external expansion mode) setting is performed by means of the mode specification pins (MODE0, MODE2) and the memory expansion mode register (MM: refer to 3.4.6 (1)).

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

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

PM5	7	6	5	4	3	2	1	0	Address FFFFFF02AH	After reset FFH
	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50		

Bit Position	Bit Name	Function
7 to 0	PM5n (n = 7 to 0)	Port Mode Sets P5n in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**(b) Operation mode of port 5**

Bit of MM Register				Operation Mode							
MM3	MM2	MM1	MM0	P50	P51	P52	P53	P54	P55	P56	P57
0	0	0	0	Port (P50 to P57)							
0	0	0	1	Data bus (D8 to D15)							
0	0	1	0								
0	0	1	1								
0	1	0	0								
0	1	0	1								
0	1	1	0								
0	1	1	1								
1	don't care			Port (50 to P57)							

For the details of mode selection by the MODE0 and MODE2 pins, refer to 3.3.2 **Operating mode specification**.

The MM0 to MM3 bits are initialized to 1110 at system reset, enabling the external expansion mode. External expansion can be disabled by programming the MM0 to MM3 bits and setting the port mode. If MM0 to MM3 are set to  $\times\times\times 1$  or 0000, the subsequent external instruction cannot be fetched.

**Remark**  $\times$ : don't care



### 12.3.7 Port 6

Port 6 is an 8-bit input/output port that can be set to input or output in 1-bit units.

	7	6	5	4	3	2	1	0		
P6	P67	P66	P65	P64	P63	P62	P61	P60	Address FFFFFF00CH	After reset Undefined

Bit Position	Bit Name	Function
7 to 0	P6n (n = 7 to 0)	Port 6 Input/output port

In addition to their function as port pins, the port 6 pins can also operate in the control mode (external expansion mode) as an address bus used when memory is expanded externally.

#### (1) Operation in control mode

Port		Control Mode	Remark	Block Type
Port 6	P60 to P67	A16 to A23	Address bus in memory expansion	F

**(2) Input/output mode/control mode setting**

Port 6 input/output mode setting is performed by means of the port 6 mode register (PM6), and control mode (external expansion mode) setting is performed by means of the mode specification pins (MODE0, MODE2) and the memory expansion mode register (MM: refer to 3.4.6 (1)).

**(a) Port 6 mode register (PM6)**

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

PM6	7	6	5	4	3	2	1	0	Address FFFFFF02CH	After reset FFH
	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60		

Bit Position	Bit Name	Function
7 to 0	PM6n (n = 7 to 0)	Port Mode Sets P6n in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**(b) Operation mode of port 6**

Bit of MM Register				Operation Mode							
MM3	MM2	MM1	MM0	P60	P61	P62	P63	P64	P65	P66	P67
don't care	0	0	0	Port (P60 to P67)							
	0	0	1								
	0	1	0								
	0	1	1								
	1	0	0	A16	A17	P62	P63	P64	P65	P66	P67
	1	0	1			A18	A19				
	1	1	0			A18	A19				
	1	1	1								

For the details of mode selection by the MODE0 and MODE2 pins, refer to 3.3.2 **Operating mode specification**.

The MM0 to MM3 bits are initialized to 111× at system reset, enabling the external expansion mode. External expansion can be disabled by programming the MM0 to MM3 bits and setting the port mode.

**Remark** ×: don't care

### 12.3.8 Port 7

Port 7 is a 4-bit input only port and all pins of port 7 are fixed in the input mode.

	7	6	5	4	3	2	1	0		
P7	P77	P76	P75	P74	P73	P72	P71	P70	Address FFFFF00EH	After reset Undefined

In addition to their function as port pins, the port 7 pins can also operate as analog inputs for A/D converter.

This port is used also as the analog input pins (ANI0 to ANI3), but the port and analog input pins cannot be switched. By reading the port, the state of each pin can be read.

#### (1) Operation in control mode

Port		Control Mode	Remark	Block Type
Port 7	P70 to P73	ANI0 to ANI3	Analog input for A/D converter	G

**12.3.9 Port 8**

Port 8 is a 5-bit input/output port that can be set to input or output in 1-bit units.

	7	6	5	4	3	2	1	0		
P8	P87	P86	P85	P84	P83	P82	P81	P80	Address FFFFFF010H	After reset Undefined

Bit Position	Bit Name	Function
5 to 3, 0	P8n (n = 5 to 3, 0)	Port 8 Input/output port

In addition to their function as port pins, in the control mode, the port 8 pins operate as chip select signal outputs, row address strobe signal outputs for DRAM, and read/write strobe signal outputs for external I/O.

**(1) Operation in control mode**

Port		Control Mode	Remark	Block Type
Port 8	P80	$\overline{CS0}$	Chip select signal output Row address signal output	O
	P83	$\overline{CS3/RAS3}$	Chip select signal output Row address signal output	
	P84	$\overline{CS4/RAS4/IOWR}$	Chip select signal output Row address signal output Write strobe signal output	P
	P85	$\overline{CS5/RAS5/IORD}$	Chip select signal output Row address signal output Read strobe signal output	

## (2) Input/output mode/control mode setting

Port 8 input/output mode setting is performed by means of the port 8 mode register (PM8), and control mode (external expansion mode) setting is performed by means of the mode specification pins (MODE0, MODE2) and the port 8 mode control register (PMC8).

### (a) Port 8 mode register (PM8)

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

PM8	7	6	5	4	3	2	1	0	Address FFFFFF030H	After reset FFH
	1 <sup>Note</sup>	1 <sup>Note</sup>	PM85	PM84	PM83	1 <sup>Note</sup>	1 <sup>Note</sup>	PM80		

Bit Position	Bit Name	Function
5 to 3, 0	PM8n (n = 5 to 3, 0)	Port Mode Sets P8n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Note** Be sure to set these bits to 1.

**(b) Port 8 mode control register (PMC8)**

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

	7	6	5	4	3	2	1	0		
PMC8	1 <sup>Note</sup>	1 <sup>Note</sup>	PMC85	PMC84	PMC83	1 <sup>Note</sup>	1 <sup>Note</sup>	PMC80	Address FFFF050H	After reset FFH

Bit Position	Bit Name	Function
5	PMC85	Port Mode Control Sets operation mode of P85 pin. Set in combination with PCS8. 0: Input/output port mode 1: CS5/RAS5 output mode/IORD output mode
4	PMC84	Port Mode Control Sets operation mode of P84 pin. Set in combination with PCS8. 0: Input/output port mode 1: CS4/RAS4 output mode/IOWR output mode
3	PMC83	Port Mode Control Sets operation mode of P83 pin. 0: Input/output port mode 1: CS3/RAS3 output mode
0	PMC80	Port Mode Control Sets operation mode of P80 pin. 0: Input/output port mode 1: CS0 output mode

**Note** Be sure to set these bits to 1.

**(c) Port/control select register 8 (PCS8)**

This register can be read/written in 8- or 1-bit units. However, all the bits except for bits 5 and 4 are fixed at 0, so even if 1 is written, it is disregarded.

	7	6	5	4	3	2	1	0		
PCS8	0	0	PCS85	PCS84	0	0	0	0	Address FFFFF590H	After reset 00H

Bit Position	Bit Name	Function
5	PCS85	Port Control Select Specifies the operating mode when pin P85 is in the control mode. 0: $\overline{CS5}/\overline{RAS5}$ output mode 1: $\overline{IORD}$ output mode
4	PCS84	Port Control Select Specifies the operating mode when pin P84 is in the control mode. 0: $\overline{CS4}/\overline{RAS4}$ output mode 1: $\overline{IOWR}$ output mode

**Caution** When the port mode is specified by the PMC8 register, the settings of this register are ignored.

**12.3.10 Port 9**

Port 9 is an 8-bit input/output port that can be set to input or output in 1-bit units.

	7	6	5	4	3	2	1	0		
P9	P97	P96	P95	P94	P93	P92	P91	P90	Address FFFFFF012H	After reset Undefined

Bit Position	Bit Name	Function
7 to 0	P9n (n = 7 to 0)	Port 9 Input/output port

In addition to their function as port pins, the port 9 pins can also operate in the control mode (external expansion mode) as control signal outputs and bus hold control signal output used when memory is expanded externally.

**(1) Operation in control mode**

Port	Control Mode	Remark	Block Type
Port 9	P90	$\overline{\text{LWR}}/\overline{\text{LCAS}}$	O
	P91	$\overline{\text{UWR}}/\overline{\text{UCAS}}$	
	P92	$\overline{\text{RD}}$	
	P93	$\overline{\text{WE}}$	
	P94	$\overline{\text{BCYST}}$	
	P95	$\overline{\text{OE}}$	
	P96	$\overline{\text{HLDK}}$	Bus hold acknowledge signal output
	P97	$\overline{\text{HLDRQ}}$	Bus hold request signal input H



## (2) Input/output mode/control mode setting

Port 9 input/output mode setting is performed by means of the port 9 mode register (PM9), and control mode (external expansion mode) setting is performed by means of the mode specification pins (MODE0, MODE2) and the port 9 mode control register (PMC9).

### (a) Port 9 mode register (PM9)

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

PM9	7	6	5	4	3	2	1	0	Address FFFFFF032H	After reset FFH
	PM97	PM96	PM95	PM94	PM93	PM92	PM91	PM90		

Bit Position	Bit Name	Function
7 to 0	PM9n (n = 7 to 0)	Port Mode Sets P9n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**(b) Port 9 mode control register (PMC9)**

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

	7	6	5	4	3	2	1	0		
PMC9	PMC97	PMC96	PMC95	PMC94	PMC93	PMC92	PMC91	PMC90	Address FFFFF052H	After reset FFH

Bit Position	Bit Name	Function
7	PMC97	Port Mode Control Sets operation mode of P97 pin. 0: Input/output port mode 1: HLDRQ input mode
6	PMC96	Port Mode Control Sets operation mode of P96 pin. 0: Input/output port mode 1: HLDAK output mode
5	PMC95	Port Mode Control Sets operation mode of P95 pin. 0: Input/output port mode 1: OE output mode
4	PMC94	Port Mode Control Sets operation mode of P94 pin. 0: Input/output port mode 1: BCYST output mode
3	PMC93	Port Mode Control Sets operation mode of P93 pin. 0: Input/output port mode 1: WE output mode
2	PMC92	Port Mode Control Sets operation mode of P92 pin. 0: Input/output port mode 1: RD output mode
1	PMC91	Port Mode Control Sets operation mode of P91 pin. 0: Input/output port mode 1: UWR/UCAS output mode
0	PMC90	Port Mode Control Sets operation mode of P90 pin. 0: Input/output port mode 1: LWR/LCAS output mode

### 12.3.11 Port 10

Port 10 is an 8-bit input/output port that can be set to input or output in 1-bit units.

P10	7	6	5	4	3	2	1	0	Address FFFFF014H	After reset Undefined
	P107	P106	P105	P104	P103	P102	P101	P100		

Bit Position	Bit Name	Function
2, 0	P10n (n = 2, 0)	Port 10 Input/output port

In addition to their function as port pins, the port 10 pins can also operate as real-time pulse unit (RPU) inputs/outputs and external interrupt inputs in the control mode.

#### (1) Operation in control mode

Port	Control Mode	Remark	Block Type
Port 10	P100	TO120	Real-time pulse unit (RPU) output A
	P102	TCLR12	Real-time pulse unit (RPU) input B

#### (2) Input/output mode/control mode setting

Port 10 input/output mode setting is performed by means of the port 10 mode register (PM10), and control mode setting is performed by means of the port 10 mode control register (PMC10) and port/control select register 10 (PCS10).

##### (a) Port 10 mode register (PM10)

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

PM10	7	6	5	4	3	2	1	0	Address FFFFF034H	After reset FFH
	1 <sup>Note</sup>	1 <sup>Note</sup>	1 <sup>Note</sup>	1 <sup>Note</sup>	1 <sup>Note</sup>	PM102	1 <sup>Note</sup>	PM100		

Bit Position	Bit Name	Function
2, 0	PM10n (n = 2, 0)	Port Mode Sets P10n pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Note** Be sure to set these bits to 1.

**(b) Port 10 mode control register (PMC10)**

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

	7	6	5	4	3	2	1	0		
PMC10	0 <sup>Note</sup>	0 <sup>Note</sup>	0 <sup>Note</sup>	0 <sup>Note</sup>	0 <sup>Note</sup>	PMC102	0 <sup>Note</sup>	PMC100	Address FFFFFF054H	After reset 00H

Bit Position	Bit Name	Function
2	PMC102	Port Mode Control Sets operation mode of P102 pin. 0: Input/output port mode 1: TCLR12 input mode
0	PMC100	Port Mode Control Sets operation mode of P100 pin. 0: Input/output port mode 1: TO120 output mode

**Note** Be sure to set these bits to 0.

**12.3.12 Port A**

Port A is an 8-bit input/output port that can be set to input or output in 1-bit units.

PA	7	6	5	4	3	2	1	0	Address FFFFF01CH	After reset Undefined
	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0		

Bit Position	Bit Name	Function
7 to 0	PAn (n = 7 to 0)	Port A Input/output port

In addition to their function as port pins, the port A pins can also operate in the control mode (external expansion mode) as an address bus used when memory is expanded externally.

**(1) Operation in control mode**

Port		Control Mode	Remark	Block Type
Port A	PA0 to PA7	A0 to A7	Address bus in memory expansion	F

**(2) Input/output mode/control mode setting**

Port A input/output mode setting is performed by means of the port A mode register (PMA), and control mode (external expansion mode) setting is performed by means of the mode specification pins (MODE0, MODE2) and the memory expansion mode register (MM: refer to **3.4.6 (1)**).

**(a) Port A mode register (PMA)**

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

PMA	7	6	5	4	3	2	1	0	Address FFFFF03CH	After reset FFH
	PMA7	PMA6	PMA5	PMA4	PMA3	PMA2	PMA1	PMA0		

Bit Position	Bit Name	Function
7 to 0	PMA <sub>n</sub> (n = 7 to 0)	Port Mode Sets PAn pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**(b) Operation mode of port A**

Bit of MM Register				Operation Mode							
MM3	MM2	MM1	MM0	PA0	PA1	PA2	PA3	PA4	PA5	PA6	PA7
don't care	0	0	0	Port (PA0 to PA7)							
	0	0	1	Address bus (A0 to A7)							
	0	1	0								
	0	1	1								
	1	0	0								
	1	0	1								
	1	1	0								
	1	1	1								

For the details of mode selection by the MODE0 and MODE2 pins, refer to **3.3.2 Operating mode specification**.

The MM0 to MM3 bits are initialized to 111× at system reset, enabling the external address output mode. If MM0 to MM3 are set to 000× by the program, the port mode can be changed to, but the subsequent external instruction cannot be fetched from data bus.

**Remark** ×: don't care

**12.3.13 Port B**

Port B is an 8-bit input/output port that can be set to input or output in 1-bit units.

PB	7	6	5	4	3	2	1	0	Address FFFFF01EH	After reset Undefined
	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0		

Bit Position	Bit Name	Function
7 to 0	PBn (n = 7 to 0)	Port B Input/output port

In addition to their function as port pins, the port B pins can also operate in the control mode (external expansion mode) as an address bus used when memory is expanded externally.

**(1) Operation in control mode**

Port	Control Mode	Remark	Block Type
Port B	PB0 to PB7	A8 to A15	Address bus in memory expansion

**(2) Input/output mode/control mode setting**

Port B input/output mode setting is performed by means of the port B mode register (PMB), and control mode (external expansion mode) setting is performed by means of the mode specification pins (MODE0, MODE2) and the memory expansion mode register (MM: refer to **3.4.6 (1)**).

**(a) Port B mode register (PMB)**

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

PMB	7	6	5	4	3	2	1	0	Address FFFFF03EH	After reset FFH
	PMB7	PMB6	PMB5	PMB4	PMB3	PMB2	PMB1	PMB0		

Bit Position	Bit Name	Function
7 to 0	PMBn (n = 7 to 0)	Port Mode Sets PBn pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**(b) Operation mode of port B**

Bit of MM Register				Operation Mode							
MM3	MM2	MM1	MM0	PB0	PB1	PB2	PB3	PB4	PB5	PB6	PB7
don't care	0	0	0	Port (PB0 to PB7)							
	0	0	1	A8	A9	A10	A11	PB4	PB5	PB6	PB7
	0	1	0					A12	A13		
	0	1	1							A14	A15
	1	0	0								
	1	0	1								
	1	1	0								
	1	1	1								

For the details of mode selection by the MODE0 and MODE2 pins, refer to **3.3.2 Operating mode specification**.

The MM0 to MM3 bits are initialized to 111× at system reset, enabling the external address output mode. If MM0 to MM3 are set to 000× by the program, the port mode can be changed to, but the subsequent external instruction cannot be fetched from data bus. Also, if MM0 to MM3 are set to 100x or 010x, the subsequent external address output from port B is disabled.

**Remark** ×: don't care



### 12.3.14 Port X

Port X is a 2-bit input/output port that can be set to input or output in 1-bit units.

PX	7	6	5	4	3	2	1	0	Address FFFFFF41AH	After reset Undefined
	PX7	PX6	—	—	—	—	—	—		

Bit Position	Bit Name	Function
7, 6	PXn (n = 7, 6)	Port X Input/output port

In addition to their function as port pins, the port X pins can also operate as wait control input and internal system clock output in the control mode. Lower 6 bits of port X are always undefined in the case of 8-bit access.

#### (1) Operation in control mode

Port		Control Mode	Remark	Block Type
Port X	PX6	WAIT	Wait control input	D
	PX7	CLKOUT	Internal system clock output	A

#### (2) Input/output mode/control mode setting

Port X input/output mode setting is performed by means of the port X mode register (PMX), and control mode setting is performed by means of the port X mode control register (PMCX).

##### (a) Port X mode register (PMX)

This register is write-only, in 8-bit units. However, the lower 6 bits are fixed at 1 by hardware, so even if 0 is written, it is disregarded.

PMX	7	6	5	4	3	2	1	0	Address FFFFFF43AH	After reset FFH
	PMX7	PMX6	1 <sup>Note</sup>	1	1	1	1	1		

Bit Position	Bit Name	Function
7, 6	PMXn (n = 7, 6)	Port Mode Sets PXn pin in input/output mode. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

**Note** Be sure to set this bit to 1.

**Caution** Do not change the port mode using a bit manipulation instruction (CLR1, NOT1, SET1, TST1).

**(b) Port X mode control register (PMCX)**

This register is write-only, in 8-bit units. However, the lower 5 bits are fixed at 0 by hardware, so even if 1 is written, it is disregarded.

	7	6	5	4	3	2	1	0		
PMCX	PMCX7	PMCX6	1 <sup>Note</sup>	0	0	0	0	0	Address FFFFFF45AH	After reset E0H

Bit Position	Bit Name	Function
7	PMCX7	Port Mode Control Sets operation mode of PX7 pin. 0: Input/output port mode 1: CLKOUT output mode
6	PMCX6	Port Mode Control Sets operation mode of PX6 pin. 0: Input/output port mode 1: WAIT input mode

**Note** Be sure to set this bit to 1.

**Caution** Do not change the operation mode using a bit manipulation instruction (CLR1, NOT1, SET1, TST1).

## CHAPTER 13 RESET FUNCTIONS

When a low-level signal is input to the  $\overline{\text{RESET}}$  pin, a system reset is effected and the hardware is initialized.

When the  $\overline{\text{RESET}}$  signal level changes from low to high, the reset state is released and program execution is started. Register contents must be initialized as required in the program.

### 13.1 Features

The reset pin ( $\overline{\text{RESET}}$ ) incorporates a noise eliminator which uses analog delay ( $\cong 60$  ns) to prevent malfunction due to noise.

### 13.2 Pin Functions

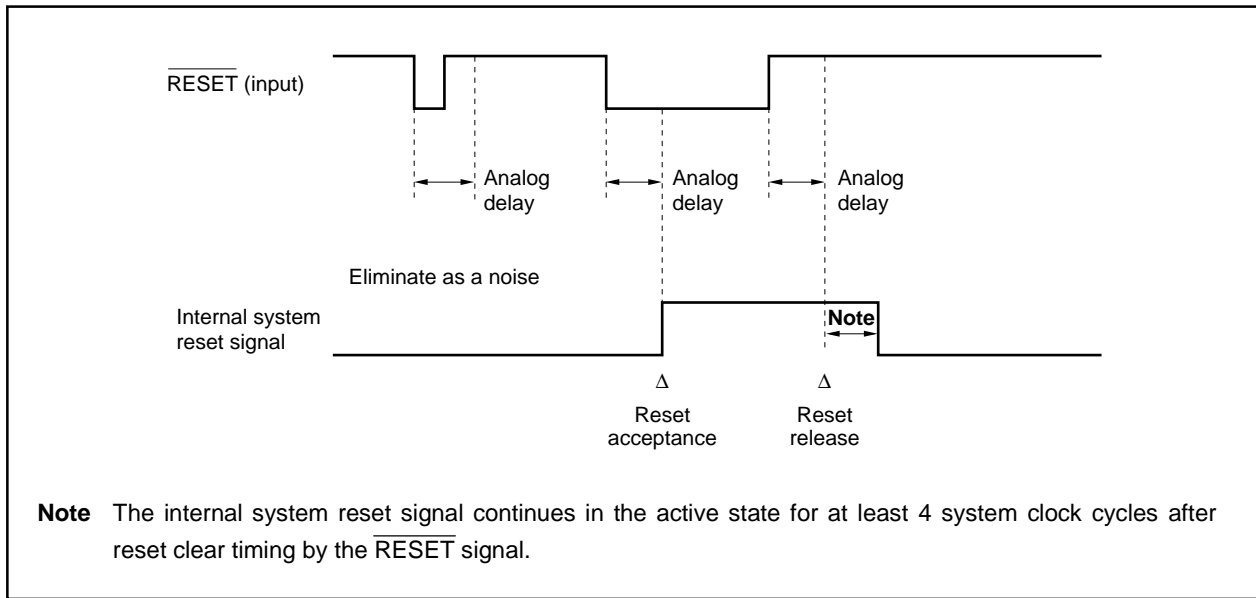
During a system reset, most pins (all but the CLKOUT,  $\overline{\text{RESET}}$ , X2, HVDD, VDD, VSS, CVDD, CVSS, AVDD, AVSS, and AVREF pins) enter the high impedance state. Therefore, when memory is connected externally, a pull-up or pull-down resistor must be connected to the specified pins of ports 4, 5, 6, 8, 9, A, B, and X. If no resistor is connected there, memory contents may be lost when these pins enter high impedance state. For the same reason, the output pins of the internal peripheral I/O functions and output ports should be handled in the same manner.

Table 13-1 shows the operating state of each output pin and each input/output pin during reset.

**Table 13-1. Operating State of Each Pin During Reset**

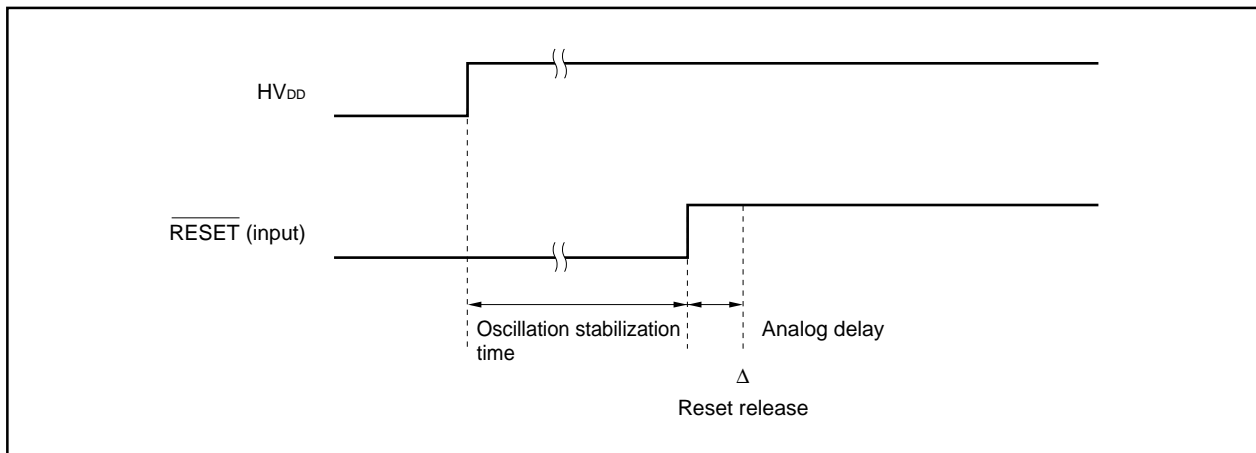
Pin Name		Pin State	
		When in ROM-less Mode 0	When in ROM-less Mode 1
D0 to D7, A0 to A23, $\overline{\text{CS0}}$ , $\overline{\text{CS3}}$ to $\overline{\text{CS5}}$ , $\overline{\text{RAS3}}$ to $\overline{\text{RAS5}}$ , $\overline{\text{LCAS}}$ , $\overline{\text{LWR}}$ , $\overline{\text{UCAS}}$ , $\overline{\text{UWR}}$ , $\overline{\text{RD}}$ , $\overline{\text{WE}}$ , $\overline{\text{BCYST}}$ , $\overline{\text{OE}}$ , $\overline{\text{HLDK}}$		High-impedance	
D8 to D15		High-impedance	(Port mode)
$\overline{\text{WAIT}}$ , $\overline{\text{HLDRQ}}$		(Input)	
CLKOUT		Operating	
Port pin	Ports 0 to 3, 10	(Input)	
	Ports 4, 6, 8, 9, A, B, X	(Control mode)	
	Port 5	(Control mode)	(Input)

### (1) Receiving the reset signal



### (2) Reset during power on

In the reset operation during power on (when the power is turned on), in accordance with the low-level width of the  $\overline{\text{RESET}}$  signal, it is necessary to secure an oscillation stabilization time of 10 ms or greater from power rise to the reception of the reset.



## 13.3 Initialization

The initial values of the CPU, internal RAM and internal peripheral I/O after reset are shown in Table 13-2.

Initialize the contents of each register as necessary during program operation. Particularly, the registers shown below are related to system settings, so set them as necessary.

- Power save control register (PSC): Sets the functions of pins X1 and X2, the operation of the CLKOUT pin, etc.
- Data wait control register (DWC): Sets the number of data wait states.

Table 13-2. Initial Values of CPU, Internal RAM, and Internal Peripheral I/O After Reset (1/2)

Internal Hardware		Register Name	Initial Value After Reset
CPU	Program registers	General-purpose register (r0)	00000000H
		General-purpose registers (r1 to r31)	Undefined
		Program counter (PC)	00000000H
	System registers	Status saving register during interrupt (EIPC, EIPSW)	Undefined
		Status saving register during NMI (FEPC, FEPSW)	Undefined
		Interrupt control register (ECR)	00000000H
		Program status word (PSW)	00000020H
		Status saving register during CALLT execution (CTPC, CTPSW)	Undefined
		Status saving register during exception trap (DBPC, DBPSW)	Undefined
		CALLT base pointer (CTBP)	Undefined
Internal RAM		—	Undefined
Internal peripheral I/O		Command register (PRCMD)	Undefined
	Bus control functions	Data wait control register (DWC1)	FFFFH
		Data wait control register (DWC2)	FFH
		Bus cycle control register (BCC)	5555H
		Bus cycle type configuration register (BCT)	0000H
		Bus size configuration register (BSC)	5555H/0000H
	Memory control functions	DRAM configuration registers (DRC0 to DRC3)	3FC1H
		DRAM type configuration register (DTC)	0000H
		Page ROM configuration register (PRC)	E0H
		Refresh control registers (RFC0 to RFC3)	0000H
		Refresh wait control register (RWC)	00H
	DMA functions	Control registers (DADC0 to DADC3)	0000H
		Source address registers (DSA0H to DSA3H, DSA0L to DSA3L)	Undefined
		Channel control registers (DCHC0 to DCHC3)	00H
		Destination address registers (DDA0H to DDA3H, DDA0L to DDA3L)	Undefined
		Trigger factor registers (DTFR0 to DTFR3)	00H
		Byte count registers (DBC0 to DBC3)	Undefined
		Fly-by transfer data wait control register (FDW)	00H
		DMA disable status register (DDIS)	00H
		DMA restart register (DRST)	00H
	Interrupt/exception control functions	In-service priority register (ISPR)	00H
		External interrupt mode registers (INTM0 to INTM2, INTM4)	00H
		Interrupt control registers (OVIC10 to OVIC13, CMIC40, CMIC41, P10IC0 to P10IC3, P11IC0 to P11IC3, P12IC0 to P12IC3, P13IC0 to P13IC3, DMAIC0 to DMAIC3, CSIC0, CSIC1, SEIC0, STIC0, SRIC0, SRIC1, SEIC1, STIC1, ADIC)	47H

Table 13-2. Initial Values of CPU, Internal RAM, and Internal Peripheral I/O After Reset (2/2)

Internal Hardware		Register Name	Initial Value After Reset
Internal peripheral I/O	Clock generator functions	System status register (SYS)	0000000xB
		Clock control register (CKC)	00H
		Power save control register (PSC)	00H
	Timer/counter functions	Capture/compare registers (CC100 to CC103, CC110 to CC113, CC120 to CC123, CC130 to CC133)	Undefined
		Compare registers (CM40, CM41)	Undefined
		Timer overflow status register (TOVS)	00H
		Timer control register (TMC10 to TMC13, TMC40, TMC41)	00H
		Timer unit mode register (TUM10 to TUM13)	0000H
		Timers (TM10 to TM13, TM40, TM41)	0000H
		Timer output control registers (TOC10 to TOC13)	00H
	Serial interface functions	Asynchronous serial interface status registers (ASIS0, ASIS1)	00H
		Asynchronous serial interface mode registers (ASIM00, ASIM10)	80H
		Asynchronous serial interface mode registers (ASIM01, ASIM11)	00H
		Receive buffers (RXB0, RXB1, RXB0L, RXB1L)	Undefined
		Transmit shift registers (TXS0, TXS1, TXS0L, TXS1L)	Undefined
		Clocked serial interface mode registers (CSIM0, CSIM1)	00H
		Serial I/O shift registers (SIO0, SIO1)	Undefined
		Baud rate generator compare registers (BRGC0, BRGC1)	Undefined
		Baud rate generator prescaler mode registers (BPRM0, BPRM1)	00H
	A/D converters	Mode register (ADM0)	00H
		Mode register (ADM1)	07H
		A/D conversion result registers (ADCR0 to ADCR3, ADCR0H to ADCR3H)	Undefined
	Port functions	Ports (P0 to P10, PA, PB, PX)	Undefined
		Port/control select registers (PCS0, PCS1, PCS8)	00H
		Mode registers (PM0 to PM10, PMA, PMB, PMX)	FFH
		Mode control registers (PMC0, PMC1, PMC3, PMC10)	00H
		Mode control register (PMC2)	01H
		Mode control registers (PMC8, PMC9)	FFH
		Mode control register (PMCX)	E0H
		Memory expansion mode register (MM)	07H/0FH

**Caution** “Undefined” in the above table is undefined during power-on reset, or undefined as a result of data destruction when  $\overline{\text{RESET}}\downarrow$  is input and the data writing timing has been synchronized. For other  $\overline{\text{RESET}}\downarrow$ s, data is held in the same state it was in before the  $\overline{\text{RESET}}\downarrow$  operation.

## APPENDIX A CAUTIONS

### A.1 Restriction on Repeated Execution of sld Instruction

The data that should be loaded may not be transferred correctly to the register when sld instructions accessing external memory are repeatedly executed, and interrupt processing is executed.

#### A.1.1 Details of malfunction

This malfunction occurs when the following conditions are met:

The instruction sequence (1) to (4) is executed, and at (4) the sld instruction is executed repeatedly. The malfunction occurs when an interrupt occurs during the execution of the second sld instruction.

When the malfunction occurs, the data loaded by the sld instruction immediately before the interrupt is incorrectly written to the register.

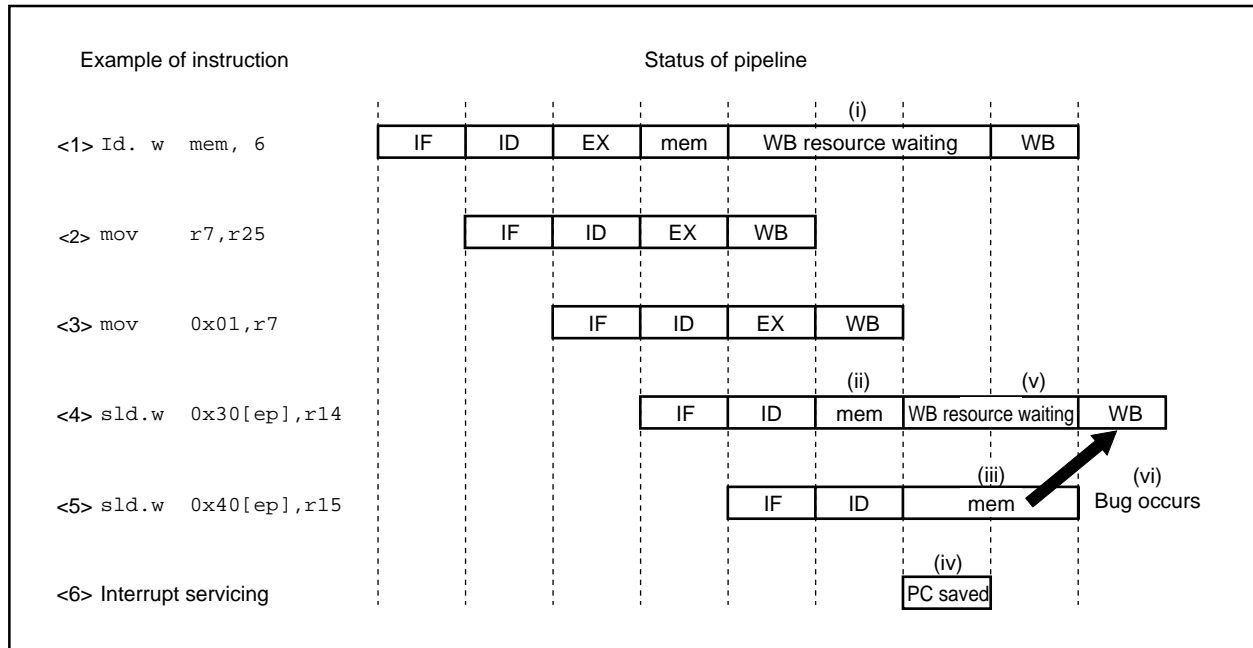
- (1) ld instruction or sld instruction
- (2) One or more of any instruction other than the ld or mul instruction
- (3) One or more instructions that write to a register (refer to **Table A-1**)
- (4) sld instructions (repeated execution of sld instructions where data is loaded from external memory.)  
    sld instruction ← Interrupt occurs.  
    sld instruction  
    :

This malfunction does not occur unless the sequence (1) to (4) is executed. Cases where this malfunction does not occur are shown below:

- The sld instructions in (4) load from internal memory
- The instructions in (4) are load instructions other than sld
- The repeated sld instructions are separated by a branch instruction (such as a jr instruction or a reti instruction)
- An ep (element pointer) setting occurs immediately before the repeated sld instructions
- A nop instruction is executed immediately before the repeated sld instructions
- Interrupts are disabled before executing the sequence (1) to (4).

This error occurs when the following pipeline status occurs.

**Figure A-1. Malfunction When Executing sld Instructions**



- (i) To improve performance with the V850E/MS2, the pipelining of other instructions goes ahead regardless of the dst (destination register) waiting for resources in instruction <1>, and consequently the WB of instructions <2> and <3> are executed before the WB of instruction <1>.
- (ii) The EX stage of instruction <4> is omitted and mem access is executed because address calculation resources are available in the ID stage.
- (iii) Execution of mem access cannot be stopped even if an interrupt is received in the ID stage of instruction <5>, so a dummy access is performed.
- (iv) The PC is saved, and so on, due to the interrupt processing. Since this uses WB resources, the WB of instruction <1> is delayed even more.
- (v) The WB of instruction <4> is made to wait one clock cycle because otherwise it would conflict with the WB of instruction <1>.
- (vi) The data written in the WB stage of instruction <4> is the result of the dummy cycle of the mem stage of instruction <5>, because reading cannot be masked due to the dummy cycle in instruction <5>, and an incorrect write to r14 is performed, causing the malfunction.



**Table A-1. Instructions That Write to a Register Immediately Before the sld Instructions**

Mnemonic	Operand	Mnemonic	Operand
mov	R, r	movea	imm16, R, r
not	R, r	movhi	imm16, R, r
divh	R, r	satsubi	imm16, R, r
satsubr	R, r	mov	Imm32, R
satsub	R, r	ori	imm16, R, r
satadd	R, r	xori	imm16, R, r
zxb	R	andi	imm16, R, r
sxb	R	setf	cccc, r
zxh	R	ldsr	R, sr
sxh	R	stsr	SR, r
or	R, r	shr	R, r
xor	R, r	sar	R, r
and	R, r	shl	R, r
subr	R, r	sasf	cccc, r
sub	R, r	divh	R, r, w
add	R, r	divhu	R, r, w
mov	imm5, r	div	R, r, w
satadd	imm5, r	divu	R, r, w
add	imm5, r	cmov	imm5, r, m
shr	imm5, r	cmov	R, r, w
sar	imm5, r	bsw	r, w
shl	imm5, r	bsh	r, w
addi	imm16, R, r	hsw	r, w

### A.1.2 Countermeasures

#### (1) Assembler

This malfunction can be avoided by using any of the following countermeasures.

- Change the first sld instruction to an ld instruction when repeated sld instructions are used. This malfunction will not occur if all the sld instructions are changed to ld instructions.
- Set ep immediately before the first sld instruction when repeated sld instructions are used.
- Insert a nop instruction immediately before the first sld instruction when repeated sld instructions are used.

#### (2) NEC Compiler

The NEC compiler does not output the instruction sequence that causes this malfunction. Therefore, it is not necessary to take any countermeasures.

**(3) GHS<sup>Note</sup> Compiler**

This malfunction can be avoided using the following two countermeasures so that the compiler does not output repeated sld instructions.

- Specify the following option at compilation:  
    -Z1412
- Avoid using a TDA (Tiny Data Area) function pragma.

**Note** Green Hills Software<sup>TM</sup>, Inc.

## APPENDIX B REGISTER INDEX

(1/6)

Register Symbol	Register Name	Unit	Page
ADCR0	A/D conversion result register 0	ADC	293
ADCR0H	A/D conversion result register 0H	ADC	293
ADCR1	A/D conversion result register 1	ADC	293
ADCR1H	A/D conversion result register 1H	ADC	293
ADCR2	A/D conversion result register 2	ADC	293
ADCR2H	A/D conversion result register 2H	ADC	293
ADCR3	A/D conversion result register 3	ADC	293
ADCR3H	A/D conversion result register 3H	ADC	293
ADIC	Interrupt control register	INTC	189
ADM0	A/D converter mode register 0	ADC	290
ADM1	A/D converter mode register 1	ADC	292
ASIM00	Asynchronous serial interface mode register 00	UART0	259
ASIM01	Asynchronous serial interface mode register 01	UART0	259
ASIM10	Asynchronous serial interface mode register 10	UART1	259
ASIM11	Asynchronous serial interface mode register 11	UART1	259
ASIS0	Asynchronous serial interface status register 0	UART0	263
ASIS1	Asynchronous serial interface status register 1	UART1	263
BCC	Bus cycle control register	BCU	91
BCT	Bus cycle type configuration register	BCU	79
BPRM0	Baud rate generator prescaler mode register 0	BRG0	286
BPRM1	Baud rate generator prescaler mode register 1	BRG1	286
BRGC0	Baud rate generator compare register 0	BRG0	285
BRGC1	Baud rate generator compare register 1	BRG1	285
BSC	Bus size configuration register	BCU	82
CC100	Capture/compare register 100	RPU	224
CC101	Capture/compare register 101	RPU	224
CC102	Capture/compare register 102	RPU	224
CC103	Capture/compare register 103	RPU	224
CC110	Capture/compare register 110	RPU	224
CC111	Capture/compare register 111	RPU	224
CC112	Capture/compare register 112	RPU	224
CC113	Capture/compare register 113	RPU	224
CC120	Capture/compare register 120	RPU	224
CC121	Capture/compare register 121	RPU	224
CC122	Capture/compare register 122	RPU	224
CC123	Capture/compare register 123	RPU	224

Register Symbol	Register Name	Unit	Page
CC130	Capture/compare register 130	RPU	224
CC131	Capture/compare register 131	RPU	224
CC132	Capture/compare register 132	RPU	224
CC133	Capture/compare register 133	RPU	224
CKC	Clock control register	CG	205
CM40	Compare register 40	RPU	225
CM41	Compare register 41	RPU	225
CMIC40	Interrupt control register	INTC	189
CMIC41	Interrupt control register	INTC	189
CSIC0	Interrupt control register	INTC	189
CSIC1	Interrupt control register	INTC	189
CSIM0	Clocked serial interface mode register 0	CSI0	273
CSIM1	Clocked serial interface mode register 1	CSI1	273
CTBP	CALLT base pointer	CPU	52
CTPC	Status saving register during CALLT execution	CPU	52
CTPSW	Status saving register during CALLT execution	CPU	52
DADC0	DMA addressing control register 0	DMAC	142
DADC1	DMA addressing control register 1	DMAC	142
DADC2	DMA addressing control register 2	DMAC	142
DADC3	DMA addressing control register 3	DMAC	142
DBC0	DMA byte count register 0	DMAC	141
DBC1	DMA byte count register 1	DMAC	141
DBC2	DMA byte count register 2	DMAC	141
DBC3	DMA byte count register 3	DMAC	141
DBPC	Status saving register during exception trap	CPU	52
DBPSW	Status saving register during exception trap	CPU	52
DCHC0	DMA channel control register 0	DMAC	144
DCHC1	DMA channel control register 1	DMAC	144
DCHC2	DMA channel control register 2	DMAC	144
DCHC3	DMA channel control register 3	DMAC	144
DDA0H	DMA destination address register 0H	DMAC	139
DDA0L	DMA destination address register 0L	DMAC	140
DDA1H	DMA destination address register 1H	DMAC	139
DDA1L	DMA destination address register 1L	DMAC	140
DDA2H	DMA destination address register 2H	DMAC	139
DDA2L	DMA destination address register 2L	DMAC	140
DDA3H	DMA destination address register 3H	DMAC	139
DDA3L	DMA destination address register 3L	DMAC	140
DDIS	DMA disable status register	BCU	147

Register Symbol	Register Name	Unit	Page
DMAIC0	Interrupt control register	INTC	189
DMAIC1	Interrupt control register	INTC	189
DMAIC2	Interrupt control register	INTC	189
DMAIC3	Interrupt control register	INTC	189
DRC0	DRAM configuration register 0	BCU	113
DRC1	DRAM configuration register 1	BCU	113
DRC2	DRAM configuration register 2	BCU	113
DRC3	DRAM configuration register 3	BCU	113
DRST	DMA restart register	BCU	147
DSA0H	DMA source address register 0H	DMAC	137
DSA0L	DMA source address register 0L	DMAC	138
DSA1H	DMA source address register 1H	DMAC	137
DSA1L	DMA source address register 1L	DMAC	138
DSA2H	DMA source address register 2H	DMAC	137
DSA2L	DMA source address register 2L	DMAC	138
DSA3H	DMA source address register 3H	DMAC	137
DSA3L	DMA source address register 3L	DMAC	138
DTC	DRAM type configuration register	BCU	116
DTFR0	DMA trigger factor register 0	DMAC	145
DTFR1	DMA trigger factor register 1	DMAC	145
DTFR2	DMA trigger factor register 2	DMAC	145
DTFR3	DMA trigger factor register 3	DMAC	145
DWC1	Data wait control register 1	BCU	87
DWC2	Data wait control register 2	BCU	87
ECR	Interrupt source register	CPU	52
EIPC	Status saving register during interrupt	CPU	52
EIPSW	Status saving register during interrupt	CPU	52
FDW	Flyby transfer data wait control register	BCU	148
FEPC	Status saving register during NMI	CPU	52
FEPSW	Status saving register during NMI	CPU	52
INTM0	External interrupt mode register 0	INTC	181
INTM1	External interrupt mode register 1	INTC	193
INTM2	External interrupt mode register 2	INTC	193
INTM4	External interrupt mode register 4	INTC	193
ISPR	In-service priority register	INTC	191
MM	Memory expansion mode register	Port	63
OVIC10	Interrupt control register	INTC	189
OVIC11	Interrupt control register	INTC	189
OVIC12	Interrupt control register	INTC	189

Register Symbol	Register Name	Unit	Page
OVIC13	Interrupt control register	INTC	189
P0	Port 0	Port	328
P1	Port 1	Port	331
P2	Port 2	Port	334
P3	Port 3	Port	337
P4	Port 4	Port	339
P5	Port 5	Port	341
P6	Port 6	Port	343
P7	Port 7	Port	345
P8	Port 8	Port	346
P9	Port 9	Port	350
P10	Port 10	Port	353
P10IC0	Interrupt control register	INTC	189
P10IC1	Interrupt control register	INTC	189
P10IC2	Interrupt control register	INTC	189
P10IC3	Interrupt control register	INTC	189
P11IC0	Interrupt control register	INTC	189
P11IC1	Interrupt control register	INTC	189
P11IC2	Interrupt control register	INTC	189
P11IC3	Interrupt control register	INTC	189
P12IC0	Interrupt control register	INTC	189
P12IC1	Interrupt control register	INTC	189
P12IC2	Interrupt control register	INTC	189
P12IC3	Interrupt control register	INTC	189
P13IC0	Interrupt control register	INTC	189
P13IC1	Interrupt control register	INTC	189
P13IC2	Interrupt control register	INTC	189
P13IC3	Interrupt control register	INTC	189
PA	Port A	Port	355
PB	Port B	Port	357
PC	Program counter	CPU	51
PCS0	Port/control select register 0	Port	330
PCS1	Port/control select register 1	Port	333
PCS8	Port/control select register 8	Port	349
PM0	Port 0 mode register	Port	328
PM1	Port 1 mode register	Port	331
PM2	Port 2 mode register	Port	335
PM3	Port 3 mode register	Port	338
PM4	Port 4 mode register	Port	340

Register Symbol	Register Name	Unit	Page
PM5	Port 5 mode register	Port	342
PM6	Port 6 mode register	Port	344
PM8	Port 8 mode register	Port	347
PM9	Port 9 mode register	Port	351
PM10	Port 10 mode register	Port	353
PMA	Port A mode register	Port	355
PMB	Port B mode register	Port	357
PMC0	Port 0 mode control register	Port	329
PMC1	Port 1 mode control register	Port	332
PMC2	Port 2 mode control register	Port	336
PMC3	Port 3 mode control register	Port	338
PMC8	Port 8 mode control register	Port	348
PMC9	Port 9 mode control register	Port	352
PMC10	Port 10 mode control register	Port	354
PMCX	Port X mode control register	Port	360
PMX	Port X mode register	Port	359
PRC	Page ROM configuration register	BCU	108
PRCMD	Command register	CPU	74
PSC	Power save control register	CPU	209
PSW	Program status word	CPU	53
PX	Port X	Port	359
r0 to r31	General-purpose register	CPU	51
RFC0	Refresh control register 0	BCU	127
RFC1	Refresh control register 1	BCU	127
RFC2	Refresh control register 2	BCU	127
RFC3	Refresh control register 3	BCU	127
RWC	Refresh wait control register	BCU	130
RXB0	Receive buffer 0 (9 bits)	UART0	264
RXB0L	Receive buffer 0L (Lower order 8 bits)	UART0	264
RXB1	Receive buffer 1 (9 bits)	UART1	264
RXB1L	Receive buffer 1L (Lower order 8 bits)	UART1	264
SEIC0	Interrupt control register	INTC	189
SEIC1	Interrupt control register	INTC	189
SIO0	Serial I/O shift register 0	CSI0	275
SIO1	Serial I/O shift register 1	CSI1	275
SRIC0	Interrupt control register	INTC	189
SRIC1	Interrupt control register	INTC	189
STIC0	Interrupt control register	INTC	189
STIC1	Interrupt control register	INTC	189

Register Symbol	Register Name	Unit	Page
SYS	System status register	CPU	75
TM10	Timer 10	RPU	223
TM11	Timer 11	RPU	223
TM12	Timer 12	RPU	223
TM13	Timer 13	RPU	223
TM40	Timer 40	RPU	225
TM41	Timer 41	RPU	225
TMC10	Timer control register 10	RPU	228
TMC11	Timer control register 11	RPU	228
TMC12	Timer control register 12	RPU	228
TMC13	Timer control register 13	RPU	228
TMC40	Timer control register 40	RPU	230
TMC41	Timer control register 41	RPU	230
TOC10	Timer output control register 10	RPU	231
TOC11	Timer output control register 11	RPU	231
TOC12	Timer output control register 12	RPU	231
TOVS	Timer overflow status register	RPU	232
TUM10	Timer unit mode register 10	RPU	226
TUM11	Timer unit mode register 11	RPU	226
TUM12	Timer unit mode register 12	RPU	226
TUM13	Timer unit mode register 13	RPU	226
TXS0	Transmit shift register 0 (9 bits)	UART0	265
TXS0L	Transmit shift register 0L (Lower order 8 bits)	UART0	265
TXS1	Transmit shift register 1 (9 bits)	UART1	265
TXS1L	Transmit shift register 1L (Lower order 8 bits)	UART1	265



## APPENDIX C INSTRUCTION SET LIST

### C.1 General Examples

#### (1) Register symbols used to describe operands

Register Symbol	Explanation
reg1	General-purpose registers (r0 to r31): Used as source registers.
reg2	General-purpose registers (r0 to r31): Used mainly as destination registers.
reg3	General-purpose registers (r0 to r31): Used mainly to store the remainders of division results and the higher 3 bits of multiplication results.
immX	X bit immediate
dispX	X bit displacement
regID	System register number
bit#3	3-bit data for specifying the bit number
ep	Element pointer (r30)
cccc	4-bit data which show the conditions code
vector	5-bit data which specify the trap vector (00H to 1FH)
listX	X item register list

#### (2) Register symbols used to describe op codes

Register Symbol	Explanation
R	1-bit data of a code which specifies reg1 or regID
r	1-bit data of the code which specifies reg2
w	1-bit data of the code which specifies reg3
d	1-bit displacement data
i	1-bit immediate data
cccc	4-bit data which show the conditions code
bbb	3-bit data for specifying the bit number
L	1-bit data which specifies a register list

#### (3) Register symbols used in operation (1/2)

Register Symbol	Explanation
←	Input for
GR [ ]	General-purpose register
SR [ ]	System register
zero-extend (n)	Expand n with zeros until word length.
sign-extend (n)	Expand n with signs until word length.
load-memory (a, b)	Read data from address a until size b.
store-memory (a, b, c)	Write data b in address a to size c.
load-memory-bit (a, b)	Read bit b of address a.

**(3) Register symbols used in operation (2/2)**

Register Symbol	Explanation
store-memory-bit (a, b, c)	Write bit b of address a to c.
saturated (n)	Execute saturated processing of n (n is a 2's complement). If, as a result of calculations, $n \geq 7FFFFFFFH$ , let it be $7FFFFFFFH$ . $n \leq 80000000H$ , let it be $80000000H$ .
result	Reflects the results in a flag.
Byte	Byte (8 bits)
Half-word	Half word (16 bits)
Word	Word (32 bits)
+	Addition
–	Subtraction
	Bit concatenation
×	Multiplication
÷	Division
%	Remainder from division results
AND	Logical product
OR	Logical sum
XOR	Exclusive OR
NOT	Logical negation
logically shift left by	Logical shift left
logically shift right by	Logical shift right
arithmetically shift right by	Arithmetic shift right

**(4) Register symbols used in an execution clock**

Register Symbol	Explanation
i : issue	If executing another instruction immediately after executing the first instruction.
r : repeat	If repeating execution of the same instruction immediately after executing the first instruction.
l : latency	If referring to the results of instruction execution immediately after execution using another instruction.

**(5) Register symbols used in flag operations**

Identifier	Explanation
(Blank)	No change
0	Clear to 0
X	Set or cleared in accordance with the results.
R	Previously saved values are restored.

**(6) Condition codes**

Condition Name (cond)	Condition Code (cccc)	Condition Formula	Explanation
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 Not 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	$(CY \text{ or } Z) = 1$	Not higher (Less than or equal)
H	1 0 1 1	$(CY \text{ or } Z) = 0$	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	$(S \text{ xor } OV) = 1$	Less than signed
GE	1 1 1 0	$(S \text{ xor } OV) = 0$	Greater than or equal signed
LE	0 1 1 1	$((S \text{ xor } OV) \text{ or } Z) = 1$	Less than or equal signed
GT	1 1 1 1	$((S \text{ xor } OV) \text{ or } Z) = 0$	Greater than signed

## C.2 Instruction Set (in Alphabetical Order)

(1/6)

Mnemonic	Operand	Op Code	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
ADD	reg1,reg2	rrrrr001110RRRRR	GR[reg2]←GR[reg2]+GR[reg1]	1	1	1	x	x	x	x	
	imm5,reg2	rrrrr010010iiii	GR[reg2]←GR[reg2]+sign-extend(imm5)	1	1	1	x	x	x	x	
ADDI	imm16,reg1,reg2	rrrrr110000RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+sign-extend(imm16)	1	1	1	x	x	x	x	
AND	reg1,reg2	rrrrr001010RRRRR	GR[reg2]←GR[reg2]AND GR[reg1]	1	1	1		0	x	x	
ANDI	imm16,reg1,reg2	rrrrr110110RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]AND zero-extend(imm16)	1	1	1		0	0	x	
Bcond	disp9	dddd1011ddcccc <b>Note 1</b>	if conditions are satisfied then PC←PC+sign-extend(disp9)	2	2	2					
			When conditions are satisfied	<b>Note 2</b>	<b>Note 2</b>	<b>Note 2</b>					
			When conditions are not satisfied	1	1	1					
BSH	reg2,reg3	rrrrr11111100000 wwwww01101000010	GR[reg3]←GR[reg2] (23 : 16)    GR[reg2] (31 : 24)    GR[reg2] (7 : 0)    GR[reg2] (15 : 8)	1	1	1	x	0	x	x	
BSW	reg2,reg3	rrrrr11111100000 wwwww01101000000	GR[reg3]←GR[reg2] (7 : 0)    GR[reg2] (15 : 8)    GR [reg2] (23 : 16)    GR[reg2] (31 : 24)	1	1	1	x	0	x	x	
CALLT	imm6	0000001000iiii	CTPC←PC+2(return PC) CTPSW←PSW adr←CTBP+zero-extend(imm6 logically shift left by 1) PC←CTBP+zero-extend(Load-memory(adrs,Half-word))	4	4	4					
CLR1	bit#3, disp16[reg1]	10bbb111110RRRRR dddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adrs,bit#3)) Store-memory-bit(adrs,bit#3,0)	3	3	3				x	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100100	adr←GR[reg1] Z flag←Not(Load-memory-bit(adrs,reg2)) Store-memory-bit(adrs,reg2,0)	3	3	3				x	
CMOV	cccc,imm5,reg2,reg3	rrrrr111111iiii wwwww01100cccc0	if conditions are satisfied then GR[reg3]←sign-extended(imm5) else GR[reg3]←GR[reg2]	1	1	1					
	cccc,reg1,reg2,reg3	rrrrr111111RRRRR wwwww011001cccc0	if conditions are satisfied then GR[reg3]←GR[reg1] else GR[reg3]←GR[reg2]	1	1	1					
CMP	reg1,reg2	rrrrr001111RRRRR	result←GR[reg2]−GR[reg1]	1	1	1	x	x	x	x	
	imm5,reg2	rrrrr010011iiii	result←GR[reg2]−sign-extend(imm5)	1	1	1	x	x	x	x	
CTRET		0000011111100000 0000000101000100	PC←CTPC PSW←CTPSW	3	3	3	R	R	R	R	R
DI		0000011111100000 0000000101100000	PSW.ID←1	1	1	1					

Mnemonic	Operand	Op Code	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
DISPOSE	imm5,list12	0000011001iiiiL LLLLLLLLLLLL00000	sp←sp+zero-extend(imm5 logically shift left by 2) GR[reg in list12]←Load-memory(sp,Word) sp←sp+4 repeat 2 steps above until all regs in list12 is loaded	N+1 Note 4	N+1 Note 4	N+1 Note 4					
	imm5,list12,[reg1]	0000011001iiiiL LLLLLLLLLLLLRRRRR <b>Note 5</b>	sp←sp+zero-extend(imm5 logically shift left by 2) GR[reg in list12]←Load-memory(sp,Word) sp←sp+4 repeat 2 steps above until all regs in list12 is loaded PC←GR[reg1]	N+3 Note 4	N+3 Note 4	N+3 Note 4					
DIV	reg1,reg2,reg3	rrrrr11111RRRRR wwwww01011000000	GR[reg2]←GR[reg2]÷GR[reg1] GR[reg3]←GR[reg2]%GR[reg1]	35	35	35		×	×	×	
DIVH	reg1,reg2	rrrrr000010RRRRR	GR[reg2]←GR[reg2]÷GR[reg1] <sup>Note 6</sup>	35	35	35		×	×	×	
	reg1,reg2,reg3	rrrrr11111RRRRR wwwww01010000000	GR[reg2]←GR[reg2]÷GR[reg1] <sup>Note 6</sup> GR[reg3]←GR[reg2]%GR[reg1]	35	35	35		×	×	×	
DIVHU	reg1,reg2,reg3	rrrrr11111RRRRR wwwww01010000010	GR[reg2]←GR[reg2]÷GR[reg1] <sup>Note 6</sup> GR[reg3]←GR[reg2]%GR[reg1]	34	34	34		×	×	×	
DIVU	reg1,reg2,reg3	rrrrr11111RRRRR wwwww01011000010	GR[reg2]←GR[reg2]÷GR[reg1] GR[reg3]←GR[reg2]%GR[reg1]	34	34	34		×	×	×	
EI		100001111100000 0000000101100000	PSW.ID←0	1	1	1					
HALT		000001111100000 0000000100100000	Stop	1	1	1					
HSW	reg2,reg3	rrrrr1111100000 wwwww01101000100	GR[reg3]←GR[reg2](15 : 0)    GR[reg2] (31 : 16)	1	1	1	×	0	×	×	
JARL	disp22,reg2	rrrrr11110ddddd ddddddddddddddd0 <b>Note 7</b>	GR[reg2]←PC+4 PC←PC+sign-extend(disp22)	2	2	2					
JMP	[reg1]	00000000011RRRRR	PC←GR[reg1]	3	3	3					
JR	disp22	0000011110ddddd ddddddddddddddd0 <b>Note 7</b>	PC←PC+sign-extend(disp22)	2	2	2					
LD.B	disp16[reg1],reg2	rrrrr111000RRRRR ddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	n Note 9					
LD.BU	disp16[reg1],reg2	rrrrr11110bRRRRR ddddddddddddddd1 <b>Notes 8, 10</b>	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←zero-extend(Load-memory(adr,Byte))	1	1	n Note 11					
LD.H	disp16[reg1],reg2	rrrrr111001RRRRR ddddddddddddddd0 <b>Note 8</b>	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Half-word))	1	1	n Note 9					

Mnemonic	Operand	Op Code	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
LD.HU	disp16[reg1],reg2	rrrrr111111RRRRR dddddddddddddd1 <b>Note 8</b>	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←zero-extend(Load-memory(adrr,Half-word))	1	1	n					
LD.W	disp16[reg1],reg2	rrrrr111001RRRRR dddddddddddddd1 <b>Note 8</b>	adr←GR[reg1]+sign-exend(disp16) GR[reg2]←Load-memory(adrr,Word)	1	1	n					
LDSR	reg2,regID	rrrrr111111RRRRR 0000000000100000 <b>Note 12</b>	SR[regID]←GR[reg2] Other than regID=PSW regID=PSW	1	1	1					
MOV	reg1,reg2	rrrrr000000RRRRR	GR[reg2]←GR[reg1]	1	1	1					
	imm5,reg2	rrrrr010000iiii	GR[reg2]←sign-extend(imm5)	1	1	1					
	imm32,reg1	00000110001RRRRR iiiiiiiiiiiiiiii iiiiiiiiiiiiiiii	GR[reg1]←imm32	2	2	2					
MOVEA	imm16,reg1,reg2	rrrrr110001RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+sign-extend(imm16)	1	1	1					
MOVHI	imm16,reg1,reg2	rrrrr110010RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+(imm16 ll 0 <sup>16</sup> )	1	1	1					
MUL	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100000	GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1]	1	2	2					
	imm9,reg2,reg3	rrrrr111111iiii wwwww01001111100 <b>Note 13</b>	GR[reg3] ll GR[reg2]←GR[reg2]xsign-extend(imm9) <b>Note 13</b>	1	2	2					
MULH	reg1,reg2	rrrrr000111RRRRR	GR[reg2]←GR[reg2] <sup>Note 6</sup> xGR[reg1] <sup>Note 6</sup>	1	1	2					
	imm5,reg2	rrrrr010111iiii	GR[reg2]←GR[reg2] <sup>Note 6</sup> xsign-extend(imm5)	1	1	2					
MULHI	imm16,reg1,reg2	rrrrr110111RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] <sup>Note 6</sup> ximm16	1	1	2					
MULU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100010	GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1]	1	2	2					
	imm9,reg2,reg3	rrrrr111111iiii wwwww0100111110 <b>Note 13</b>	GR[reg3] ll GR[reg2]←GR[reg2]xzero-extend(imm9) <b>Note 13</b>	1	2	2					
NOP		0000000000000000	Pass at least one clock cycle doing nothing.	1	1	1					
NOT	reg1,reg2	rrrrr000001RRRRR	GR[reg2]←NOT(GR[reg1])	1	1	1		0	x	x	
NOT1	bit#3,disp16[reg1]	01bbb111110RRRRR dddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adrr,bit#3)) Store-memory-bit(adrr,bit#3,Z flag)	3	3	3				x	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100010	adr←GR[reg1] Z flag←Not(Load-memory-bit(adrr,reg2)) Store-memory-bit(adrr,reg2,Z flag)	3	3	3				x	
OR	reg1,reg2	rrrrr001000RRRRR	GR[reg2]←GR[reg2]OR GR[reg1]	1	1	1		0	x	x	

Mnemonic	Operand	Op Code	Operation	Execution Clock			Flags				
				i	r	I	CY	OV	S	Z	SAT
ORI	imm16,reg1,reg2	rrrrr110100RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]OR zero-extend(imm16)	1	1	1		0	×	×	
PREPARE	list12,imm5	0000011110iiiiL LLLLLLLLLLLL00001	Store-memory(sp−4,GR[reg in list12],Word) sp←sp−4 repeat 1 step above until all regs in list12 is stored sp←sp-zero-extend(imm5)	N+1 Note 4	N+1 Note 4	N+1 Note 4					
	list12,imm5, sp/imm <sup>Note 15</sup>	0000011110iiiiL LLLLLLLLLLLLff011 imm16/imm32 <b>Note 16</b>	Store-memory(sp−4,GR[reg in list12],Word) sp←sp−4 repeat 1 step above until all regs in list12 is stored sp←sp-zero-extend(imm5) ep←sp/imm	N+2 Note 4 Note 17	N+2 Note 4 Note 17	N+2 Note 4 Note 17					
RETI		0000011111100000 0000000101000000	if PSW.EP=1 then PC ←EIPC PSW ←EIPSW else if PSW.NP=1 then PC ←FEPC PSW ←FEPSW else PC ←EIPC PSW ←EIPSW	3	3	3	R	R	R	R	R
SAR	reg1,reg2	rrrrr111111RRRRR 0000000010100000	GR[reg2]←GR[reg2]arithmetically shift right by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010101iiii	GR[reg2]←GR[reg2]arithmetically shift right by zero-extend (imm5)	1	1	1	×	0	×	×	
SASF	cccc,reg2	rrrrr1111110cccc 0000001000000000	if conditions are satisfied then GR[reg2]←(GR[reg2]Logically shift left by 1) OR 00000001H else GR[reg2]←(GR[reg2]Logically shift left by 1) OR 00000000H	1	1	1					
SATADD	reg1,reg2	rrrrr000110RRRRR	GR[reg2]←saturated(GR[reg2]+GR[reg1])	1	1	1	×	×	×	×	×
	imm5,reg2	rrrrr010001iiii	GR[reg2]←saturated(GR[reg2]+sign-extend(imm5))	1	1	1	×	×	×	×	×
SATSUB	reg1,reg2	rrrrr000101RRRRR	GR[reg2]←saturated(GR[reg2]−GR[reg1])	1	1	1	×	×	×	×	×
SATSUBI	imm16,reg1,reg2	rrrrr110011RRRRR iiiiiiiiiiiiiiii	GR[reg2]←saturated(GR[reg1]−sign-extend(imm16))	1	1	1	×	×	×	×	×
SATSUBR	reg1,reg2	rrrrr000100RRRRR	GR[reg2]←saturated(GR[reg1]−GR[reg2])	1	1	1	×	×	×	×	×
SETF	cccc,reg2	rrrrr1111110cccc 0000000000000000	If conditions are satisfied then GR[reg2]←00000001H else GR[reg2]←00000000H	1	1	1					

Mnemonic	Operand	Op Code	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
SET1	bit#3,disp16[reg1]	00bbb111110RRRRR dddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not (Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,1)	3 Note 3	3 Note 3	3 Note 3				x	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100000	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,1)	3 Note 3	3 Note 3	3 Note 3				x	
SHL	reg1,reg2	rrrrr111111RRRRR 0000000011000000	GR[reg2]←GR[reg2] logically shift left by GR[reg1]	1	1	1	x	0	x	x	
	imm5,reg2	rrrrr010110iiii	GR[reg2]←GR[reg2] logically shift left by zero-extend(imm5)	1	1	1	x	0	x	x	
SHR	reg1,reg2	rrrrr111111RRRRR 0000000010000000	GR[reg2]←GR[reg2] logically shift right by GR[reg1]	1	1	1	x	0	x	x	
	imm5,reg2	rrrrr010100iiii	GR[reg2]←GR[reg2] logically shift right by zero-extend(imm5)	1	1	1	x	0	x	x	
SLD.B	disp7[ep],reg2	rrrrr0110dddddd	adr←ep+zero-extend(disp7) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	n Note 9					
SLD.BU	disp4[ep],reg2 Note 18	rrrrr0000110dddd	adr←ep+zero-extend(disp4) GR[reg2]←zero-extend(Load-memory(adr,Byte))	1	1	n Note 9					
SLD.H	disp8[ep],reg2	rrrrr1000dddddd Note 19	adr←ep+zero-extend(disp8) GR[reg2]←sign-extend(Load-memory(adr,Half-word))	1	1	n Note 9					
SLD.HU	disp5[ep],reg2 Notes 18, 20	rrrrr0000111dddd	adr←ep+zero-extend(disp5) GR[reg2]←zero-extend(Load-memory(adr,Half-word))	1	1	n Note 9					
SLD.W	disp8[ep],reg2	rrrrr1010dddddd0 Note 21	adr←ep+zero-extend(disp8) GR[reg2]←Load-memory(adr,Word)	1	1	n Note 9					
SST.B	reg2,disp7[ep]	rrrrr0111dddddd	adr←ep+zero-extend(disp7) Store-memory(adr,GR[reg2],Byte)	1	1	1					
SST.H	reg2,disp8[ep]	rrrrr1001dddddd Note 19	adr←ep+zero-extend(disp8) Store-memory(adr,GR[reg2],Half-word)	1	1	1					
SST.W	reg2,disp8[ep]	rrrrr1010dddddd1 Note 21	adr←ep+zero-extend(disp8) Store-memory(adr,GR[reg2],Word)	1	1	1					
ST.B	reg2,disp16[reg1]	rrrrr111010RRRRR dddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Store-memory(adr,GR[reg2],Byte)	1	1	1					
ST.H	reg2,disp16[reg1]	rrrrr111011RRRRR dddddddddddddd0 Note 8	adr←GR[reg1]+sign-extend(disp16) Store-memory (adr,GR[reg2], Half-word)	1	1	1					
ST.W	reg2,disp16[reg1]	rrrrr111011RRRRR dddddddddddddd1 Note 8	adr←GR[reg1]+sign-extend(disp16) Store-memory (adr,GR[reg2], Word)	1	1	1					
STSR	regID,reg2	rrrrr111111RRRRR 000000001000000	GR[reg2]←SR[regID]	1	1	1					



(6/6)

Mnemonic	Operand	Op Code	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
SUB	reg1,reg2	rrrrr001101RRRRR	GR[reg2]←GR[reg2]−GR[reg1]	1	1	1	×	×	×	×	
SUBR	reg1,reg2	rrrrr001100RRRRR	GR[reg2]←GR[reg1]−GR[reg2]	1	1	1	×	×	×	×	
SWITCH	reg1	0000000010RRRRR	adr←(PC+2) + (GR [reg1] logically shift left by 1) PC←(PC+2) + (sign-extend (Load-memory (adr,Half-word))) logically shift left by 1	5	5	5					
SXB	reg1	00000000101RRRRR	GR[reg1]←sign-extend (GR[reg1] (7 : 0))	1	1	1					
SXH	reg1	00000000111RRRRR	GR[reg1]←sign-extend (GR[reg1] (15 : 0))	1	1	1					
TRAP	vector	000001111111iiii 0000000100000000	EIPC ←PC+4 (Return PC) EIPSW ←PSW ECR.EICC ←Interrupt Code PSW.EP ←1 PSW.ID ←1 PC ←00000040H (when vector is 00H to 0FH) 00000050H (when vector is 10H to 1FH)	3	3	3					
TST	reg1,reg2	rrrrr001011RRRRR	result←GR[reg2] AND GR[reg1]	1	1	1		0	×	×	
TST1	bit#3,disp16[reg1]	11bbb111110RRRRR dddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not (Load-memory-bit (adr,bit#3))	3	3	3	Note 3	Note 3	Note 3		×
	reg2, [reg1]	rrrrr111111RRRRR 0000000011100110	adr←GR[reg1] Z flag←Not (Load-memory-bit (adr,reg2))	3	3	3	Note 3	Note 3	Note 3		×
XOR	reg1,reg2	rrrrr001001RRRRR	GR[reg2]←GR[reg2] XOR GR[reg1]	1	1	1		0	×	×	
XORI	imm16,reg1,reg2	rrrrr110101RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] XOR zero-extend (imm16)	1	1	1		0	×	×	
ZXB	reg1	00000000100RRRRR	GR[reg1]←zero-extend (GR[reg1] (7 : 0))	1	1	1					
ZXH	reg1	00000000110RRRRR	GR[reg1]←zero-extend (GR[reg1] (15 : 0))	1	1	1					

- Notes**
1. dddddddd: Higher 8 bits of disp9.
  2. 3 clocks if the final instruction includes PSW write access.
  3. If there is no wait state (3 + the number of read access wait states).
  4. N is the total number of list 12 read registers. (According to the number of wait states. Also, if there are no wait states, N is the number of list 12 registers.)
  5. RRRRR: other than 00000.
  6. The lower halfword data only are valid.
  7. dddddddddddddddddd: The higher 21 bits of disp22.
  8. dddddddddddddddd: The higher 15 bits of disp16.
  9. According to the number of wait states (1 if there are no wait states).
  10. b: bit 0 of disp16.
  11. According to the number of wait states (2 if there are no wait states).

**Notes** 12. In this instruction, for convenience of mnemonic description, the source register is made reg2, but the reg1 field is used in the op code. Therefore, the meaning of register specification in the mnemonic description and in the op code differs from other instructions.

rrrrr = regID specification

RRRRR = reg2 specification

13. i i i i: Lower 5 bits of imm9.

l l l l: Lower 4 bits of imm9.

14. In the case of r = w (the lower 32 bits of the results are not written in the register) or w = r0 (the higher 32 bits of the results are not written in the register), 1.

15. sp/imm: specified by bits 19 and 20 of the sub op code.

16. ff = 00: Load sp in ep.

01: Load sign expanded 16-bit immediate data (bits 47 to 32) in ep.

10: Load 16-bit logically left shifted 16-bit immediate data (bits 47 to 32) in ep.

11: Load 32-bit immediate data (bits 63 to 32) in ep.

17. If imm = imm32, N + 3 blocks.

18. rrrrr: Other than 00000.

19. dddddd: Higher 7 bits of disp8.

20. dddd: Higher 4 bits of disp5.

21. dddddd: Higher 6 bits of disp8.

## APPENDIX D INDEX

### [A]

A/D conversion result registers .....	293
A/D converter .....	287
A/D converter mode register 0 .....	290
A/D converter mode register 1 .....	292
A/D trigger mode .....	296
A0 to A7 .....	43
A8 to A15 .....	44
A16 to A23 .....	39
ADn0 to ADn9 (n = 0 to 3) .....	293
ADCR0 to ADCR3 .....	293
ADCR0H to ADCR3H .....	293
Address multiplex function .....	112
Address space .....	55
ADIC .....	189
ADIF .....	189
ADM0 .....	290
ADM1 .....	292
ADMK .....	189
ADPR0 to ADPR2 .....	189
ALV1n0 (n = 0 to 2) .....	231
ANI0 to ANI3 .....	40
ANIS0, ANIS1 .....	290
Applications .....	22
ASIM00, ASIM01, ASIM10, ASIM11 .....	259
ASIS0, ASIS1 .....	263
Assembler-reserved register .....	51
Asynchronous serial interfaces 0, 1 .....	256
Asynchronous serial interface mode registers 00, 01, 10, 11 .....	259
Asynchronous serial interface status registers 0, 1 .....	263
AVDD .....	45
AVREF .....	45
AVSS .....	45

### [B]

Basic operation of A/D converter .....	295
Baud rate generator compare registers 0, 1 .....	285
Baud rate generator prescaler mode registers 0, 1 .....	286
BC0 to BC15 .....	141
BCC .....	91
BCn0, BCn1 (n = 0 to 7) .....	91
BCT .....	79

BCYST .....	42
Block diagram of port .....	316
Block transfer mode .....	154
Boundary of memory area .....	168
Boundary operation conditions .....	96
BPRM0, BPRM1 .....	286
BPRn0, BPRn2 (n = 0, 1) .....	286
BRCE0, BRCE1 .....	286
BRG0, BRG1 .....	283
BRGC0, BRGC1 .....	285
BRGn0 to BRGn7 (n = 0, 1) .....	285
BS .....	290
BSC .....	82
BSn0, BSn1 (n = 0 to 7) .....	82
BTn0, BTn1 (n = 0 to 7) .....	79
Bus access .....	81
Bus arbitration for CPU .....	171
Bus control function .....	77
Bus control pins .....	77
Bus cycle control register .....	91
Bus cycle type configuration register .....	79
Bus cycle type control function .....	79
Bus cycles in which the wait function is valid .....	89
Bus hold function .....	93
Bus hold timing .....	95
Bus priority order .....	96
Bus size configuration register .....	82
Bus sizing function .....	82
Bus width .....	83
Byte access .....	83

### [C]

CALLT base pointer .....	52
Capture/compare registers 1n0 to 1n3 (n = 0 to 3) .....	224
Capture operation (timer 1) .....	237
CBR refresh timing .....	151
CBR self-refresh timing .....	153
CC1n0 to CC1n3 (n = 0 to 3) .....	244
CE .....	290
CE10 to CE13 .....	228
CE40, CE41 .....	230
CES1m0, CES1m1 (n = 0 to 2) .....	227
CESEL .....	209
CG .....	203

CH0 to CH3 .....	147	CTPC .....	52
CKC .....	205	CTPSW .....	52
CKDIV0, CKDIV1 .....	205	CTXE0, CTXE1 .....	273
CKSEL .....	44	CV <sub>DD</sub> .....	45
CL0, CL1 .....	259	CV <sub>SS</sub> .....	45
Clearing/starting timer (timer1) .....	236	CY .....	53
CLKOUT .....	44		
Clock control register .....	205	<b>[D]</b>	
Clock generator .....	203	D0 to D7 .....	38
Clock generator functions .....	203	D8 to D15 .....	39
Clock output inhibit mode .....	215	DA0 to DA15 .....	140
Clock selection .....	204	DA16 to DA25 .....	139
Clocked serial interfaces 0, 1 .....	271	DAC0n, DAC1n (n = 0 to 3) .....	114
Clocked serial interface mode registers 0, 1 .....	273	DAD0, DAD1 .....	143
Clocks of DMA transfer .....	168	DADC0 to DADC3 .....	142
CLS <sub>n</sub> 0, CLS <sub>n</sub> 1 (n = 0, 1) .....	261	Data wait control registers 1, 2 .....	87
CM40, CM41 .....	225	DAW0n, DAW1n (n = 0 to 3) .....	112
CMIC40, CMIC41 .....	189	DBC0 to DBC3 .....	141
CMIF40, CMIF41 .....	189	DBPC .....	52
CMMK40, CMMK41 .....	189	DBPSW .....	52
CMPR40n, CMPR41n (n = 0 to 2) .....	189	DCHC0 to DCHC3 .....	144
CMS1n0 to CMS1n3 (n = 0, 3) .....	227	DCLK0, DCLK1 .....	209
Command register .....	74	DCm0, DCm1 (m = 0 to 7) .....	116
Compare operation (timer 1) .....	240	DDA0 to DDA3 .....	139
Compare operation (timer 4) .....	243	DDIS .....	147
Compare registers 40, 41 .....	225	Dedicated baud rate generators 0, 1 .....	282
Control register (CG) .....	209	Direct mode .....	204
Control register (DMAC) .....	137	DMA addressing control registers 0 to 3 .....	142
Control register (RPU) .....	226	DMA bus states .....	149
Count clock selection (timer 1) .....	234	DMA byte count registers 0 to 3 .....	141
Count clock selection (timer 4) .....	242	DMA channel control registers 0 to 3 .....	144
Count operation (timer 1) .....	233	DMA channel priorities .....	164
Count operation (timer 4) .....	242	DMA controller .....	135
CPC0n, CPC1n (n = 0 to 3) .....	114	DMA destination address registers 0 to 3 .....	139
CPU address space .....	55	DMA disable status register .....	147
CPU function .....	49	DMA functions .....	135
CPU register set .....	50	DMA restart register .....	147
CRXE0, CRXE1 .....	273	DMA source address registers 0 to 3 .....	137
CS .....	290	DMA transfer start factors .....	165
CS0, CS3 to CS5 .....	40	DMA trigger factor registers 0 to 3 .....	145
CSI0, CSI1 .....	271	DMAAK0 to DMAAK3 .....	37
CSIC0, CSIC1 .....	189	DMAC .....	135
CSIF0, CSIF1 .....	189	DMAC bus cycle state transition diagram .....	152
CSIM0, CSIM1 .....	273	DMAIC0 to DMAIC3 .....	189
CSMK0, CSMK1 .....	189	DMAIF0 to DMAIF3 .....	189
CSOT0, CSOT1 .....	273	DMAMK0 to DMAMK3 .....	189
CSPRmn (m = 0, 1, n = 0 to 2) .....	189	DMAPRmn to DMAPRmn (m = 0 to 3, n = 0 to 2) .....	189
CTBP .....	52	DMARQ0 to DMARQ3 .....	35

DRAM access .....	117	FEPSW .....	52
DRAM access during DMA flyby transfer .....	125	Flyby transfer .....	159
DRAM connections .....	111	Flyby transfer data wait control register .....	148
DRAM controller .....	110	FR2 to FR0 .....	292
DRAM configuration registers 0 to 3 .....	113	Frequency measurement .....	250
DRAM type configuration register .....	116		
DRC0 to DRC3 .....	113	<b>[G]</b>	
DRST .....	147	General-purpose registers .....	51
DS .....	142	Global pointer .....	51
DSA0 to DSA3 .....	137		
DTC .....	116	<b>[H]</b>	
DTFR0 to DTFR3 .....	145	Halfword access .....	84
DWC1, DWC2 .....	87	HALT mode .....	207
DWn0 to DWn2 (n = 0 to 7) .....	87	High-speed page DRAM access timing .....	117
		HLD $\overline{\text{DAK}}$ .....	42
<b>[E]</b>		H $\overline{\text{LDRQ}}$ .....	42
EBS0, EBS1 .....	262	HV $\overline{\text{DD}}$ .....	45
Edge detection function .....	181, 193		
ECLR10 to ECLR12 .....	226	<b>[I]</b>	
ECR .....	52	ID .....	53
EDO DRAM access timing .....	121	IDLE .....	209
EICC .....	52	IDLE mode .....	212
EIPC .....	52	Idle state insertion function .....	91
EIPSW .....	52	Idle state insertion timing .....	92
Element pointer .....	51	IFCn5 to IFCn0 (n = 0 to 3) .....	145
EN0 to EN3 .....	144	Illegal op code definition .....	198
ENTO1n0 (n = 0 to 2) .....	231	Image .....	56
EP .....	53	IMS1n0 to IMS1n3 (n = 0 to 3) .....	227
ESmn0, ESmn1 (m = 0 to 3, n = 0 to 3) .....	194	In-service priority register .....	191
ESN0 .....	181	Initialization .....	362
ETI13 .....	228	INIT0 to INIT3 .....	144
Example of DRAM refresh interval .....	129	INTC .....	173
Example of interval factor settings .....	129	Internal block diagram .....	25
Exception trap .....	198	Internal peripheral I/O area .....	61
External bus cycle during DMA transfer .....	163	Internal peripheral I/O interface .....	81
External expansion mode .....	63	Internal RAM area .....	61
External interrupt mode registers 1 to 2, 4 .....	181, 193	Interrupt control register .....	189
External I/O interface .....	99	Interrupt latency time .....	202
External memory area .....	62	Interrupt stack pointer .....	51
External ROM interface .....	99	Interrupt source register .....	52
External wait function .....	88	Interrupting DMA transfer .....	166
		Interrupt/exception processing function .....	173
<b>[F]</b>		Interrupt/exception table .....	61
FDW .....	148	Interval timer .....	245
FDW0 to FDW7 .....	148	INTM0 .....	181
FE0, FE1 .....	263	INTM1, INTM2, INTM4 .....	193, 232
FECC .....	52	INTP100 to INTP103 .....	36
FEPC .....	52	INTP110 to INTP113 .....	37

INTP130 .....	38	Ordering information.....	22
INTSER0, INTSER1 .....	226	OST0 to OST2 .....	226
INTSR0, INTSR1 .....	226	OV .....	53
INTST0, INTST1 .....	226	OVE0, OVE1 .....	263
$\overline{\text{IORD}}$ .....	41	Overflow (timer 1).....	235
$\overline{\text{IOWR}}$ .....	41	Overflow (timer 4).....	242
ISPR .....	191	OVFn (n = 10 to 13, 40, 41).....	232
ISPR0 to ISPR7 .....	191	OVIC10 to OVIC13.....	189
		OVIF10 to OVIF13.....	189
		OVMK10 to OVMK13 .....	189
		OVPR1mn (m = 0 to 3, n = 0 to 2).....	189
<b>[L]</b>			
$\overline{\text{LCAS}}$ .....	41	<b>[P]</b>	
Link pointer.....	51	P0.....	328
$\overline{\text{LOCK}}$ .....	206	P00, P02, P04 to P07 .....	35, 328
LWR .....	41	P1.....	331
		P10.....	353
<b>[M]</b>		P100, P102 .....	43, 353
MA5 to MA3.....	106	P10, P12, P14 to P17 .....	36, 331
Maskable interrupts .....	182	P10IC0 to P10IC3 .....	189
Maskable interrupt status flag.....	191	P10IF0 to P10IF3 .....	189
Maximum response time to DMA request.....	168	P10MK0 to P10MK3.....	189
Memory access control function .....	99	P10PRmn (m = 0 to 3, n = 0 to 2) .....	189
Memory block function.....	78	P11IC0 to P11IC3 .....	189
Memory expansion mode register .....	63	P11IF0 to P11IF3 .....	189
Memory map .....	58	P11MK0 to P11MK3.....	189
MM .....	63	P11PRmn (m = 0 to 3, n = 0 to 2) .....	189
MM3 to MM0 .....	64	P12IC0 to P12IC3 .....	189
MOD0, MOD1 .....	273	P12IF0 to P12IF3 .....	189
MODE0, MODE2 .....	44	P12MK0 to P12MK3.....	189
MS .....	290	P12PRmn (m = 0 to 3, n = 0 to 2) .....	189
Multiple interrupt servicing control .....	200	P13IC0 to P13IC3 .....	189
		P13IF0 to P13IF3 .....	189
<b>[N]</b>		P13MK0 to P13MK3.....	189
Next address setting function .....	164	P13PRmn (m = 0 to 3, n = 0 to 2) .....	189
NMI .....	37	P2.....	334
Noise elimination .....	181, 192	P20, P22 to P27 .....	37, 334
Non-maskable interrupt .....	177	P3.....	337
Normal operation mode .....	54	P33, P34 .....	38, 337
NP .....	53	P4.....	339
Number of access clocks.....	81	P40 to P47 .....	38, 339
		P5.....	341
<b>[O]</b>		P50 to P57 .....	39, 341
$\overline{\text{OE}}$ .....	42	P6.....	343
One time single transfer with $\overline{\text{DMARQ0}}$ to		P60 to P67 .....	39, 343
$\overline{\text{DMARQ3}}$ .....	170	P7.....	345
On-page/off-page judgment.....	106	P70 to P73 .....	40, 345
Operation in A/D trigger mode.....	300	P8.....	346
Operation in timer trigger mode.....	303		
Operation modes.....	54		

P80, P83 to P85 .....	40, 346	PM9.....	351
P9.....	350	PM90 to PM97 .....	351
P90 to P97 .....	41, 350	PMA .....	355
PA .....	355	PMA0 to PMA7 .....	355
PA0 to PA7 .....	43, 355	PMB .....	357
PAE.....	108	PMB0 to PMB7 .....	357
PAE0n, PAE1n (n = 0 to 3) .....	113	PMC0.....	329
Page ROM access .....	109	PMC00, PMC02, PMC04 to PMC07 .....	329
Page ROM configuration register .....	108	PMC1 .....	332
Page ROM controller.....	104	PMC10 (register).....	354
PB .....	357	PMC10, PMC12, PMC14 to PMC17 (bit) .....	332
PB0 to PB7 .....	43, 357	PMC100, PMC102.....	354
PC .....	51	PMC2.....	336
PCS0.....	330	PMC22 to PMC27 .....	336
PCS04 to PCS07 .....	330	PMC3.....	338
PCS1.....	333	PMC33, PMC34 .....	338
PCS14 to PCS17 .....	333	PMC8.....	348
PCS8.....	349	PMC80, PMC83 to PMC85 .....	348
PCS84, PCS85 .....	349	PMC9.....	352
PE0, PE1 .....	263	PMC90 to PMC97 .....	352
Periods in which interrupt is not acknowledged.....	202	PMCX.....	360
Peripheral I/O registers .....	67	PMCX6, PMCX7 .....	360
Pin configuration .....	23	PMX .....	359
Pin functions.....	29	PMX6, PMX7 .....	359
Pin I/O circuit.....	47	Port/control select register 0 .....	330
Pin I/O circuit types .....	48	Port/control select register 1 .....	333
Pin name.....	24	Port/control select register 8 .....	349
Pin status .....	33	Port 0 .....	328
PLL lockup .....	206	Port 1 .....	331
PLL mode.....	204	Port 2 .....	334
PM0.....	328	Port 3 .....	337
PM00, PM02, PM04 to PM07.....	328	Port 4 .....	339
PM1.....	331	Port 5 .....	341
PM10 (register) .....	353	Port 6 .....	343
PM100, PM102 .....	353	Port 7 .....	345
PM10, PM12, PM14 to PM17 (bit).....	331	Port 8 .....	346
PM2.....	335	Port 9 .....	350
PM22 to PM27 .....	335	Port 10 .....	353
PM3.....	338	Port A.....	355
PM33, PM34 .....	338	Port B.....	357
PM4.....	340	Port X.....	359
PM40 to PM47 .....	340	Port functions.....	311
PM5.....	342	Port 0 mode control register.....	329
PM50 to PM57 .....	342	Port 1 mode control register.....	332
PM6.....	344	Port 2 mode control register.....	336
PM60 to PM67 .....	344	Port 3 mode control register.....	338
PM8.....	347	Port 8 mode control register.....	348
PM80, PM83 to PM85 .....	347	Port 9 mode control register.....	352

Port 10 mode control register .....	354	Receive buffers 0, 0L, 1, 1L .....	264
Port X mode control register .....	360	Receive error interrupt .....	266
Port 0 mode register .....	328	Reception completion interrupt .....	266
Port 1 mode register .....	331	Recommended connection of unused pins .....	46
Port 2 mode register .....	335	Refresh control function .....	127
Port 3 mode register .....	338	Refresh control registers 0 to 3 .....	127
Port 4 mode register .....	340	Refresh timing .....	131
Port 5 mode register .....	342	Refresh wait control register .....	130
Port 6 mode register .....	344	REG0 to REG7 .....	74
Port 8 mode register .....	347	Relationship between analog input voltage and	
Port 9 mode register .....	351	A/D conversion results .....	294
Port 10 mode register .....	353	Relationship between programmable wait and	
Port A mode register .....	355	external wait .....	88
Port B mode register .....	357	REN0 to REN3 (DRST register) .....	147
Port X mode register .....	359	RENn (RFCn register) (n = 0 to 3) .....	128
Power saving control .....	207	RESET .....	45
Power save control register .....	209	Reset functions .....	361
PRC .....	108	RFC0 to RFC3 .....	127
PRCMD .....	74	RHC0n, RHC1n (n = 0 to 3) .....	114
Precaution (A/D converter) .....	310	RHD0 to RHD3 .....	114
Precaution (DMA) .....	171	RIn0 to RIn5 (n = 0 to 3) .....	128
Precaution (RPU) .....	252	ROMC .....	104
PRERR .....	75	ROM-less modes 0, 1 .....	63
Priorities of maskable interrupts .....	185	RPC0n, RPC1n (n = 0 to 3) .....	113
PRM1n1 (n = 0 to 3) .....	229	RRW0, RRW1 .....	130
PRM4n0, PRM4n1 (n = 0, 1) .....	230	RWC .....	130
Program counter .....	51	RXB0, RXB0L, RXB1, RXB1L .....	264
Program register set .....	51	RXBn0 to RXBn7 (n = 0, 1) .....	264
Program status word .....	53	RXD0, RXD1 .....	37
Programmable wait function .....	87	RXE0, RXE1 .....	259
PRS1n0, PRS1n1 (n = 0 to 3) .....	234	RXEB0, RXEB1 .....	264
PRS400, PRS410 .....	230		
PRW0 to PRW2 .....	108		
PS00, PS01, PS10, PS11 .....	259		
PSC .....	209		
PSW .....	53		
PWM output .....	248		
PX .....	359		
PX6, PX7 .....	44, 359		
Pulse width measurement .....	246		

**[R]**

r0 to r31 .....	51
$\overline{\text{RAS3}}$ to $\overline{\text{RAS5}}$ .....	40
RCCn0, RCCn1 (n = 0 to 3) .....	128
RCW0 to RCW2 .....	130
$\overline{\text{RD}}$ .....	42
Real-time pulse unit .....	219

**[S]**

S .....	53
SA0 to SA15 .....	138
SA16 to SA25 .....	137
SAD0, SAD1 .....	142
SAT .....	53
Scan mode .....	302
$\overline{\text{SCK0}}$ , $\overline{\text{SCK1}}$ .....	37
SCLS00, SCLS01, SCLS10, SCLS11 .....	259
Securing oscillation stabilization time .....	216
SEIC0, SEIC1 .....	189
SEIF0, SEIF1 .....	189
Select mode .....	304
Self-refresh functions .....	132
SEMK0, SEMK1 .....	189
SEPR0n, SEPR1n (n = 0 to 2) .....	189



Serial I/O shift registers 0, 1 .....	275	Timer control registers 10 to 13 .....	228
Serial interface function .....	255	Timer control registers 40, 41 .....	230
SIO, SI1 .....	37	Timer output control registers 10 to 12 .....	231
Single-step transfer mode .....	154	Timer overflow status register .....	232
Single transfer mode .....	153	Timer trigger mode .....	296
SIO0, SIO1 .....	275	Timer unit mode registers 10 to 13 .....	226
SIO <sub>n</sub> 0 to SIO <sub>n</sub> 7 (n = 0, 1) .....	275	Timer 1 .....	223
SL0, SL1 .....	259	Timer 1 operation .....	233
SO0, SO1 .....	37	Timer 4 .....	225
Software exception .....	195	Timer 4 operation .....	242
Software STOP mode .....	207	Timers 10 to 13 .....	223
SOT0, SOT1 .....	263	Timers 40, 41 .....	225
Specific registers .....	73	Timer/counter function .....	219
SRAM interface .....	99	TM0, TM1 .....	143
SRAM connections .....	99	TM10 to TM13 .....	223
SRIC0, SRIC1 .....	189	TM40, TM41 .....	225
SRIF0, SRIF1 .....	189	TMC10 to TMC13 .....	228
SRMK0, SRMK1 .....	189	TMC40, TMC41 .....	230
SRPR0 <sub>n</sub> , SRPR1 <sub>n</sub> (n = 0 to 2) .....	189	TO100 .....	36
SRW2 to SRW0 .....	130	TO110 .....	37
Stack pointer .....	51	TO120 .....	43
Status saving register during CALLT execution .....	52	TOC10 to TOC12 .....	231
Status saving register during exception trap .....	52	TOVS .....	232
Status saving register during interrupt .....	52	Transfer mode .....	153
Status saving register during NMI .....	52	Transfer objects .....	163
STG0 to STG3 .....	144	Transfer of misalign data .....	168
STIC0, STIC1 .....	189	Transfer types .....	155
STIF0, STIF1 .....	189	Transmission completion interrupt .....	266
STMK0, STMK1 .....	189	Transmit shift registers 0, 0L, 1, 1L .....	265
STP .....	209	TRG0, TRG1 .....	292
STPR0 <sub>n</sub> , STPR1 <sub>n</sub> (n = 0 to 2) .....	189	Trigger mode .....	296
SYS .....	75	TTYP .....	142
System register set .....	52	TUM10 to TUM13 .....	226
System status register .....	75	Two-cycle transfer .....	155
		TXD0, TXD1 .....	37
		TXE0, TXE1 .....	259
		TXED0, TXED1 .....	265
		TXS0, TXS0L, TXS1, TXS1L .....	265
		TXS <sub>n</sub> 7 to TXS <sub>n</sub> 0 (n = 0, 1) .....	265
<b>[T]</b>		<b>[U]</b>	
TBC .....	218	UART0, UART1 .....	256
TBCS .....	209	UCAS .....	41
TC0 to TC3 .....	144	UWR .....	41
TCLR10 .....	36		
TCLR11 .....	37	<b>[V]</b>	
TCLR12 .....	43	V <sub>DD</sub> .....	45
TDIR .....	143	V <sub>SS</sub> .....	45
Terminating DMA transfer .....	166		
TES130, TES131 .....	228		
Text pointer .....	51		
TI13 .....	38		
Time base counter .....	218		

**[W]**

$\overline{\text{WAIT}}$ .....	44
Wait function.....	87
$\overline{\text{WE}}$ .....	42
Word access.....	84
Wrap-around .....	57

**[X]**

X1, X2.....	45
-------------	----

**[Z]**

Z .....	53
Zero register .....	51

## 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: 044-435-9608

### South America

NEC do Brasil S.A.  
Fax: +55-11-6462-6829

### 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"/>